UNIT- I

Graphics Systems and Graphical User Interface: Pixel – Resolution – types of video display devices – Graphical input devices – output devices – Hard copy devices – Direct screen interaction – Logical input function – GKS User dialogue – Interactive picture construction techniques.

INTRODUCTION TO GRAPHICS

DEFINITION:

Graphics are visual presentations on some surface such as wall, canvas, computer screen or paper.

Eg: Photographs, drawings, line art, graphs, diagrams, symbols, maps, geometric designs & engineering drawings.

Graphics can be functional, artistic, realistic or imaginary.

HISTORY:

The earliest graphics known to anthropologists studying of prehistoric periods are cave painting & markings on boulders, bone, antlers during the upper Paleolithic period from 40,000-10,000 BC.

Records from Egypt predate graphics & Papyrus was used by the Egyptians as a material for planning the building of Pyramids.

Greeks played a major role in geometry. Greeks used graphics to represent mathematical memories.

Eg: Circle Theorem, Pythagorean Theorem.

In 1950, one first computer – driven display was attached to MIT's Whirlwind I computer generate simple pictures.

In 1962, Ivan Sutherland invented Sketch pad , an innovative program that influenced alternative forms of interaction with computers. In the mid 1960's large computer graphics research projects were begun at MIT, General Motors, Bell Labs & Lockheed Aircraft. In 1968, Ray Tracing was invented by Appel.

During the late 1970's personal computers were capable of drawing basic & complex shapes.

Modern computer systems dating from the 1980's & onwards used a GUI(Graphical User Interface).

3D graphics became more popular in the 1990's in Gaming, Multimedia and Animation.

i) Graphics is one of the fire key elements of Multimedia Technology.
ii) Quake – one of the first fully 3D games.
ii) 1995, the first fully animated film "TOY STORY "was released in Cinemas worldwide.

USES OF GRAPHICS:

Graphics are often used to point readers and viewers to particular information.

The practical areas where graphics are applied are given below :

i) Business
ii) Advertising
iii) Political News
iv) Education
v) Film & Animation
vi) Graphics Education

PIXEL:

DEFINITION:

A pixel is generally thought of as the smallest complete sample of an image.

(or)

A pixel is a single point in a graphic image.

Each such point (or) information element is not really a dot, nor a square.

Pixels are measured in dpi (dots per inch) (or) ppi (pixels per inch).

HISTORY :

The term Pixel is the Abbreviation for " Picture Element ".

The word pixel was first published in 1965 by Frederic C. Billingsley to describe the picture elements of video images from space probes to the moon & mars.

The word Pix was actually coined in 1932 in a magazine as an abbreviation for the word pictures in reference to movies.

The earliest publications of the term picture element was in "Wireless World " magazine in 1927.

TERMINOLOGIES

Some of the terminologies related to Pixel are given below:

i) Resolution

ii) Subpixel

iii) Megapixel

iv) Bits per inch (BPI)

i) RESOLUTION

The number of pixels in an image is called the Resolution.

Eg: 640 by 480 display

(or)

640* 480 display

ie 640 pixels from side to side & 480 pixels from top to bottom

640 * 480 = 307,200 pixels

(or)

0.3 Megapixels.

ii) BITS PER PIXEL

The number of distinct colours that can be represented by a pixel depends on the number of " bits per pixel" (bpp).

The maximum number of colours a pixel can take can be found by taking two to the power of the colour depth.

Eg: 256 colours = 2^8, 8 bpp

2^16 = 65536 colours = 16 bpp (highcolour or thousands)

2^24 = 16,777,216 colours = 24 bpp (truecolour or millions)

2^48 = 48 bpp (for all practical purposes & in flatbed scanners)

256 colours

Stored in the computers video memory.

Examples :

Animted startup logos of windows 95 & windows 98.

16 bpp

Divided into its RGB components.

ie: 5 bits for Red.
5 bits for Blue &
6 bits for green.

24 bpp

Divided into its RGB components with 8 bits each for Red, Blue & Green.

iii) SUBPIXEL

Many display and image acquisition systems are not capable of displaying (or) sensing the different colour channels at the same site. The above problem is generally resolved by using multiple subpixels, each of which handles a single colour channel either Red, Green (or) Blue.

Example,

i) LCD's typically divide each pixel horizontally into three subpixels.

ii) Most LCD displays divide each pixel into four subpixels ; one Red, one Green and two Blue.

iv) MEGAPIXEL

A megapixel is 1 million pixels.

Eg: 2048 * 1536 pixels = 3.1 megapixel (3,145,728 pixels)

Several other types of object are derived from the idea of pixel namely

- Voxel - volume element
- Texel - texture element
- Surfel - surface element

Have been created for computer graphics & image processing uses.

RESOLUTION:

Resolution is defined as the total number of pixels per image.

DISCUSS ABOUT VIDEO DISPLAY DEVICES. (APRIL / MAY 2012), (APRIL 2013), (NOV 2013)

Typically the primary output device in a graphics system is a video monitor.

The operation of most video monitors is based on the standard cathode-ray tube. The video display devices discussed here are given below:

- Refresh cathode ray tubes
- Raster scan displays
- Random sacn displays
- Color CRT monitors
- Direct view storage tubes
- Flat panel displays &
- Three dimensional viewing devices

- REFRESH CATHODE RAY TUBE

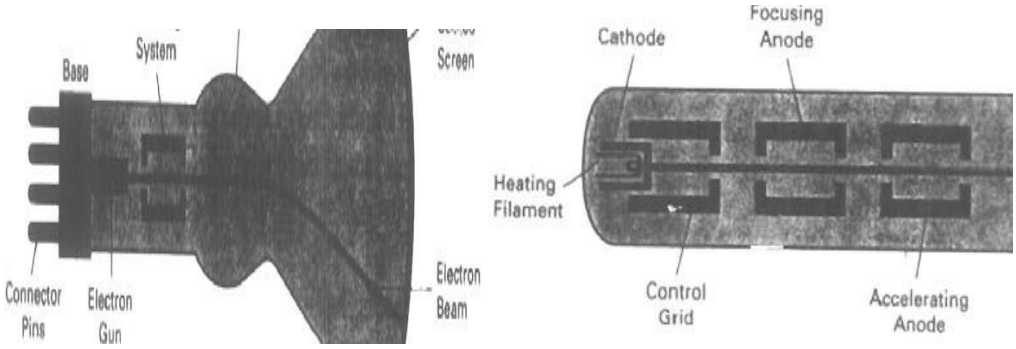
A beam of electrons emitted by an electron gun passes through focusing & deflecting systems that direct the beam toward specified positions on the phosphor-coated screen.

The phosphor then emit a small spot of light at each position contacted by the electron beam.

Because the light emitted by the phosphor fades very rapidly some method is needed for maintaining the screen picture.

One way is to redraw the picture repeatedly by quickly directing the electron beam back over the same points.

This type of display is called a refresh CRT.



WORKING

The primary component of an electron gun are heated metal cathode and a control grid.

Heat is supplied to the cathode by directing a current through a coil of wire called the filament inside the cylindrical cathode structure.

This causes electrons to be boiled off

In the vacuum inside the CRT envelope the free negatively charged electrons are then accelerated toward the phosphor coating by a high positive voltage.

The accelerating voltage can be generated with a +vely charged metal coating on the inside of the CRT near the phosphor screen.

ELECTROSTATIC DEFLECTION

Deflection of the electron beam can be controlled either with electric fields or with magnetic fields.

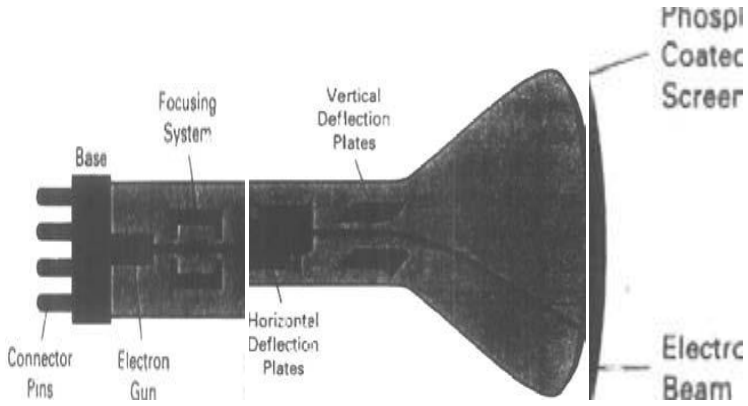
Cathode ray tubes constructed with magnetic deflection coils mounted on the outside of the CRT envelope.

Two pairs of coils are used.

One pair is mounted on the top & bottom of the neck.

Other pair is mounted on the opposite sides of the neck.

The magnetic fields produced by each pair is of coils results in transverse deflection force that is perpendicular both to the direction of the magnetic field & to the direction of travel of the electron beam.



- RASTER SCAN DSIPLAYS

The most common type of graphics monitor employing a CRT is the Raster scan display.

WORKING PRINCIPLE

In a raster scan system the electron beam is swept across the screen one row at a time from top to bottom.

When the electron beam moves across each row the beam intensity is turned on and off to create a pattern of illuminated spots.

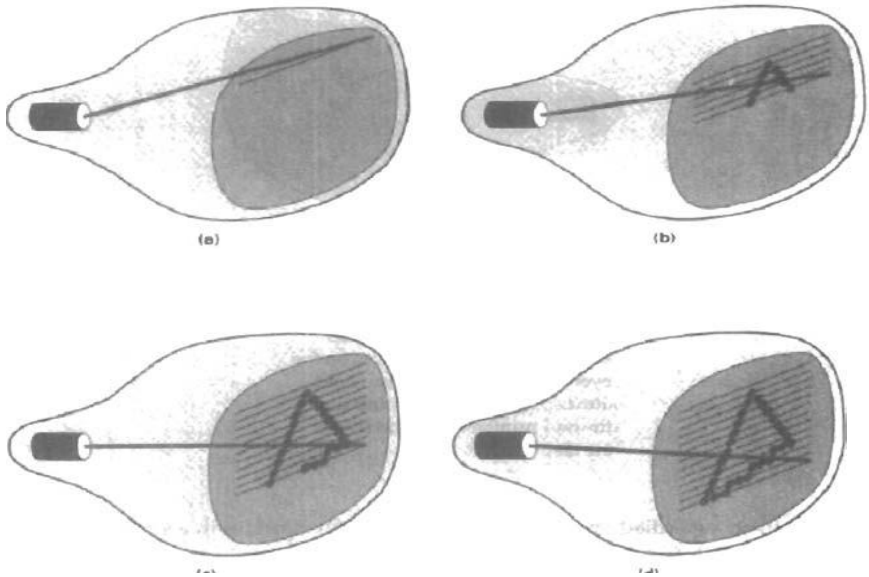
Picture definition is stored in memory area called Refresh buffer (or) Frame buffer.

The frame buffer holds the set of intensity values for all the screen points.

The stored intensity values are then retrieved from the refresh buffer and painted on the screen one row at a time.

One row is also referred to as Scan Line.

Each screen point is referred to as Pixel (or) Pel.



BIT PER PIXEL

Intensity range for pixel positions depend on the capability of the raster system.

In black & white system each screen point is either on (or) off.

- Only one bit per pixel is needed to control the intensity of screen positions.

1 – electron beam is turned on

0 – electron beam is turned off

High quality system use upto 24 bits per pixel. The frame buffer of such system require several MB's of storage for the frame buffer.

REFRESHING

Refreshing of Raster scan displays is carried out as the rate of 60 to 80 frames per second.

Refresh rates are described in units of cycles per second or Hertz.

1 cycle – 1 frame

At the end of each scan line the electron beam returns to the left side of the screen to begin displaying the next scan line.

The return of the electron beam to the left of the screen is called the horizontal retrace.

At the end of each frame the electron beam returns to the top left corner of the screen.

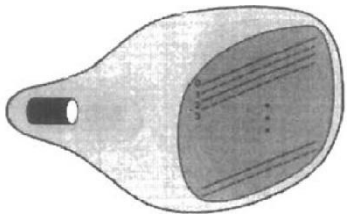
This is referred to as vertical retrace.

- RANDOM – SCAN SYSTEMS

In a Random- scan display unit a CRT has the electron beam directed only to the parts of the screen where a picture is to be drawn.

Random scan monitors draw a picture one line at a time & for this reason they are referred to as vector displays.

The component lines of a picture can be drawn and refreshed by as random scan system in any specified order.



REFRESH RATE

Refresh rate depends on the number of lines to be displayed.

A refresh buffer is used to store picture definition. The line drawing commands are stored in the refresh buffer.

To display a specified picture the system processes the set of drawing commands.

Random scan displays are designed to draw all the components lines of a picture 30 to 60 times each second.

Random scan system are designed for line drawing applications and cannot display realistic shaded scenes.

Random scan displays produce smooth line drawings because the CRT beam directly follows the line path.

- COLOR CRT MONITORS

A CRT monitor displays color pictures by using a combination of phosphors that emit different colored light.

Two basic techniques are used for producing color display with a CRT namely

i) Beam penetration method.

ii) Shadow mask method.

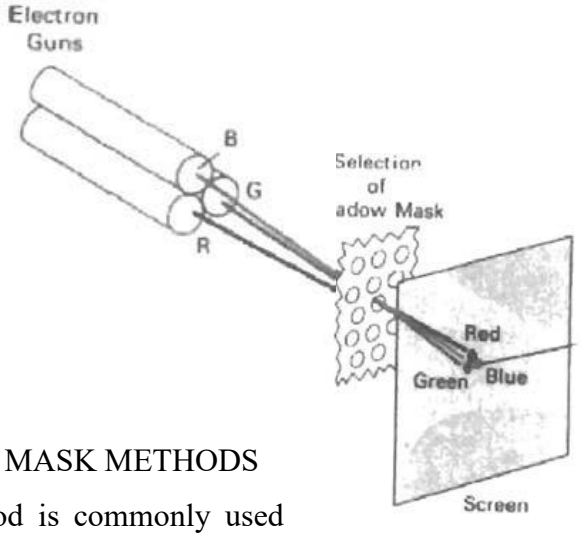
BEAM PENETRATION METHOD

This method is used with random scan monitors .

Here two layers of phosphor namely red & green are coated onto the inside of the CRT screen.

The displayed color depends on how far the electron beam penetrates into the phosphor layers.

- A slow beam of electrons penetrates through the screen & excites only the outer red layer.
- A beam of very fast electrons penetrates through the red layer and excites the inner green layer.
- An intermediate beam speed produces orange & yellow color (combinations of red and green lights)
- The speed of the electrons & hence the screen color at any point is controlled by the beam acceleration voltage.



SHADOW MASK METHODS

This method is commonly used in Raster scan systems.

A shadow mask CRT has three phosphor color dots at each pixel position.

- One for red light, one for green light & one for blue light.

This type of CRT has three electron guns, one for each color dot.

This CRT also has a shadow mask grid just behind the phosphor coated screen. The three electron beams are deflected & focused as a group onto the shadow mask. The shadow mask are aligned with the phosphor dot patterns.

When the beams pass through the holes in the shadow mask they activate a dot triangle.

Another configuration for the electron guns is an in-line arrangement where the 3 electron beams are aligned to a single scan line. By varying the intensity of levels of the three electron beams various color variations are obtained.

The color got depends on the amount of red, green and blue phosphors.

- A white area indicates that all the three electron beams are with the same intensity.

COLOR COMBINATIONS

Yellow – green & red beams

Magenta - blue & red beams

Cyan – blue & green beams

More sophisticated systems can set intermediate intensity levels for the electron beams.

- DIRECT VIEW STORAGE TUBES

An alternative method for maintaining a screen image is to store the picture information inside the CRT instead of refreshing the screen.

Two electron gns are used in DVST

DVST stores the picture information as a change distribution just behind the phosphor coated screen.

The primary electron gun stores the picture pattern.

The flood gun maintains the picture display.

ADVANTAGES

- No refreshing is needed.
- Very complex pictures can be displayed at very high resolutions without flicker.

DISADVANTAGES

- Selected parts of a picture cannot be erased.
- They ordinarily do not display color.

- FLAT PANEL DISPLAY

The term refers to a class of video devices that have reduced volume, weight & power requirements. A significant feature of that panel displays is that they are thinner than CRT's. Flat panel displays are available as pocket notepads.

Uses of flat panel displays are

- TV monitors
- Calculators
- Pocket video games
- Laptop computers
- Armrest viewing of movies on airplanes.
- Advertisement board in elevators

CATEGORIES

There are two major categories of flat panel displays

i) Emissive displays

ii) Non emissive displays

Emissive displays

Convert electrical energy into light.

Examples

i) Plasma panels

ii) Thin film electroluminescent displays

iii) Light emitting diodes.

PLASMA PANELS

Also called as Gas-discharge displays. Plasma panels are constructed by filling the region between two glass plates with a mixture of gases that usually include Neon. A series of vertical plates is placed in one glass panel. A series of horizontal plates is built into the other glass panel. When firing voltages are applied to a pair of horizontal & vertical conductors the gas at the intersection of the plates breaks down into a glowing plasma of electrons & ions. Alternating current AC methods are used to provide faster application of the firing voltages & thus brighter

displays. One advantage of plasma panels was that they were monochromatic devices but systems have been developed that are now capable of displaying color & grayscale.

THIN FILM ELECTROMAGNETIC DISPLAYS

These displays are similar in construction to a plasma panel. The difference is that the region between the glass plates is filled with a phosphor , such as Zinc Sulphide doped with manganese instead of a gas. When a high voltage is applied to a pair of crossing electrodes the phosphor becomes a conductor in the area of intersection of the electrodes.

The manganese atoms absorb the electrical energy as a spot of light. Electroluminescent displays require more power than plasma panels. Good color & gray scale displays are hard to achieve.

LIGHT EMITTING DIODE (LED)

A matrix of diodes is arranged to form the pixel positions in the display.

Picture definition is stored in a refresh buffer.

Information is read from the refresh buffer & converted to voltage levels that are applied to the diodes to produce the light patterns in the display.

i) non emissive displays

These displays use optical effects to convert sunlight from some other source into graphics patterns.

Examples

Liquid crystal device.

- LIQUID CRYSTAL DEVICE (LCD's)

These displays are commonly used in small systems such as calculators & portable laptop computers. The term liquid crystal refers to compounds which have a crystalline arrangement of molecules. However it flows like liquid.

Flat panel displays use nomadic (thread like) liquid crystal compounds. The liquid crystal material is sandwiched between two glass plates each containing a light polarizer at right angles to the plate.

Horizontal conductors are built into one glass plate & vertical conductors are built into another glass plate.

The intersection of the two conductors defines a pixel position. Polarized light passing through the material is twisted so that it will pass through the opposite polarizer. The light is then reflected back to the viewer. This type of flat panel device is referred to as a passive matrix LCD.

Another method for constructing LCD's is to place the transistor at each pixel location using thin- film transistor technology. These devices are called Active – Matrix LCD.

THREE DIMENSIONAL VIEWING DEVICES

Graphics monitors for the display of three dimensional scenes have been devised using a technique that reflects a CRT image from a vibrating, flexible mirror. As the mirror vibrates it changes focal length. These vibrations are synchronized with the display of on a CRT so that each point on the object is reflected from the mirror into a spatial position corresponding to the distance of that point from a specified viewing position. This allows a person to walk around an object or scence and view it from different sides.

Real – time example

- Geisco space graph system – used in medical applications
- For analyzing data from ultrasonography & CAT scan devices.
- In geological applications to analyze topographical & seismic data.

STEREOGRAPHIC & VIRTUAL REALITY SYSTEMS

Stereoscopic views – alternate method for displaying 3D objects.

This method provides a 3D effect by presenting a different view to each eye of an observer. To obtain stereoscopic projection the first step is to obtain two view of a scene.

The two views of a scene are generated from a viewing direction corresponding to each eye (left & right). Simultaneous viewing of each view merges into a single image.

The screen is viewed with glasses (specially designed) Stereoscopic viewing is also a component in virtual-reality systems where users can step into a scene & interact with the environment.

A headset containing an optical system to generate the stereoscopic views is commonly used in conjunction with interactive i/p devices to locate & manipulate objects in the scene. A sensing system in the headset keeps track of the viewers position so that the front & back of objects can be seen as the viewer walks through.

LIST OUT AND DISCUSS THE INPUT DEVICES. (APR 2013) (NOV 2013) (APR 2014)

Various devices are available for data input on graphics workstations.

Most systems have a keyboard & one or more additional devices specially designed for interactive input.

The various input devices that are discussed in this section are

i) keyboards
ii) mouse
iii) trackball & spaceball
iv) joysticks
v) data glove
vi) digitizers
vii) image scanners
viii) touch panels
ix) light pens
x) voice systems

KEYBOARDS

The keyboard is an efficient device for inputting such nongraphic data as picture labels associated with a graphic display.

Alphanumeric keyboard

Keyboards are provided with features to facilitate entry of screen co-ordinates, menu selections or graphics functions.

The common features on general purpose keyboards are

i) cursor-control keys
ii) function keys

cursor control keys:

It can be used to select displayed objects or co-ordinate positions by positioning the screen cursor.

functional keys:

This allow users to enter frequently used operations in a single keystroke. Additionally a numeric keypad is often included on the keyboard for fast entry of numeric data.

MOUSE

A mouse is a small hand held device used to position the screen cursor. Wheels or rollers on the bottom of the mouse can be used to record the amount & direction of movement.

Another method for detecting mouse movements is with an optical sensor. One,two or three buttons are usually included on the top of the mouse for signaling the execution of some operation.

Z-MOUSE

Additional devices can be included in the basic mouse design to increase the number of allowable input parameters. The z-mouse includes three buttons ,a thumbwheel on the side, a trackball on the top & a standard mouse ball underneath. With the z-mouse , one can pick up an object , rotate it and move it in any direction or the navigation of one's viewing position & ordination through a 3D scene.

Application of z-mouse are

- virtual reality
- CAD
- Animation

TRACKBALL AND SPACEBALL

TRACKBALL

Trackball is a ball that can be rotated with thr fingers or palm of the hand to produce screen cursor movements.

Potentiometers attached to the ball measure the amount & direction of rotation.

They are often mounted on keyboard or z-mouse.

SPACEBALL

A spaceball provides size degrees of freedom.

A spaceball does not actually move. Strain gauges measure the amount of pressure applied to the spaceball.

Applications where spaceballs are used are

i) 3D positioning

ii) Virtual reality systems

iii) Modeling

iv) Animation

v) CAD

JOYSTICKS

A joystick consists of a small vertical lever called the stick. The stick is mounted on a base which steers the screen around.

There are 2 types of joystick

i) movable stick joystick

ii) non-movable stick joystick

MOVABLE STICK JOYSTICK

The stick is actually placed in the center position. Screen cursor movement is achieved by moving the stick in any direction from the center. Potentiometers also mounted at the base of the joystick & they measure the amount of movement. Potentiometers also return the stick to the center position when released.

In another type of movable joystick the stick is used to activate switches that cause the screen cursor to move at a constant rate in the selected direction.

NON MOVABLE STICK JOYSTICk

These joysticks are also called isometric joysticks. They have a non movable stick & pressure applied on the joystick is measured by strain gauges & converted to the movement of the cursor in the specified direction.

DATA GLOVE

Data glove can be used to grasp a virtual object. The glove has a series of sensors that detect hand & finger movements. Electromagnetic coupling between transmitting antennas & receiving antennas is used ot provide information about the position & orientation of the screen. Input from the glove can be used to position or manipulate objects in a virtual scene. A 2D projection of the scene can be viewed using video monitor. A 3D projection can be viewed with a headset.

DIGITIZERS

A digitizer is a common device for drawing , painting or interactively selecting co-ordinating positions on an object. Digitizers are used to input co-ordinate values in either a 2D or 3D space. A digitizer is used to scan over a drawing or object & to input a set of discrete co-ordinate positions. Graphics tablet is an example of digitizer.

IMAGE SCANNERS

An image scanner is used for storing drawings , graph , color & black and white photos or text by passing an optical scanning mechanism for computer processing. The gradations of gray scale or color are then recorded and stored in an array. Transformations can be applied to rotate , scale or crop the picture to a particular screen area. Various image processing methods can be applied to modify the arrayrepresentations or text & they come in a variety of sizes & capabilities.

TOUCH PANELS

Touch panel allow displayed objects or screen positions to be selected with the touch of a finger.A typical application of touch panel's is for the selection of processing options that are represented with graphical icons.

Eg ATM center touch panel

Touch input can be recorded using optical, electrical or acoustical methods.

OPTICAL TOUCH PANELS

These panels employ a line of infrared light emitting diodes (LED's) along one vertical edge & along one horizontal edge of the frame. The opposite vertical edge & horizontal edge contain light detectors. When the panel is touched these light detectors record which beams are interrupted. The 2 crose beams that are interrupted identify the horizontal & vertical co-ordnates of the screen position selected. Positions can be selected with an accuracy of about ¼ inch. The LED's operate at infrared frequencies so that the light is not visible to a user.

ELECTRICAL TOUCH PANEL

These plates are constructed with 2 transparent plates separated by a small distance. One plate is coated with a conducting material & the other with resistive material. Touching the outer plate forces it into contact with the inner plate. This contact creates a voltage drop across the resistive plate that is converted into co-ordinate values of the selected screen position.

ACOUSTICAL TOUCH PANEL

Here high frequency sound waves are generated by the horizontal & vertical directions across a glass plate. Touching the screen causes part of each wave to be reflected from the fingers to the emitters. The screen position is calculated from a measurement of the time interval between the transmission of each wave & its reflection to the emitter.

LIGHT PEN

Light pens are pencil-shaped devices used to select screen positions by detecting the light coming fro points on the CRT screen. Light pens are sensitive to short burst of light emitted from the phosphor coating of the CRT. Other light sources are not usually detected by a light pen. An activated light pen pointed at a spot on the screen as the electron beam lights up that spot generates an electric pulse that causes the co-ordinate position of the electron beam to be recorded. DISADVANTAGES

i) Screen image obscured by hand & light pen
ii) Prolonged use causes arm fatigue
iii) Light pen require special implementation for some applications.
iv) Sometimes light pens give false readings due to background lighting in room.

VOICE SYSTEMS

Speech recognizers are used in some graphics workstations as input devices to accept voice commands. The voice system input can be used to initiate graphics operations or to enter data. These systems operate by matching an input against a predefined dictionary of words and pharses.

DICTIONARY

A dictionary is set up for a particular operator. The operator speak the command words to be used into the system. Each word is spoken several times & the system analyzes the word & establishes a frequency pattern for that word in the dictionary. Later when a voice command is given the system searches the dictionary for a frequency pattern match

Voice input is typically spoken into a microphone mounted on a headset. The microphone is designed to minimize input of other background sounds.

LIST OUT AND DISCUSS THE OUTPUT DEVICES. (APR 2012) (NOV 2012)

Hard copy output for images can be obtained in several formats. Users can put pictures on paper by directing graphics output to a printer or plotter. The quality of pictures obtained from a device depends on the dot size & dot per inch or lines per inch. Smooth characters can be produced in printed text strings by high quality printers. These printers shift dot positions so that adjacent dots overlap. There are 2 major methods or types of printers namely

i) impact devices &

ii) non impact devices

IMPACT DEVICES

Impact printers press formed character faces against an inked ribbon onto paper

A typeface is used to make the impression. Typefaces are normally mounted on bands , chains , drums or wheels.

Eg dot matrix printer

DOT MATRIX PRINTER

It is an example of impact printer

Dot matrix printers have a dot matrix print head containing a rectangular array of protruding pins. The no. of pins depends on the quality of the printer. Individual characters or graphic patterns are obtained by retracting certain pins so that the remaining pins form the pattern to be printed

NON IMPACT DEVICES

Non impact plotters and printers use laser techniques , ink jet sprays , xerographic processes, electrostatic methods & electro thermal methods to get images on the paper

LASER PRINTER

In a laser device , a laser beam creates a charge distribution on a rotating drum coated with a photoelectric material such as selenium Toner is applied to the drum and then transferred to paper. The quality of the printout depends upon the dots per inch (dpi)

INK JET PRINTER

These printers produce output by squirting ink in horizontal rows across a roll of paper wrapped on a drum. The electrically charged ink stream is deflected by an electric field to produce dot matrix patterns.

ELECTROSTATIC PRINTER

These printers place a negative charge on the paper one complete row at a time along the length of the paper. The paper is then exposed to the toner. The toner is positively charged and is attracted to negative charged areas on the paper.

COLOR OUTPUT ON IMPACT AND NON IMPACT PRINTERS

Limited color output can be obtained using non impact printers by using different color ribbons. Various techniques are used by non impact printers to combine three color pigments (cyan, magenta & yellow). Laser & xerographic devices deposit the three pigments on separate passes. Ink jet methods shoot the three colors simultaneously on a single pass along each print line on the paper.

PEN PLOTTER

A pen plotter has one or more pens mounted on a carriage or across bar that spasm a sheet of paper. Pens with varying color and widths are used to produce a variety of shadings & line styles. Wet ink , ball point and felt tip pens are all possible choices for use with a pen plotter. Pen plotter paper can lie flat or be rolled onto a drum or belt

Crossbars can be either movable or stationary. The paper is held in position by using clamps , vaccum or by an electrostatic charge.

GRAPHICS SOFTWARE

There are 2 general classifications for graphics software namely,

i) general programming graphic packages
ii) special purpose application packages

GENERAL PROGRAMMING GRAPHIC PACKAGES

This package provides an extensive set of graphics functions that can be used in high level programming languages

An example is GL. GL – Graphics Library System. Some basic functions in GL

- Straight lines
- Polygons
- Circles
- Setting color & intensity values
- Selecting views
- Applying transformations

Graphics programming languages needs programming capability by the user

APPLICATION GRAPHIC PACKAGES

These packages are designed for non programmers

Users can generate displays without worrying about how graphics operation work

The interface in graphics packages allow users to communicate with the programs

- An artist's painting programs
- Business applications
- Medical applications
- CAD systems

CO-ORDINATE REPRESENTAIONS

Co-ordinate values for a picture must be converted to Cartesian co-ordinates before they can be input to the graphics package

The various types are

- Modelling coordinates
- World coordinates
- Device coordinates

MODELLING COORDINATES

The shape of individual objects in a scene can be constructed using separate coordinate references frames called modelling coordinates. Modeling coordinates are also called as local coordinates or master coordinates

WORLD COORDINATES

Once individual object shapes have been specified the objects can be placed into appropriate positions within the scene. The object can be placed in the scene using reference frame called world coordinates

DEVICE COORDINATES

The world coordinate description of the scene is transferred to one or more o/p references frames for display. This is also known as device coordinates or screen coordinates or normalized device coordinates.

GRAPHICS FUNCTIONS

A general purpose graphics packages provides users with a variety of functions for creating and manipulating pictures. These functions are categorized according to whether they deal with outputs , input , attributes, transformations, viewing or general control

i) Output primitives

These are the basic building blocks for pictures. They included character strings, points , straight lines etc

ii) Attributes

These are the properties of the output primitives. They specify how an output primitive is to be displayed. They include intensity & color specifications, line styles , text styles & area filling patterns

iii) Geometric transformations

The size , position or orientation of an object can be changed within a scene using this

Modeling transformation are used to construct a scene using object description

iv) Structures

Also called as segments or objects. Pictures are subdivided into components parts called structures. Each structure defines one logical unit of the picture.

Routines for processing structures carry out operations such as the creation, modification and transformation of structures.

v) Input functions

Are used to control & process the data flow from interactive devices

vi) Control operations

Are used to do housekeeping task

- Cleaning a display screen
- Initializing parameters

SOFTWARE STANDARDS

The primary goal standardized graphics software is portability

When packages are designed with standard graphics functions , software can be moved easily from one hardware system to another

Standards discussed are

- Graphical Kernel Systems (GKS)
- Programmers Hierarchial interactive Graphics Standard (PHIGS)

GRAPHICS KERNEL SYSTEM (GKS):

The GKS was the first ISO standard for low level computer graphics , introduced in 1977

GKS provides a set of low level drawing feature for 2D line & vector graphics

Applications written using GKS will be readily portable to many platform & devices

GKS has 2 major advantages over other graphics packages

- international standard & real portability of graphics software is possible
- NCAR package
- Device independent graphics programs can be written using GKS

GKS is 2D graphics packages

PROGRAMMERS HIERARCHIAL INTERACTIVE GRAPHICS STANDARD (PHIGS)

PHIGS is an extension of GKS

Increased capabilities for object modeling ,color specifications, surface rendering & picture

PHIGS+ was developed to provide 3D surface capabilities not available in PHIGS

Standard graphics functions are defined as a set of specification that is independent of any programming language. A language binding is then defined for a particular programming language . This binding gives the syntax for accessing the various standard graphics functions

PHIGS WORKSTATIONS

Used to identify various combinations of graphics s/w & h/w
Eg single i/p device, i/p &o/p devices , file , window displayed on a monitor

GENERAL STRUCTURE OF PHIGS PROGRAM

Open phigs(error file , memory size)
Open workstation(ws, connection, type)
{
Create and display picture
}
Close workstation (WS)
Close phigs
Where,

Errorfile - parameter which contains any error message generated
Memorysize – parameter that specifies the size of an internal storage area
WS – Workstation identifier
Connection – states access mechanism
Type – specifies the category for the workstation
Eg i/p devices , o/p devices or i/p & o/p devices.

PONDICHERRY UNIVERSITY QUESTIONS
GRAPHICS AND IMAGE PROCESSING

1. Discuss about Video Display Devices. **(APR 2012) (APRIL 2013) (NOV 2013) (Pg.no.6)**

2. List out and discuss the Output devices. **(APR 2012) (NOV 2012) (Pg.no.19)**
3. Discuss the function of raster-scan and random-scan display. **(NOV 2012) (Pg.no.8)**
4. List out and discuss the input devices. **(APR 2013) (NOV 2013) (APR 2014) (Pg.no.15)**

UNIT II

Geometric Display Primitives and Attributes: Geometric display primitives – Points – Lines and Polygons – Point display method – Line drawing methods.
2D Transformations and Viewing: Transformations – types – matrix representation – Concatenation – Scaling – Rotation – Translation – Shearing – Mirroring – Homogeneous coordinates.
Window to view port transformations: Windowing And Clipping: Point – Lines – Polygons - boundary intersection methods.

DISPLAY PRIMITIVES AND ATTRIBUTES

LINE DRAWING ALGORITHM

Line drawing is our first adventure into the area of scan conversion. The need for scan conversion, or rasterization, techniques is a direct result of scanning nature of raster displays.

The goal of any line drawing algorithm is to construct the best possible approximation of an ideal line given the inherent limitations of a raster display. Following is a list of some of line qualities that are often considered.

- Continuous appearance
- Uniform thickness and brightness
- Are the pixels nearest the ideal line turned on
- How fast is the line generated

The Bresenham line algorithm is an algorithm that determines which points in an n-dimensional raster should be plotted in order to form a close approximation to a straight line between two given points. It is commonly used to draw lines on a computer screen, as it uses only integer addition, subtraction and bit shifting all of which are very cheap operations in standard computer architectures. It is one of the earliest algorithms developed in the field of computer graphics.

Through a minor expansion, the original algorithm for lines can also be used to draw circles. Also this can be done with simple arithmetic operations; quadratic or trigonometric expressions can be avoided or recursively dissolved into simpler steps.

The common conventions that pixel coordinates increase in the down and right directions and that pixel centers have integer coordinates will be used. The endpoints of the line are the pixels at (x_0, y_0) and (x_1, y_1), where the first coordinate of the pair is the column and the second is the row.

The algorithm will be initially presented only for the octant in which the segment goes down and to the right ($x_0 \le x_1$ and $y_0 \le y_1$) , and its horizontal projection $x_1 - x_0$ is longer than the vertical projection $y_1 - y_0$ (in other words, the line has a slope less than 1 and greater than 0.) In this octant, for each column x between x_0 and x_1, there is exactly one row y (computed by the algorithm) containing a pixel of the line, while each row between y_0 and y_1 contains multiple rasterized pixels.

Bresenham's algorithm chooses the integer y corresponding to the pixel center that is closest to the ideal (fractional) y for the same x; on successive columns y can remain the same or increase by 1. The general equation of the line through the endpoints is given by:

$$y - y_0 = \frac{y_1 - y_0}{x_1 - x_0}(x - x_0).$$

Since we know the column, x, the pixel's row, y, is given by rounding this quantity to the nearest integer:

$$\frac{y_1 - y_0}{x_1 - x_0}(x - x_0) + y_0.$$

The slope $(y_1 - y_0) / (x_1 - x_0)$ depends on the endpoint coordinates only and can be precomputed, and the ideal y for successive integer values of x can be computed starting from y_0 and repeatedly adding the slope.

In practice, the algorithm can track, instead of possibly large y values, a small error value between −0.5 and 0.5: the vertical distance between the rounded and the exact y values for the current x. Each time x is increased, the error is increased by the slope; if it exceeds 0.5, the rasterization y is increased by 1 (the line continues on the next lower row of the raster) and the error is decremented by 1.0.

In the following pseudocode sample `plot(x,y)` plots a point and `abs` returns absolute value:

```
function line(x0, x1, y0, y1)
    int deltax := x1 - x0
    int deltay := y1 - y0
    real error := 0
    real deltaerr := deltay / deltax    // Assume deltax != 0 (line is not vertical)
    int y := y0
    for x from x0 to x1
```

```
plot(x,y)
error := error + deltaerr
if abs(error) ≥ 0.5 then
   y := y + 1
   error := error - 1.0
```

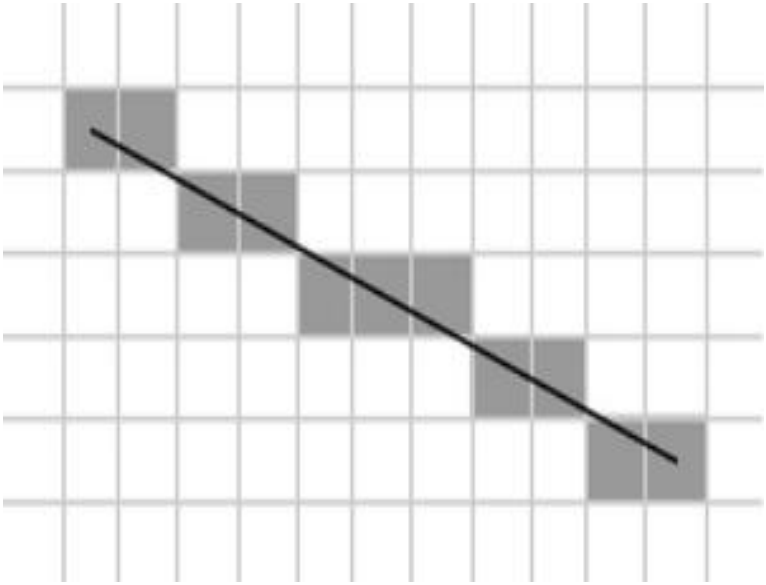


Illustration of the result of Bresenham's line algorithm.

The basic Bresenham algorithm

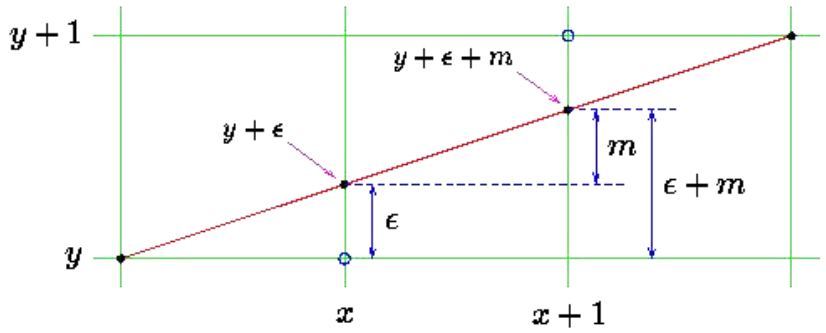
Consider drawing a line on a raster grid where we restrict the allowable slopes of the line to the range $0 \leq m \leq 1$.

If we further restrict the line-drawing routine so that it always increments x as it plots, it becomes clear that, having plotted a point at (x,y), the routine has a severely limited range of options as to where it may put the next point on the line:

- It may plot the point (x+1,y), or:
- It may plot the point (x+1,y+1).

So, working in the first positive octant of the plane, line drawing becomes a matter of deciding between two possibilities at each step.

We can draw a diagram of the situation which the plotting program finds itself in having plotted (x,y).



In plotting (x,y) the line drawing routine will, in general, be making a compromise between what it would like to draw and what the resolution of the screen actually allows it to draw. Usually the plotted point (x,y) will be in error, the actual, mathematical point on the line will not be addressable on the pixel grid. So we associate an error, ϵ, with each y ordinate, the real value of y should be $y+\epsilon$. This error will range from -0.5 to just under +0.5.

In moving from x to x+1 we increase the value of the true (mathematical) y-ordinate by an amount equal to the slope of the line, m. We will choose to plot (x+1,y) if the difference between this new value and y is less than 0.5.

$$y+\epsilon+m<y+0.5$$

Otherwise we will plot (x+1,y+1). It should be clear that by so doing we minimise the total error between the mathematical line segment and what actually gets drawn on the display.

The error resulting from this new point can now be written back into ϵ, this will allow us to repeat the whole process for the next point along the line, at x+2.

The new value of error can adopt one of two possible values, depending on what new point is plotted. If (x+1,y) is chosen, the new value of error is given by:

$$\epsilon_{new} \leftarrow (y+\epsilon+m)-y$$

Otherwise it is:

$$\epsilon_{new} \leftarrow (y+\epsilon+m)-(y+1)$$

This gives an algorithm for a DDA which avoids rounding operations, instead using the error variable ϵ to control plotting:

$\epsilon \leftarrow 0, \quad y \leftarrow y_1$

For $x \leftarrow x_1$ **to** x_2 **do**

  Plot point at (x, y).

  If ($\epsilon + m < 0.5$)

    $\epsilon \leftarrow \epsilon + m$

  Else

    $y \leftarrow y + 1, \quad \epsilon \leftarrow \epsilon + m - 1$

  EndIf

EndFor

This still employs floating point values. Consider, however, what happens if we multiply across both sides of the plotting test by Δx and then by 2:

$$\begin{aligned} \epsilon + m &< 0.5 \\ \epsilon + \Delta y/\Delta x &< 0.5 \\ 2\epsilon\Delta x + 2\Delta y &< \Delta x \end{aligned}$$

All quantities in this inequality are now integral.

Substitute ϵ' for $\epsilon\Delta x$. The test becomes:

$$2(\epsilon' + \Delta y) < \Delta x$$

This gives an integer-only test for deciding which point to plot.

The update rules for the error on each step may also be cast into ϵ' form. Consider the floating-point versions of the update rules:

$$\begin{aligned} \epsilon &\leftarrow \epsilon + m \\ \epsilon &\leftarrow \epsilon + m - 1 \end{aligned}$$

Multiplying through by Δx yields:

$$\begin{aligned} \epsilon\Delta x &\leftarrow \epsilon\Delta x + \Delta y \\ \epsilon\Delta x &\leftarrow \epsilon\Delta x + \Delta y - \Delta x \end{aligned}$$

which is in ϵ' form.

$$\epsilon' \leftarrow \epsilon' + \Delta y$$
$$\epsilon' \leftarrow \epsilon' + \Delta y - \Delta x$$

Using this new ``error'' value, ϵ', with the new test and update equations gives Bresenham's integer-only line drawing algorithm:

$\epsilon' \leftarrow 0, \quad y \leftarrow y_1$
For $x \leftarrow x_1$ **to** x_2 **do**
    Plot point at (x, y).
    If ($2(\epsilon' + \Delta y) < \Delta x$)
        $\epsilon' \leftarrow \epsilon' + \Delta y$
    Else
        $y \leftarrow y + 1, \quad \epsilon' \leftarrow \epsilon' + \Delta y - \Delta x$
    EndIf
EndFor

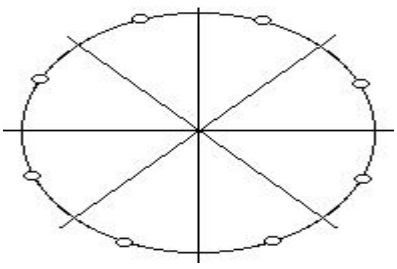
- Integer only - hence efficient (fast).
- Multiplication by 2 can be implemented by left-shift.
- This version limited to slopes in the first octant, $0 \leq m \leq 1$.

ALGORITHM FOR GENERATING CIRCLE

Many incremental methods exist to plot circles and arcs.
1) If the hardware exists for line generation, then circles are generated using short line segments.
2) If the line generation hardware does not exist, then circles are generated using closely spaced dots.

Central Symmetry

A usable fact is that if the center of the circle is at location (0,0) and a point (x,y) lies on the circumference of the circle, then we know 7 more points on the circle.



We can set eight symmetric pixels that correspond to a point (x,y) on the circle using the following procedure:

```
procedureset_eight(x,y,xcenter,ycenter:integer);
begin
plot_pixel(x+xcenter,y+ycenter);
plot_pixel(-x+xcenter,y+ycenter);
plot_pixel(x+xcenter,-y+ycenter);
plot_pixel(-x+xcenter,-y+ycenter);
plot_pixel(y+xcenter,x+ycenter);
plot_pixel(-y+xcenter,x+ycenter);
plot_pixel(y+xcenter,-x+ycenter);
plot_pixel(-y+xcenter,-x+ycenter);
end;
```

Circle Generating Algorithms

Non-DDA Methods

1) Parameter Method - this method consists of calculating values (x,y) on the circle using the following formulas:

x=rcos(u)
y = rsin(u)

where u increases incrementally by small steps over the interval 0 to 2(pi). We can use symmetry to improve the execution time so that u is calculated over the interval 0 to pi/4. We also note that if the incremental value of u is <= 1/r, where r is the length of r expressed in pixels, the circle generator advances by approximately one pixel each iteration. This algorithm is slow because of the constant recalulation of sin(u) and cos(u).

are generating a circle from the origin with radius 10. This would imply the following table of values:

u	X=rcos(u)	Y=rsin(u)
0	10	0
.1	9.95	.999
.2	9.8	1.99
.3	9.55	2.96

.4	9.21	3.89
.5	8.78	4.79
.6	8.25	5.65
.7	7.65	6.44
.8	6.97	7.17 (Note: pi/4 = .7854)
....		
1.5708	0	10 (Note: pi/2 = 1.5708)

2) Rotation Method - this method consists of using the point (x,y) on the circle and rotating it about the origin. The following equations will produce the desired result:

a) Clockwise

Xrot=Xcos(u)+Ysin(u)

Yrot = -Xsin(u) + Ycos(u)

b) Counterclockwise

Xrot=Xcos(u)-Ysin(u)

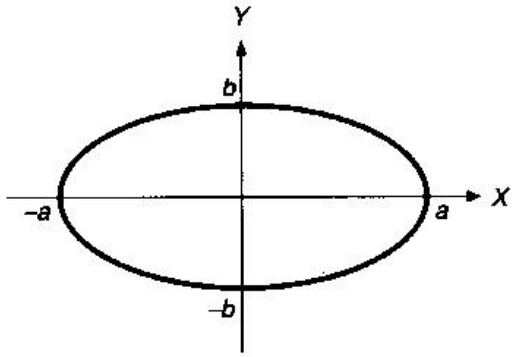
Yrot = Xsin(u) + Ycos(u)

To use the rotation method, we assign (X0,Y0) equal to (r,0). We can then use the following equations to generate successive points:

Xn+1=Xncos(u)-Ynsin(u)

Yn+1 = Xnsin(u) + Yncos(u)

ELLIPSE GENERATING ALGORITHM



- 2a is the length of the major axis along the x axis.
- 2b is the length of the minor axis along the y axis.

- The midpoint can also be applied to ellipses.
- For simplicity, we draw only the arc of the ellipse that lies in the first quadrant, the other three quadrants can be drawn by symmetry
- Firstly we divide the quadrant into two regions
- Boundary between the two regions is
  - the point at which the curve has a slope of -1
  - the point at which the gradient vector has the i and j components of equal magnitude
- At the next midpoint, if a2(yp-0.5)<=b2(xp+1), we switch region 1=>2
- In region 1, choices are E and SE
  - Initial condition: dinit = b2+a2(-b+0.25)
  - For a move to E, dnew = dold+DeltaE with DeltaE = b2(2xp+3)
  - For a move to SE, dnew = dold+DeltaSE with DeltaSE = b2(2xp+3)+a2(-2yp+2)
- In region 2, choices are S and SE
  - Initial condition: dinit = b2(xp+0.5)2+a2((y-1)2-b2)
  - For a move to S, dnew = dold+Deltas with Deltas = a2(-2yp+3)
  - For a move to SE, dnew = dold+DeltaSE with DeltaSE = b2(2xp+2)+a2(-2yp+3)
- Stop in region 2 when the y value is zero.

```
d2 = b²(x + 0.5)² + a²(y − 1)² − a²b²;
while (y > 0) {                          /* Region 2 */
    if (d2 < 0) {                        /* Select SE */
        d2 += b²(2x + 2) + a²(−2y + 3);
        x++;
    } else
        d2 += a²(−2y + 3);               /* Select S */
    y−−;
    EllipsePoints (x, y, value);
} /* Region 2 */
} /* MidpointEllipse */
```

AREA FILLED ATTRIBUTES

It is an area filled with the foreground color, but it depends on the current interior style. The SOLID style depends only on the foreground color. The HATCH and STIPPLE style depend on the foreground color, background color and on the back opacity attribute. The hatch lines drawn with this style do not depend on the other line attributes. The PATTERN style depends only on global canvas attributes.

The filled area includes the line at the edge of the area. So if you draw a filled rectangle, sector or polygon on top of a non filled one using the same coordinates, no style and 1 pixel width, the non filled primitive should be obscured by the filled primitive. But depending on the driver implementation some pixels at the edges may be not included. IMPORTANT: In the Postscript and PDF drivers the line at the edge is not included at all.

If either the background or the foreground color are modified, the hatched and monochromatic fillings must be modified again in order to be updated.

Note that when a Filling Attribute is modified, the active filling style is now that of the modified attribute (hatch, stipple or pattern). Notice that this is not true for the clipping area. When the clipping area is modified, the clipping is only affected if it is active.

Fills the arc of an ellipse aligned with the axis, according to the current interior style, in the shape of a pie. It is drawn counter-clockwise. The coordinate (xc,yc) defines the center of the ellipse. Dimensions w and h define the elliptic axes X and Y, respectively.

Angles angle1 and angle2, in degrees, define the arc's beginning and end, but they are not the angle relative to the center, except when w==h and the ellipse is reduced to a circle. The arc starts at the point (xc+(w/2)*cos(angle1),yc+(h/2)*sin(angle1)) and c+(w/2)*cos(angle2),yc+(h/2)*sin(angle2)).
A complete ellipse can be drawn using 0 and 360 as the angles.

The angles are specified so if the size of the ellipse (w x h) is changed, its shape is preserved. So the angles relative to the center are dependent from the ellipse size. The actual angle can be obtained using `rangle = atan2((h/2)`*sin(angle),(w/2)*cos(angle)).

The angles are given in degrees. To specify the angle in radians, you can use the definition CD_RAD2DEG to multiply the value in radians before passing the angle to CD. When the interior style CD_HOLLOW is defined, the function behaves like its equivalent cd Arc, plus two lines connecting to the center.

LINE ATTRIBUTES

The graph drawing routines may be freely mixed with those described in this section, allowing the user to control line color, width and styles. The attributes set up by these routines apply modally, i.e, all subsequent objects (lines, characters and symbols) plotted until the next change in attributes are affected in the same way. The only exception to this rule is that characters and symbols are not affected by a change in the line style, but are always drawn using a continuous line.

Line color is set using the routine `plcol0`. The argument is ignored for devices which can only plot in one color, although some terminals support line erasure by plotting in color zero.

Line width is set using `plwid`. This option is not supported by all devices.

Line style is set using the routine `plstyl` or `pllsty`. A broken line is specified in terms of a repeated pattern consisting of marks (pen down) and spaces (pen up). The arguments to this routine are the number of elements in the line, followed by two pointers to integer arrays specifying the mark and space lengths in micrometers. Thus a line consisting of long and short dashes of lengths 4 mm and 2 mm, separated by spaces of length 1.5 mm is specified by:

```
mark[0] = 4000;
mark[1] = 2000;
space[0] = 1500;
space[1] = 1500;
plstyl(2, mark, space);
```

To return to a continuous line, just call `plstyl` with first argument set to zero. You can use `pllsty` to choose between 8 different predefined styles.

Lines and line segments

In mathematics, lines are infinite – that is, they have no ends. It is not often, in the real world, that we deal with such lines. When people use the word 'line' to talk about the lines of a tennis court, for example, what they actually mean is 'line segment. A line segment is the set of points on the straight line between any two points, including the two endpoints themselves.

A section of a line between and including any two points on that line is a line segmentThe region of a circle bounded by an arc and the chord joining its two end points.

PONDICHERRY UNIVERSITY QUESTIONS

GRAPHICS AND IMAGE PROCESSING

1. How to represent the matrix in Two-Dimensional Transformation? Discuss it. **(APRIL / MAY 2012)**
2. Write and algorithm in Line Clipping. **(APRIL / MAY 2012)**
3. Explain DDA line drawing algorithm in detail. **(NOVEMBER 2012)**
4. Illustrate the procedure for applying Reflect transformations. **(NOVEMBER 2012)**
5. Illustrate line drawing algorithm **(APRIL 2013)**
6. Briefly explain Polygon Clipping. **(APRIL 2013)**
7. Explain DDA Line Drawing Algorithm in detail with and example. Also list down the disadvantages of DDA algorithm over Bresenham's algorithm. **(NOVEMBER 2013)**
8. Describe the general procedure for applying translation, rotation and scaling parameters to reposition and resize two-dimensional object. **(APRIL 2014)**
9. Explain Cohen Sutherland Line Clipping algorithm in detail with a suitable example. **(NOVEMBER 2013) (APRIL 2014)**

UNIT III

Digital Image Fundamentals and Transforms: Nature of Image processing – related fields – Image representations – Image types – Image processing operations – Applications of Image processing – Imaging system – Image Acquisition – Image Sampling and Quantization – Image quality – Image storage and file formats - Image processing operations - Image Transforms - need for Transforms – Fourier Transforms and its properties – Introduction to Walsh, Hadamard, Discrete Cosine, Haar, Slant, SVD, KL and Hotelling Transforms.

IMAGE PROCESSING

An image may be defined as a two-dimensional function, f(x, y), where x and y are *spatial* (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the *intensity* or *gray level* of the image at that point. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a *digital image*. The field of *digital image processing* refers to processing digital images by means of a digital computer. A digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are referred to as *picture elements*, *image elements*, *pels*, and *pixels*. *Pixel* is the term most widely used to denote the elements of a digital image.

DIGITAL IMAGE REPRESENTATION

"*Virtual image, a point or system of points, on one side of a mirror or lens, which, if it existed, would emit the system of rays which actually exists on the other side of the mirror or lens.*"

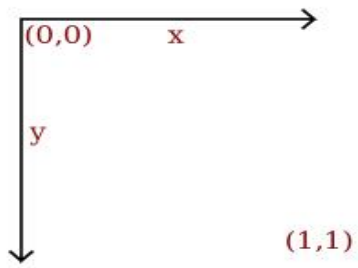
Vector images

One way to describe an image using numbers is to declare its contents using position and size of geometric forms and shapes like lines, curves, rectangles and circles; such images are called vector images.

Co-ordinate system

We need a coordinate system to describe an image, the coordinate system used to place elements in relation to each other is called ***user space***, since this is the coordinates the user uses to define elements and position them in relation to each other.

Coordinate system.

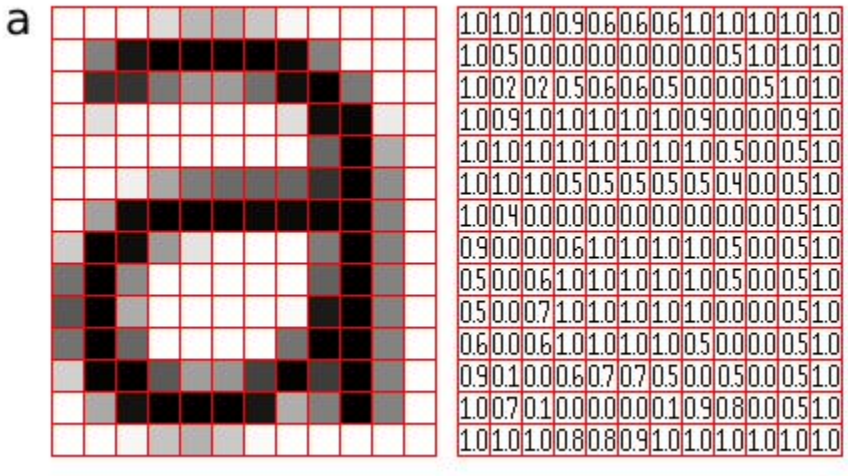


Bitmap images

Bitmap, or raster, images are "digital photographs", they are the most common form to represent natural images and other forms of graphics that are rich in detail. Bitmap images is how graphics is stored in the video memory of a computer. The term bitmap refers to how a given pattern of bits in a pixel maps to a specific color.

A bitmap images take the form of an array, where the value of each element, called a ***pixel*** picture element, correspond to the color of that portion of the image. Each horizontal line in the image is called a ***scan line***.

Raster image



A rasterized form of the letter 'a' magnified 16 times using pixel.

The letter 'a' might be represented in a 12x14 matrix as depicted in Figure the values in the matrix depict the brightness of the ***pixels*** (picture elements). Larger values correspond to brighter areas whilst lower values are darker.

Raster dimensions

The number of horizontal and vertical samples in the pixel grid is called ***Raster dimensions***, it is specified as width x height.

Resolution

Resolution is a measurement of sampling density, resolution of bitmap images gives a relationship between pixel dimensions and physical dimensions. The most often used measurement is ppi, pixels per inch

Megapixels

Megapixels refer to the total number of pixels in the captured image, an easier metric is ***raster dimensions*** which represent the number of horizontal and vertical samples in the sampling grid. An image with a 4:3 aspect ratio with dimension 2048x1536 pixels, contain a total of 2048x1535=3,145,728 pixels; approximately 3 million, thus it is a 3 megapixel image.

Scaling / Resampling

When we need to create an image with different dimensions from what we have we scale the image. A different name for scaling is resampling, when resampling algorithms try to reconstruct the original continuous image and create a new sample grid.

Scaling image down

The process of reducing the raster dimensions is called ***decimation***; this can be done by averaging the values of source pixels contributing to each output pixel.

Scaling image up

When we increase the image size we actually want to create sample points between the original sample points in the original raster, this is done by ***interpolation*** the values in the sample grid, effectively guessing the values of the unknown pixels.

Colors

The most common way to model color in Computer Graphics is the RGB color model, this corresponds to the way both CRT monitors and LCD screens/projectors reproduce color. Each pixel is represented by three values, the amount of red, green and blue. Thus an RGB color image will use three times as much memory as a gray-scle image of the same pixel dimensions.

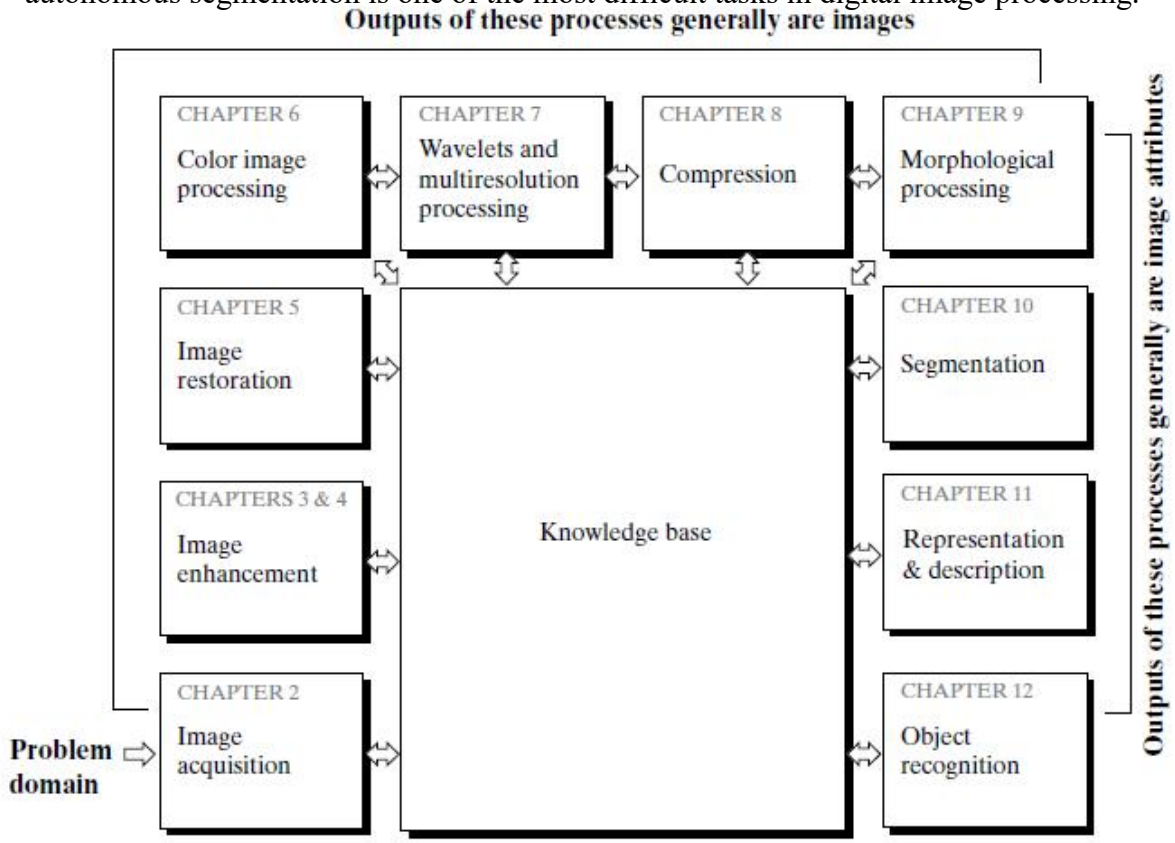
Palette/Indexed image

It was earlier common to store images in a palletized mode, this works similar to a paint by numbers strategy. We store just the number of the palette entry used for each pixel. And for each palette entry we store the amount of red, green and blue light.

FUNDAMENTAL STEPS IN DIGITAL IMAGE PROCESSING

- *Image acquisition* is the first process that acquisition could be as simple as being given an image that is already in digital form. Generally, the image acquisition stage involves preprocessing, such as scaling.
- *Image enhancement* is among the simplest and most appealing areas of digital image processing. Basically, the idea behind enhancement techniques is to bring out detail that is obscured, or simply to highlight certain features of interest in an image.
- *Image restoration* is an area that also deals with improving the appearance of an image. However, unlike enhancement, which is subjective, image restoration is objective, in the sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation.
- *Color image processing* is an area that has been gaining in importance because of the significant increase in the use of digital images over the Internet.
- *Wavelets* are the foundation for representing images in various degrees of resolution.
- *Compression*, as the name implies, deals with techniques for reducing the storage required to save an image, or the bandwidth required to transmit it. Although storage technology has improved significantly over the past decade, the same cannot be said for transmission capacity.

- *Morphological processing* deals with tools for extracting image components that are useful in the representation and description of shape.
- *Segmentation* procedures partition an image into its constituent parts or objects. In general, autonomous segmentation is one of the most difficult tasks in digital image processing.



Representation and description almost always follow the output of segmentation stage, which usually is raw pixel data, constituting either the boundary of a region (i.e., the set of pixels separating one image region from another) or all the points in the region itself.
Recognition is the process that assigns a label (e.g., "vehicle") to an object based on its descriptors.

Knowledge about a problem domain is coded into an image processing system in the form of a knowledge database. This knowledge may be as simple as detailing regions of an image where the information of interest is known to be located, thus limiting the search that has to be conducted in seeking that information. The knowledge base also can be quite complex, such as an interrelated list of all major possible defects in a materials inspection problem or an image database containing high-resolution satellite images of a region in connection with change-detection applications. In addition to guiding the operation of each processing module, the knowledge base also controls the interaction between modules.

COMPONENTS OF AN IMAGE PROCESSING SYSTEM

With reference to *sensing*, two elements are required to acquire digital images. The first is a physical device that is sensitive to the energy radiated by the object we wish to image.

The second, called a *digitizer*, is a device for converting the output of the physical sensing device into digital form. For instance, in a digital video camera, the sensors produce an electrical output proportional to light intensity. The digitizer converts these outputs to digital data.

Specialized image processing hardware usually consists of the digitizer just mentioned, plus hardware that performs other primitive operations, such as an arithmetic logic unit (ALU), which performs arithmetic and logical operations in parallel on entire images. One example of how an ALU is used is in averaging images as quickly as they are digitized, for the purpose of noise reduction.
This type of hardware sometimes is called a *front-end subsystem*, and its most distinguishing characteristic is speed. In other words, this unit performs functions that require fast data throughputs (e.g., digitizing and averaging video images at 30 frames/s) that the typical main computer cannot handle.

The *computer* in an image processing system is a general-purpose computer and can range from a PC to a supercomputer. In dedicated applications, sometimes specially designed computers are used to achieve a required level of performance, but our interest here is on general-purpose image processing systems.

In these systems, almost any well-equipped PC-type machine is suitable for offline image processing tasks.

Software for image processing consists of specialized modules that perform specific tasks. A well-designed package also includes the capability for the user to write code that, as a minimum, utilizes the specialized modules. More sophisticated software packages allow the integration of those modules and general- purpose software commands from at least one computer language.

Mass storage capability is a must in image processing applications. An image of size 1024*1024 pixels, in which the intensity of each pixel is an 8-bit quantity, requires one megabyte of storage space if the image is not compressed. When dealing with thousands, or even millions, of images, providing adequate storage in an image processing system can be a challenge.

Digital storage for image processing applications falls into three principal categories:

(1) short term storage for use during processing, (2) on-line storage for relatively fast recall,

and (3) archival storage, characterized by infrequent access.

Storage is measured in bytes (eight bits), Kbytes (one thousand bytes), Mbytes (one million bytes), Gbytes (meaning giga, or one billion, bytes), and Tbytes (meaning tera, or one trillion, bytes). One method of providing short-term storage is computer memory. Another is by specialized boards, called *frame buffers*, that store one or more images and can be accessed rapidly, usually at video rates (e.g., at 30 complete images per second).The latter method allows virtually instantaneous image *zoom*, as well as *scroll* (vertical shifts) and *pan* (horizontal shifts).

Image displays in use today are mainly color (preferably flat screen) TV monitors. Monitors are driven by the outputs of image and graphics display cards that are an integral part of the computer system.

Hardcopy devices for recording images include laser printers, film cameras, heat-sensitive devices, inkjet units, and digital units, such as optical and CD-ROM disks.

Networking is almost a default function in any computer system in use today. Because of the large amount of data inherent in image processing applications, the key consideration in image transmission is bandwidth.

FIELDS THAT USE DIGITAL IMAGE PROCESSING

One of the simplest ways to develop a basic understanding of the extent of image processing applications is to categorize images according to their source (e.g., visual, X-ray, and so on).The

principal energy source for images in use today is the electromagnetic energy spectrum. Other important sources of energy include acoustic, ultrasonic, and electronic (in the form of electron beams used in electron microscopy).

Synthetic images, used for modeling and visualization, are generated by computer. Images based on radiation from the EM spectrum are the most familiar, especially images in the X-ray and visual bands of the spectrum. Electromagnetic waves can be conceptualized as propagating sinusoidal waves of varying wavelengths, or they can be thought of as a stream of massless particles, each traveling in a wavelike pattern and moving at the speed of light.

Major uses of imaging based on gamma rays include nuclear medicine and astronomical observations. In nuclear medicine, the approach is to inject a patient with a radioactive isotope that emits gamma rays as it decays. Images are produced from the emissions collected by gamma ray detectors.

X-rays are among the oldest sources of EM radiation used for imaging. The best known use of X-rays is medical diagnostics, but they also are used extensively in industry and other areas, like astronomy. X-rays for medical and industrial imaging are generated using an X-ray tube, which is a vacuum tube with a cathode and anode.

Angiography is another major application in an area called contrast enhancement radiography. This procedure is used to obtain images (called *angiograms*) of blood vessels. A catheter (a small, flexible, hollow tube) is inserted, for example, into an artery or vein in the groin. The catheter is threaded into the blood vessel and guided to the area to be studied. When the catheter reaches the site under investigation, an X-ray contrast medium is injected through the catheter. This enhances contrast of the blood vessels and enables the radiologist to see any irregularities or blockages.

Applications of ultraviolet “light” are varied. They include lithography, industrial inspection, microscopy, lasers, biological imaging, and astronomical observations.

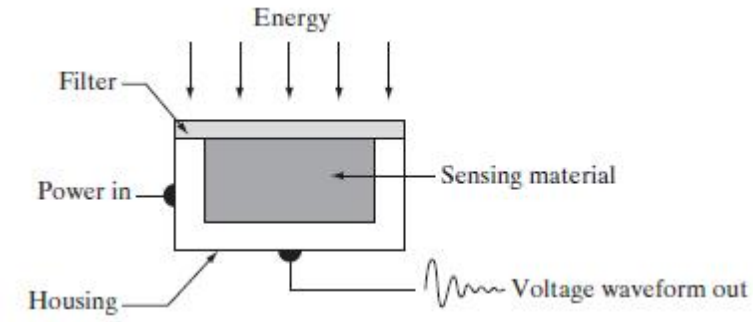
Another major area of visual processing is remote sensing, which usually includes several bands in the visual and infrared regions of the spectrum.

On the other hand, there are fields such as computer vision whose ultimate goal is to use computers to emulate human vision, including learning and being able to make inferences and take actions based on visual inputs. This area itself is a branch of artificial intelligence (AI) whose objective is to emulate human intelligence. The field of AI is in its earliest stages of infancy in terms of development, with progress having been much slower than originally anticipated. The area of

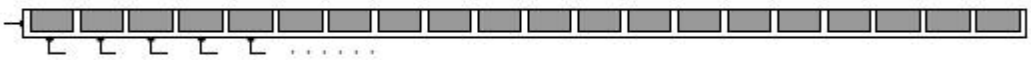
image analysis (also called image understanding) is in between image processing and computer vision.

IMAGE SENSING AND ACQUISITION

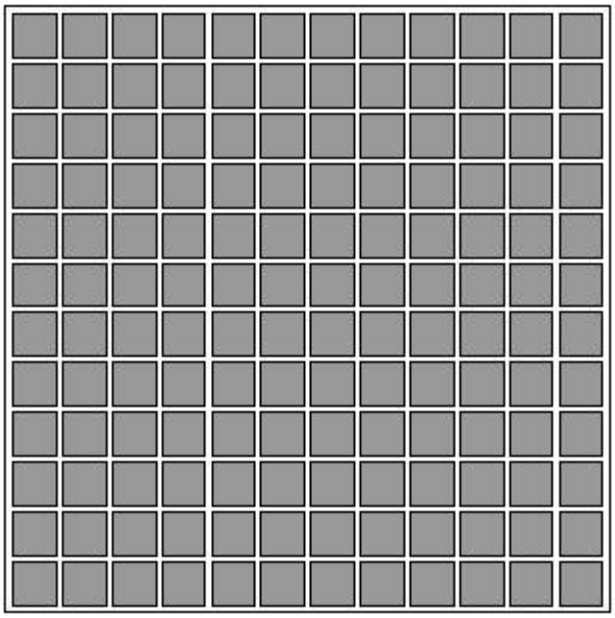
The types of images are generated by the combination of an "illumination" source and the reflection or absorption of energy from that source by the elements of the "scene" being imaged. For example, the illumination may originate from a source of electromagnetic energy such as radar, infrared, or X-ray energy. But, as noted earlier, it could originate from less traditional sources, such as ultrasound or even a computer-generated illumination pattern. Similarly, the scene elements could be familiar objects, but they can just as easily be molecules, buried rock formations, or a human brain. Depending on the nature of the source, illumination energy is reflected from, or transmitted through, objects.



Single imaging sensor



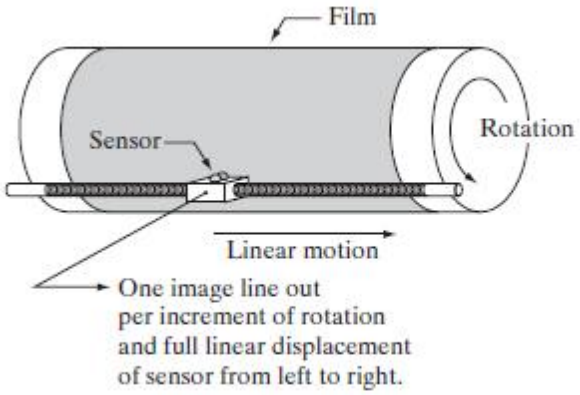
Line sensor



Array sensor

Image Acquisition Using a Single Sensor

In order to generate a 2-D image using a single sensor, there has to be relative displacements in both the x- and y-directions between the sensor and the area to be imaged. Figure shows an arrangement used in high-precision scanning, where a film negative is mounted onto a drum whose mechanical rotation provides displacement in one dimension. The single sensor is mounted on a lead screw that provides motion in the perpendicular direction. Since mechanical motion can be controlled with high precision, this method is an inexpensive (but slow) way to obtain high-resolution images. Other similar mechanical arrangements use a flat bed, with the sensor moving in two linear directions. These types of mechanical digitizers sometimes are referred to as *micro densitometers*. Another example of imaging with a single sensor places a laser source coincident with the sensor. Moving mirrors are used to control the outgoing beam in a scanning pattern and to direct the reflected laser signal onto the sensor. This arrangement also can be used to acquire images using strip and array sensors.



Combining a single sensor with motion to generate a 2-D image.

Image Acquisition Using Sensor Strips

A geometry that is used much more frequently than single sensors consists of an in-line arrangement of sensors in the form of a sensor strip. The strip provides imaging elements in one direction. Motion perpendicular to the strip provides imaging in the other direction. This is the type of arrangement used in most flat bed scanners. Sensing devices with 4000 or more in-line sensors are possible. In-line sensors are used routinely in airborne imaging applications, in which the imaging system is mounted on an aircraft that flies at a constant altitude and speed over the geographical area to be imaged. One-dimensional imaging sensor strips that respond to various bands of the electromagnetic spectrum are mounted perpendicular to the direction of flight. The imaging strip gives one line of an image at a time, and the motion of the strip completes the other dimension of a two-dimensional image. Lenses or other focusing schemes are used to project the area to be scanned onto the sensors. Sensor strips mounted in a ring configuration are used in medical and industrial imaging to obtain cross-sectional ("slice") images of 3-D objects. A rotating X-ray source provides illumination and the portion of the sensors opposite the source collect the X-ray energy that pass through the object (the sensors obviously have to be sensitive to X-ray energy).This is the basis for medical and industrial computerized axial tomography (CAT). A 3-D digital volume consisting of stacked images is generated as the object is moved in a direction perpendicular to the sensor ring. Other modalities of imaging based on the CAT principle include magnetic resonance imaging (MRI) and positron emission tomography (PET).

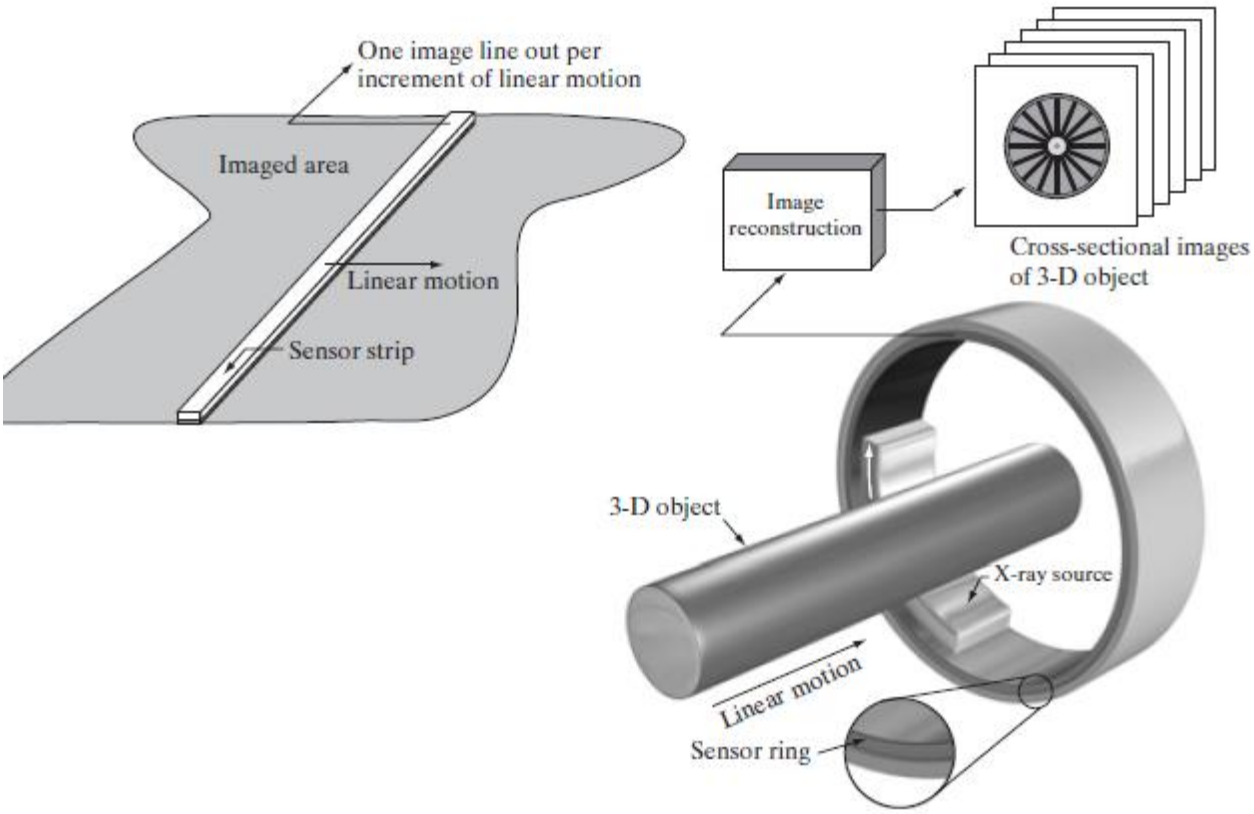
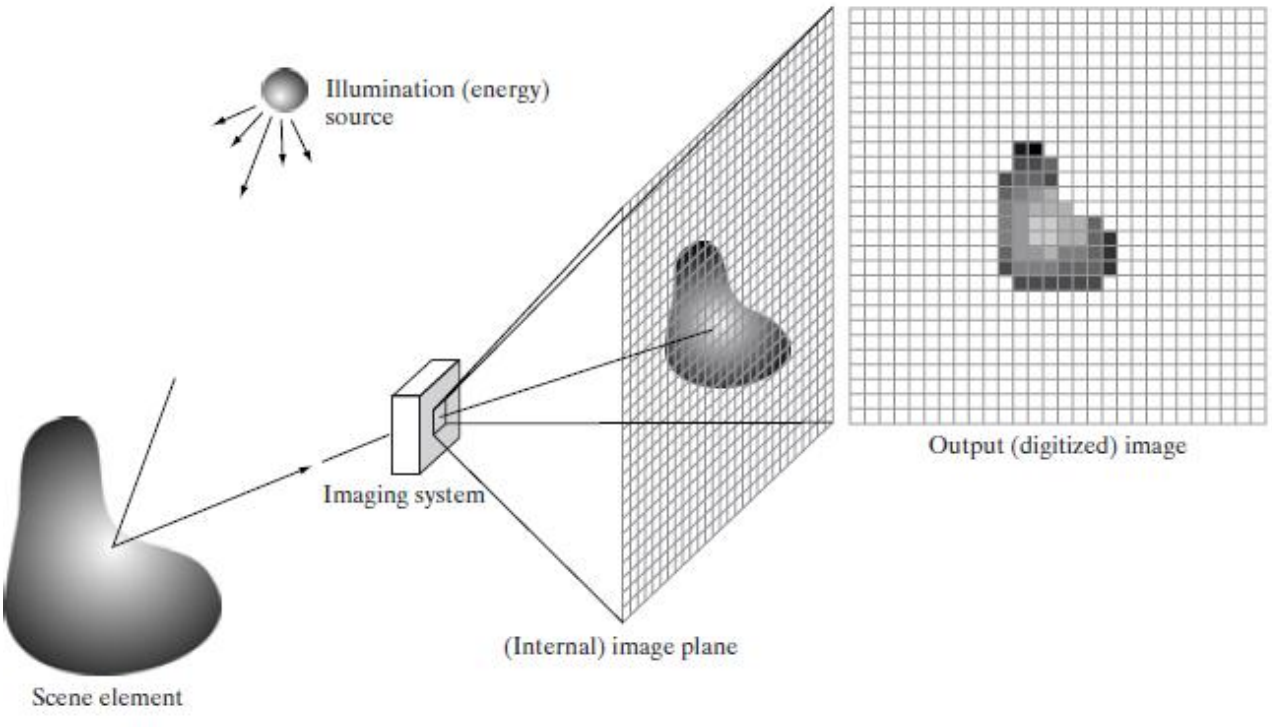


Image acquisition using linear & circular sensor strip

Image Acquisition Using Sensor Arrays

Numerous electromagnetic and some ultrasonic sensing devices frequently are arranged in an array format. This is also the predominant arrangement found in digital cameras. A typical sensor for these cameras is a CCD array, which can

be manufactured with a broad range of sensing properties and can be packaged in rugged arrays of 4000 x 4000 elements or more. CCD sensors are used widely in digital cameras and other light sensing instruments. The response of each sensor is proportional to the integral of the light energy projected onto the surface of the sensor, a property that is used in astronomical and other applications requiring low noise images. Noise reduction is achieved by letting the sensor integrate the input light signal over minutes or even hours. The first function performed by the imaging system is to collect the incoming energy and focus it onto an image plane. If the illumination is light, the front end of the imaging system is a lens, which projects the viewed scene onto the lens focal plane. The sensor array, which is coincident with the focal plane, produces outputs proportional to the integral of the light received at each sensor. Digital and analog circuitry sweeps these outputs and converts them to a video signal, which is then digitized by another section of the imaging system.



An example of digital image acquisition process

SAMPLING

Both sounds and images can be considered as signals, in one or two dimensions, respectively. Sound can be described as a fluctuation of the acoustic pressure in time, while images are spatial distributions of values of luminance or color, the latter being described in its RGB or HSB components. Any signal, in order to be processed by numerical computing devices, have to be reduced to a sequence of discrete *samples*, and each sample must be represented using a finite number of bits. The first operation is called *sampling*, and the second operation is called *quantization* of the domain of real numbers.

1-D: Sounds

Sampling is, for one-dimensional signals, the operation that transforms a continuous-time signal (such as, for instance, the air pressure fluctuation at the entrance of the ear canal) into a discrete-time signal, that is a sequence of numbers. The discrete-time signal gives the values of the continuous-time signal read at intervals of T seconds. The reciprocal of the sampling interval is called *sampling rate* Fs= 1/T

Fact 1: The Fourier Transform of a discrete-time signal is a function (called *spectrum*) of the continuous variable ω, and it is periodic with period 2π. Given a value of ω, the Fourier transform gives back a complex number that can be interpreted as magnitude and phase (translation in time) of the sinusoidal component at that frequency.

Fact 2: Sampling the continuous-time signal $x(t)$ with interval T we get the discrete-time signal $x(n) = x(nT)$, which is a function of the discrete variable n.

Fact 3: Sampling a continuous-time signal with sampling rate Fs produces a discrete-time signal whose frequency spectrum is the periodic replication of the original signal, and the replication period is Fs. The Fourier variable ω for functions of discrete variable is converted into the frequency variable f (in Hertz) by means of

$$f = \boxed{\omega/2\pi T}$$

The Figure 1 shows an example of frequency spectrum of a signal sampled with sampling rate *F*s. In the example, the continuous-time signal had all and only the frequency components between −*F*b and *F*b. The replicas of the original spectrum are sometimes called *images*.

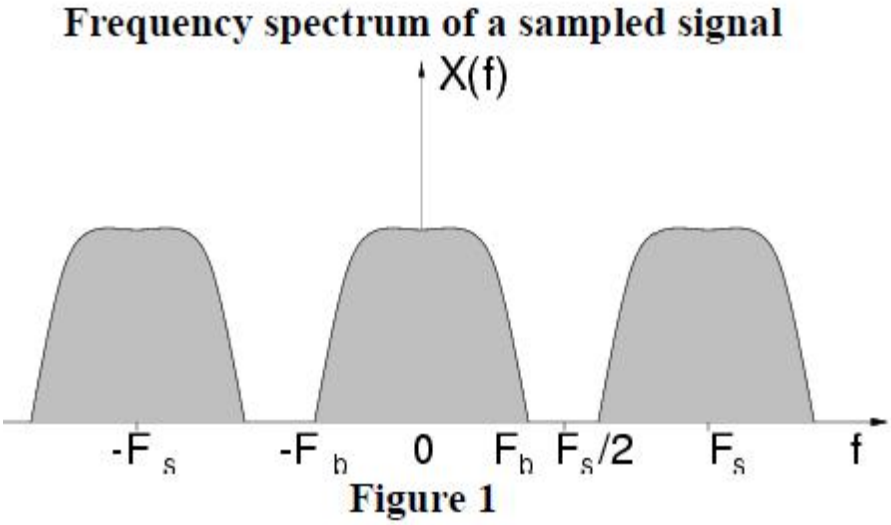


Figure 1

2-D: Images

Let us assume we have a continuous distribution, on a plane, of values of luminance or, more simply stated, an image. In order to process it using a computer we have to reduce it to a sequence of numbers by means of sampling. There are several ways to sample an image, or read its values of luminance at discrete points. The simplest way is to use a regular grid, with spatial steps *X* e *Y*. Similarly to what we did for sounds, we define the spatial sampling rates *F*X= 1/X

FY= 1/Y

As in the one-dimensional case, also for two-dimensional signals, or images, sampling can be described by three facts and a theorem.

Fact 1: The Fourier Transform of a discrete-space signal is a function (called *spectrum*) of two continuous variables ωX and ωY, and it is periodic in two dimensions with periods 2π. Given a couple of values ωX and ωY, the Fourier transform gives back a complex number that can be interpreted as magnitude and phase (translation in space) of the sinusoidal component at such spatial frequencies.

Fact 2: Sampling the continuous-space signal $s(x,y)$ with the regular grid of steps X, Y, gives a discrete-space signal $s(m,n) = s(mX,nY)$, which is a function of the discrete variables m and n.

Fact 3: Sampling a continuous-space signal with spatial frequencies FX and FY gives a discrete-space signal whose spectrum is the periodic replication along the grid of steps FX and FY of the original signal spectrum. The Fourier variables ωX and ωY correspond to the frequencies (in cycles per meter) represented by the variables $fX = \omega X/2\pi X$

The Figure 2 shows an example of spectrum of a two-dimensional sampled signal. There, the continuous-space signal had all and only the frequency components included in the central hexagon. The hexagonal shape of the spectral support (region of non-null spectral energy) is merely illustrative. The replicas of the original spectrum are often called spectral *images*.

Spectrum of a sampled image

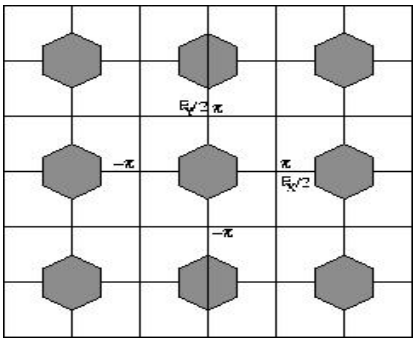


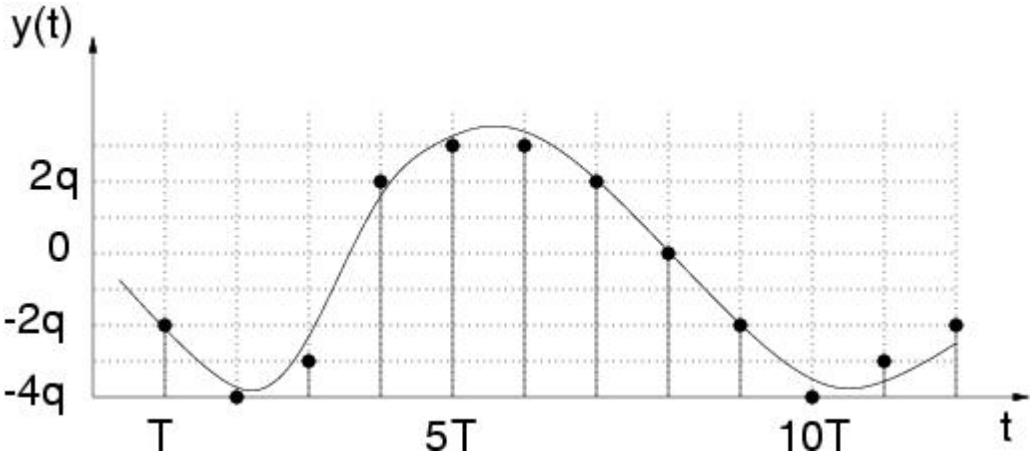
Figure 2

QUANTIZATION

With the adjective "digital" we indicate those systems that work on signals that are represented by numbers, with the (finite) precision that computing systems allow. Up to now we have considered discrete-time and discrete-space signals as if they were collections of infinite-precision numbers, or real numbers. Unfortunately, computers only allow to represent finite subsets of rational numbers. This means that our signals are subject to quantization.

For our purposes, the most interesting quantization is the linear one, which is usually occurring in the process of conversion of an analog signal into the digital domain. If the memory word dedicated to storing a number is made of b bits, then the range of such number is discretized into $2b$ quantization levels. Any value that is found between two quantization levels can be approximated by truncation or rounding to the closest value. The Figure 3 shows an example of quantization with representation on 3 bits in **two's complement.**

Sampling and quantization of an analog signal



The approximation introduced by quantization manifests itself as a noise, called *quantization noise*. Often, for the analysis of sound-processing circuits, such noise is assumed to be white and de-correlated with the signal, but in reality it is perceptually tied to the signal itself, in such an extent that quantization can be perceived as an effect.

To have a visual and intuitive exploration of the phenomenon of quantization, consider the applet that allows to vary between 1 and 8 the number of bits dedicated to the representation of each of the RGB channels representing color. The same number of bits is dedicated to the representation of an audio signal coupled to the image. The visual effect that is obtained by reducing the number of bits is similar to a *solarization*.

BASIC RELATIONSHIP BETWEEN PIXELS (APR 2014)

PIXEL

In digital imaging, a **pixel** (or **picture element**) is a single point in a raster image. The pixel is the smallest addressable screen element; it is the smallest unit of picture that can be controlled. Each pixel has its own address. The address of a pixel corresponds to its coordinates. Pixels are normally arranged in a 2-dimensional grid, and are often represented using dots or squares. Each pixel is a sample of an original image; more samples typically provide more accurate representations of the original. The intensity of each pixel is variable. In color image systems, a color is typically represented by three or four component intensities such as red, green, and blue, or cyan, magenta, yellow, and black.

The word *pixel* is based on a contraction of *pix* ("pictures") and *el* (for "element"); similar formations with *el* for "element" include the words: voxel and texel.

Bits per pixel

The number of distinct colors that can be represented by a pixel depends on the number of bits per pixel (bpp). A 1 bpp image uses 1-bit for each pixel, so each pixel can be either on or off. Each additional bit doubles the number of colors available, so a 2 bpp image can have 4 colors, and a 3 bpp image can have 8 colors:

- 1 bpp, 21 = 2 colors (monochrome)
- 2 bpp, 22 = 4 colors
- 3 bpp, 23 = 8 colors
- 8 bpp, 28 = 256 colors
- 16 bpp, 216 = 65,536 colors ("Highcolor")
- 24 bpp, 224 ≈ 16.8 million colors ("Truecolor")

For color depths of 15 or more bits per pixel, the depth is normally the sum of the bits allocated to each of the red, green, and blue components. Highcolor, usually meaning 16 bpp, normally has five bits for red and blue, and six bits for green, as the human eye is more sensitive to errors in green than in the other two primary colors. For applications involving transparency, the 16 bits may be divided into five bits each of red, green, and available: this means that each 24-bit pixel has an extra 8 bits to describe its blue, with one bit left for transparency. A 24-bit depth allows 8 bits per component. On some systems, 32-bit depth is opacity (for purposes of combining with another image).

Selected standard display resolutions include:

Name	Megapixels	Width x Height
CGA	0.064	320×200
EGA	0.224	640×350
VGA	0.3	640×480
SVGA	0.5	800×600
XGA	0.8	1024×768
SXGA	1.3	1280×1024
UXGA	1.9	1600×1200
WUXGA	2.3	1920×1200

Neighbors of a Pixel

A pixel p at coordinates (x, y) has four *horizontal* and *vertical* neighbors whose coordinates are given by

(x+1, y), (x-1, y), (x, y+1), (x, y-1)

This set of pixels, called the 4-*neighbors* of p, is denoted by $N4(p)$. Each pixel is a unit distance from (x, y), and some of the neighbors of p lie outside the digital image if (x, y) is on the border of the image. The four *diagonal* neighbors of p have coordinates

(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)

and are denoted by $ND(p)$. These points, together with the 4-neighbors, are called the 8-*neighbors* of p, denoted by $N8(p)$. As before, some of the points in $ND(p)$ and $N8(p)$ fall outside the image if (x, y) is on the border of the image.

Adjacency, Connectivity, Regions, and Boundaries

Connectivity between pixels is a fundamental concept that simplifies the definition of numerous digital image concepts, such as regions and boundaries. To establish if two pixels are connected, it must be determined if they are neighbors and if their gray levels satisfy a specified criterion of similarity (say, if their gray levels are equal).For instance, in a binary image with values 0 and 1, two pixels may be 4-neighbors, but they are said to be connected only if they have the same value.

Let V be the set of gray-level values used to define adjacency. In a binary image,$V=\{1\}$ if we are referring to adjacency of pixels with value 1. In a grayscale image, the idea is the same, but set V typically contains more elements. For example, in the adjacency of pixels with a range of possible gray-level values 0 to 255, set V could be any subset of these 256 values.We consider three types of adjacency:

(a) 4-*adjacency*. Two pixels p and q with values from V are 4-adjacent if q is in the set $N4(p)$.

(b) 8-*adjacency*. Two pixels p and q with values from V are 8-adjacent if q is in the set $N8(p)$.

(c) *m-adjacency* (mixed adjacency).Two pixels p and q with values from V are m-adjacent if

(i) q is in $N4(p)$, *or*

(ii) q is in $ND(p)$ *and* the set $N4(p) \cap N4(q)$ has no pixels whose values are from V.

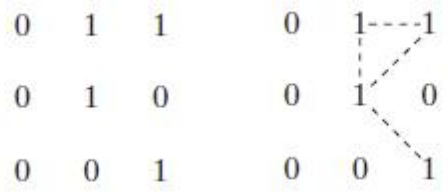
Mixed adjacency is a modification of 8-adjacency. It is introduced to eliminate the ambiguities that often arise when 8-adjacency is used.

A (*digital*) *path* (or *curve*) from pixel p with coordinates (x, y) to pixel q with coordinates (s, t) is a sequence of distinct pixels with coordinates (x0 , y0),(Ax1 , y1), .. , (xn , yn)

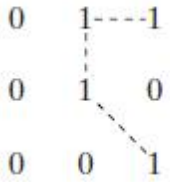
Let S represent a subset of pixels in an image.Two pixels p and q are said to be *connected* in S if there exists a path between them consisting entirely of pixels in S. For any pixel p in S, the *set* of pixels that are connected to it in S is called a *connected component* of S. If it only has one connected component,

then set *S* is called a *connected set*.

Let *R* be a subset of pixels in an image.We call *R* a *region* of the image if *R* is a connected set.The *boundary* (also called *border* or *contour*) of a region *R* is the set of pixels in the region that have one or more neighbors that are not in *R*. If *R* happens to be an entire image (which we recall is a rectangular set of pixels), then its boundary is defined as the set of pixels in the first and last rows and columns of the image.



Arrangement of pixels Pixels are 8-adjacent(shown dashed) to center line



m-adjacency

BASIC GEOMETRIC TRANSFORMATIONS

Transform theory plays a fundamental role in image processing, as working with the transform of an image instead of the image itself may give us more insight into the properties of the image. Two dimensional transforms are applied to image enhancement, restoration, encoding and description.

UNITARY TRANSFORMS

One dimensional signals

For a one dimensional sequence $\{f(x), 0 \le x \le N-1\}$ represented as a vector $\underline{f} = [f(0)\, f(1) \ldots f(N-1)]^T$ of size N, a transformation may be written as

$$\underline{g} = \underline{T} \cdot \underline{f} \Rightarrow g(u) = \sum_{x=0}^{N-1} T(u,x) f(x), \; 0 \le u \le N-1$$

where $g(u)$ is the transform (or transformation) of $f(x)$, and $T(u,x)$ is the so called **forward transformation kernel**. Similarly, the inverse transform is the relation

$$f(x) = \sum_{u=0}^{N-1} I(x,u) g(u), \; 0 \le x \le N-1$$

or written in a matrix form

$$\underline{f} = \underline{I} \cdot \underline{g} = \underline{T}^{-1} \cdot \underline{g}$$

where $I(x,u)$ is the so called **inverse transformation kernel**.

If

$$\underline{I} = \underline{T}^{-1} = \underline{T}^{*T}$$

the matrix T is called **unitary**, and the transformation is called unitary as well. It can be proven that the columns (or rows) of an N x N unitary matrix are orthonormal and therefore, form a complete set of **basis vectors** in the N-dimensional vector space. In that case

$$\underline{f} = \underline{T}^{*T} \cdot \underline{g} \Rightarrow f(x) = \sum_{u=0}^{N-1} T^*(u,x) g(u)$$

The columns of $\underline{T}^{*T}$, that is, the vectors $\underline{T}_u^* = [T^*(u,0)\, T^*(u,1) \ldots T^*(u,N-1)]^T$ are called the **basis vectors** of $\underline{T}$.

Two dimensional signals (images)

As a one dimensional signal can be represented by an orthonormal set of **basis vectors**, an image can also be expanded in terms of a discrete set of **basis arrays** called basis images through a **two dimensional (image) transform**. For an N x N image f(x,y) the forward and inverse transforms are given below

$$g(u,v)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}T(u,v,x,y)f(x,y)$$

$$f(x,y)=\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}I(x,y,u,v)g(u,v)$$

where, again, $T(u,v,x,y)$ and $I(x,y,u,v)$ are called the **forward and inverse transformation kernels**, respectively.

The forward kernel is said to be **separable** if

$$T(u,v,x,y)=T_1(u,x)T_2(v,y)$$

It is said to be **symmetric** if T_1 is functionally equal to T_2 such that

$$T(u,v,x,y)=T_1(u,x)T_1(v,y)$$

The same comments are valid for the inverse kernel.

If the kernel $T(u,v,x,y)$ of an image transform is separable and symmetric, then the transform $g(u,v)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}T(u,v,x,y)f(x,y)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}T_1(u,x)T_1(v,y)f(x,y)$ can be written in matrix form as follows

$$\underline{g}=\underline{T}_1\cdot\underline{f}\cdot\underline{T}_1^{\,T}$$

where $\underline{f}$ is the original image of size $N\times N$, and $\underline{T}_1$ is an $N\times N$ transformation matrix with elements $t_{ij}=T_1(i,j)$. If, in addition, $\underline{T}_1$ is a unitary matrix then the transform is called **separable unitary** and the original image is recovered through the relationship

$$\underline{f}=\underline{T}_1^{\,*T}\cdot\underline{g}\cdot\underline{T}_1^{\,*}$$

THE TWO DIMENSIONAL FOURIER TRANSFORM

Continuous space and continuous frequency

The Fourier transform is extended to a function $f(x,y)$ of two variables. If $f(x,y)$ is continuous and integrable and $F(u,v)$ is integrable, the following Fourier transform pair exists:

$$F(u,v)=\int_{-\infty}^{\infty}\int f(x,y)e^{-j2\pi(ux+vy)}dxdy$$

$$f(x,y)=\frac{1}{(2\pi)^2}\int_{-\infty}^{\infty}\int F(u,v)e^{j2\pi(ux+vy)}dudv$$

In general $F(u,v)$ is a complex-valued function of two real frequency variables u,v and hence, it can be written as:

$$F(u,v)=R(u,v)+jI(u,v)$$

The amplitude spectrum, phase spectrum and power spectrum, respectively, are defined as follows.

$$|F(u,v)|=\sqrt{R^2(u,v)+I^2(u,v)}$$

$$\phi(u,v)=\tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$$

$$P(u,v)=|F(u,v)|^2=R^2(u,v)+I^2(u,v)$$

Discrete space and continuous frequency

For the case of a discrete sequence $f(x,y)$ of infinite duration we can define the 2-D discrete space Fourier transform pair as follows

$$F(u,v) = \sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} f(x,y) e^{-j(xu+vy)}$$

$$f(x,y) = \frac{1}{(2\pi)^2} \int_{u=-\pi}^{\pi} \int_{v=-\pi}^{\pi} F(u,v) e^{j(xu+vy)} du dv$$

$F(u,v)$ is again a complex-valued function of two real frequency variables u,v and it is periodic with a period $2\pi \times 2\pi$, that is to say $F(u,v) = F(u+2\pi,v) = F(u,v+2\pi)$ The Fourier transform of $f(x,y)$ is said to converge uniformly when $F(u,v)$ is finite and

$$\lim_{N_1 \to \infty} \lim_{N_2 \to \infty} \sum_{x=-N_1}^{N_1} \sum_{y=-N_2}^{N_2} f(x,y) e^{-j(xu+vy)} = F(u,v) \text{ for all } u,v.$$

When the Fourier transform of $f(x,y)$ converges uniformly, $F(u,v)$ is an analytic function and is infinitely differentiable with respect to u and v.

Discrete space and discrete frequency: The two dimensional Discrete Fourier Transform (2-D DFT)

If $f(x,y)$ is an $M \times N$ array, such as that obtained by sampling a continuous function of two dimensions at dimensions M and N on a rectangular grid, then its two dimensional Discrete Fourier transform (DFT) is the array given by

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M+vy/N)}$$

$u = 0,\ldots,M-1,\ v = 0,\ldots,N-1$

and the inverse DFT (IDFT) is

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(ux/M+vy/N)}$$

When images are sampled in a square array, $M = N$ and

$$F(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux+vy)/N}$$

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(ux+vy)/N}$$

It is straightforward to prove that the two dimensional Discrete Fourier Transform is separable, symmetric and unitary.

Properties of the 2-D DFT

Most of them are straightforward extensions of the properties of the 1-D Fourier Transform. Advise any introductory book on Image Processing.

Completeness

The discrete Fourier transform is an invertible, linear transformation $\mathcal{F} : \mathbb{C}^N \to \mathbb{C}^N$

with **C** denoting the set of complex numbers. In other words, for any *N* > 0, an *N*-dimensional complex vector has a DFT and an IDFT which are in turn *N*-dimensional complex vectors.

Orthogonality

The vectors $e^{\frac{2\pi i}{N}kn}$ form an orthogonal basis over the set of *N*-dimensional complex vectors:

$$\sum_{n=0}^{N-1} \left(e^{\frac{2\pi i}{N}kn}\right) \left(e^{-\frac{2\pi i}{N}k'n}\right) = N\ \delta_{kk'}$$

Where $\delta_{kk'}$ is the Kronecker delta. This orthogonality condition can be used to derive the formula for the IDFT from the definition of the DFT, and is equivalent to the unitarity property below.

The Plancherel theorem and Parseval's theorem

If *Xk* and *Yk* are the DFTs of *xn* and *yn* respectively then the Plancherel theorem states:

$$\sum_{n=0}^{N-1} x_n y_n^* = \frac{1}{N} \sum_{k=0}^{N-1} X_k Y_k^*$$

where the star denotes complex conjugation. Parseval's theorem is a special case of the Plancherel theorem and states:

$$\sum_{n=0}^{N-1} |x_n|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X_k|^2.$$

These theorems are also equivalent to the unitary condition.

Periodicity

If the expression that defines the DFT is evaluated for all integers *k* instead of just for k=0,1,.. N-1, then the resulting infinite sequence is a periodic extension of the DFT, periodic with period *N*. The periodicity can be shown directly from the definition:

$$X_{k+N} \stackrel{\text{def}}{=} \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N}(k+N)n} = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N}kn} \underbrace{e^{-2\pi in}}_{1} = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N}kn} = X_k.$$

Similarly, it can be shown that the IDFT formula leads to a periodic extension.

The shift theorem

Multiplying *xn* by a *linear phase* $e^{\frac{2\pi i}{N}nm}$ for some integer *m* corresponds to a *circular shift* of the output *Xk*: *Xk* is replaced by *Xk* − *m*, where the subscript is interpreted modulo *N* (i.e., periodically). Similarly, a circular shift of the input *xn* corresponds to multiplying the output *Xk* by a linear phase. Mathematically, if {*xn*} represents the vector **x** then

If $\mathcal{F}(\{x_n\})_k = X_k$

Then $\mathcal{F}(\{x_n \cdot e^{\frac{2\pi i}{N}nm}\})_k = X_{k-m}$

And $\mathcal{F}(\{x_{n-m}\})_k = X_k \cdot e^{-\frac{2\pi i}{N}km}$

Circular convolution theorem and cross-correlation theorem

The convolution theorem for the continuous and discrete time Fourier transforms indicates that a convolution of two infinite sequences can be obtained as the inverse transform of the product of the individual transforms. With sequences and transforms of length N, a circularity arises**:**

$$\begin{aligned}\mathcal{F}^{-1}\{\mathbf{X}\cdot\mathbf{Y}\}_n &\stackrel{\text{def}}{=} \frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot Y_k \cdot e^{\frac{2\pi i}{N}kn} \\ &= \frac{1}{N}\sum_{k=0}^{N-1}\left(\sum_{l=0}^{N-1} x_l e^{-\frac{2\pi i}{N}kl}\right)\cdot\left(\sum_{m=0}^{N-1} y_m e^{-\frac{2\pi i}{N}km}\right)\cdot e^{\frac{2\pi i}{N}kn} \\ &= \sum_{l=0}^{N-1} x_l \sum_{m=0}^{N-1} y_m \left(\frac{1}{N}\sum_{k=0}^{N-1} e^{\frac{2\pi i}{N}k(n-l-m)}\right).\end{aligned}$$

The quantity in parentheses is 0 for all values of *m* except those of the form *n* − *l* − *pN*, where *p* is any integer. At those values, it is 1. It can therefore be replaced by an infinite sum of Kronecker delta functions, and we continue accordingly. Note that we can also extend the limits of *m* to infinity, with the understanding that the *x* and *y* sequences are defined as 0 outside [0,N-1]**:**

$$\begin{aligned}\mathcal{F}^{-1}\{\mathbf{X}\cdot\mathbf{Y}\}_n &= \sum_{l=0}^{N-1} x_l \sum_{m=-\infty}^{\infty} y_m \left(\sum_{p=-\infty}^{\infty} \delta_{m,(n-l-pN)}\right) \\ &= \sum_{l=0}^{N-1} x_l \sum_{p=-\infty}^{\infty} \underbrace{\left(\sum_{m=-\infty}^{\infty} y_m \cdot \delta_{m,(n-l-pN)}\right)}_{y_{n-l-pN}} \\ &= \sum_{l=0}^{N-1} x_l \left(\sum_{p=-\infty}^{\infty} y_{n-l-pN}\right) \ \stackrel{\text{def}}{=}\ (\mathbf{x} * \mathbf{y_N})_n \ ,\end{aligned}$$

which is the convolution of the X sequence with a periodically extended Y sequence defined by**:**

$$(\mathbf{y_N})_n \stackrel{\text{def}}{=} \sum_{p=-\infty}^{\infty} y_{(n-pN)}.$$

Similarly, it can be shown that**:**

$$\mathcal{F}^{-1}\{\mathbf{X}^*\cdot\mathbf{Y}\}_n = \sum_{l=0}^{N-1} x_l^* \cdot (y_N)_{n+l} \ \stackrel{\text{def}}{=}\ (\mathbf{x} \star \mathbf{y_N})_n \ ,$$

which is the cross-correlation of X and Y_N

A direct evaluation of the convolution or correlation summation (above) requires $O(N2)$ operations for an output sequence of length N. An indirect method, using transforms, can take advantage of the $O(N\log N)$ efficiency of the fast Fourier transform (FFT) to achieve much better performance. Furthermore, convolutions can be used to efficiently compute DFTs via Rader's FFT algorithm and Bluestein's FFT algorithm.

FAST FOURIER TRANSFORM (FFT)

Basically, the computational problem for the DFT is to compute the sequence $\{X(k)\}$ of N complex-valued numbers given another sequence of data $\{x(n)\}$ of length N, according to the formula

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \qquad 0 \le k \le N-1$$

$$W_N = e^{-j2\pi/N}$$

In general, the data sequence $x(n)$ is also assumed to be complex valued. Similarly, The IDFT becomes

$$x(n) = \frac{1}{N}\sum_{n=0}^{N-1} X(k) W_N^{-nk}, \qquad 0 \le n \le N-1$$

For each value of k, direct computation of $X(k)$ involves N complex multiplications ($4N$ real multiplications) and N-1 complex additions ($4N$-2 real additions). Consequently, to compute all N values of the DFT requires N 2 complex multiplications and N 2-N complex additions. Direct computation of the DFT is basically inefficient primarily because it does not exploit the symmetry and periodicity properties of the phase factor WN. In particular, these two properties are :

$$\text{Symmetry property}: W_N^{k+N/2} = -W_N^k$$
$$\text{Periodicit y property}: W_N^{k+N} = W_N^k$$

The computationally efficient algorithms described in this section, known collectively as fast Fourier transform (FFT) algorithms, exploit these two basic properties of the phase factor.

SEPARABLE IMAGE TRANSFORMS

THE DISCRETE COSINE TRANSFORM (DCT) (NOVEMBER 2012) (APR 2014)

One dimensional signals

This is a transform that is similar to the Fourier transform in the sense that the new independent variable represents again frequency. The DCT is defined below.

$$C(u) = a(u)\sum_{x=0}^{N-1} f(x)\cos\left[\frac{(2x+1)u\pi}{2N}\right], \quad u = 0,1,\ldots,N-1$$

with $a(u)$ a parameter that is defined below.

$$a(u) = \begin{cases} \sqrt{1/N} & u = 0 \\ \sqrt{2/N} & u = 1,\ldots,N-1 \end{cases}$$

The inverse DCT (IDCT) is defined below.

$$f(x) = \sum_{u=0}^{N-1} a(u)C(u)\cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

Two dimensional signals (images)

For 2-D signals it is defined as

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

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

$a(u)$ is defined as above and $u,v = 0,1,\ldots,N-1$

Properties of the DCT transform

- The DCT is a real transform. This property makes it attractive in comparison to the Fourier transform.
- The DCT has excellent energy compaction properties. For that reason it is widely used in image compression standards (as for example JPEG standards).
- There are fast algorithms to compute the DCT, similar to the FFT for computing the DFT.

WALSH TRANSFORM (WT) (APR 2012), (APR 2013)

One dimensional signals

This transform is slightly different from the transforms you have met so far. Suppose we have a function $f(x)$, $x = 0,..,N-1$ where $N = 2^n$ and its Walsh transform $W(u)$.

If we use binary representation for the values of the independent variables x and u we need n bits to represent them. Hence, for the binary representation of x and u we can write:

$(x)_{10} = (b_{n-1}(x)b_{n-2}(x)\ldots b_0(x))_2$, $(u)_{10} = (b_{n-1}(u)b_{n-2}(u)\ldots b_0(u))_2$
with $b_i(x)$ 0 or 1 for $i = 0,\ldots,n-1$.

Example
If $f(x), x = 0,\ldots,7$, (8 samples) then $n = 3$ and for $x = 6$,
$6 = (110)_2 \Rightarrow b_2(6) = 1, b_1(6) = 1, b_0(6) = 0$

We define now the 1-D Walsh transform as

$$W(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right] \text{ or}$$

$$W(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=0}^{n-1} b_i(x)b_{n-1-i}(u)}$$

The array formed by the Walsh kernels is again a symmetric matrix having orthogonal rows and columns. Therefore, the Walsh transform is and its elements are of the form $T(u,x) = \prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$. You can immediately observe that $T(u,x) = 0$ or 1 depending on the values of $b_i(x)$ and $b_{n-1-i}(u)$. If the Walsh transform is written in a matrix form

$$\underline{W} = \underline{T}\cdot\underline{f}$$

the rows of the matrix $\underline{T}$ which are the vectors $[T(u,0)\,T(u,1)\ldots T(u,N-1)]$ have the form of square waves. As the variable u (which represents the index of the transform) increases, the corresponding square wave's "frequency" increases as well. For example for $u = 0$ we see that $(u)_{10} = (b_{n-1}(u)b_{n-2}(u)\ldots b_0(u))_2 = (00\ldots0)_2$ and hence, $b_{n-1-i}(u) = 0$, for any i. Thus, $T(0,x) = 1$ and $W(0) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)$. We see that the first element of the Walsh transform in the mean of the original function $f(x)$ (the DC value) as it is the case with the Fourier transform.

The inverse Walsh transform is defined as follows.

$$f(x) = \sum_{u=0}^{N-1} W(u)\left[\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right] \text{ or}$$

$$f(x) = \sum_{u=0}^{N-1} W(u)(-1)^{\sum_{i=0}^{n-1} b_i(x)b_{n-1-i}(u)}$$

Two dimensional signals

The Walsh transform is defined as follows for two dimensional signals.

$$W(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}\right] \text{ or}$$

$$W(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)(-1)^{\sum_{i=0}^{n-1}(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}$$

The inverse Walsh transform is defined as follows for two dimensional signals.

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} W(u,v)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}\right] \text{ or}$$

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} W(u,v)(-1)^{\sum_{i=0}^{n-1}(b_i(x)b_{n-1-i}(u)+b_i(y)b_{n-1-i}(v))}$$

Properties of the Walsh Transform

Unlike the Fourier transform, which is based on trigonometric terms, the Walsh transform consists of a series expansion of basis functions whose values are only -1 or 1 and they have the form of square waves. These functions can be implemented more efficiently in a digital environment than the exponential basis functions of the Fourier transform.

The forward and inverse Walsh kernels are identical except for a constant multiplicative factor of for 1/ N for 1-D signals.

The forward and inverse Walsh kernels are identical for 2-D signals. This is because the array formed by the kernels is a symmetric matrix having orthogonal rows and columns, so its inverse array is the same as the array itself.

The concept of frequency exists also in Walsh transform basis functions. We can think of frequency as the number of zero crossings or the number of transitions in a basis vector and we call this number **sequency**. The Walsh transform exhibits the property of energy compaction as all the transforms.

For the fast computation of the Walsh transform there exists an algorithm called **Fast Walsh Transform (FWT).** This is a straightforward modification of the FFT.

HADAMARD TRANSFORM (HT)

Definition

In a similar form as the Walsh transform, the 2-D Hadamard transform is defined as follows.

Forward

$$H(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_i(u)+b_i(y)b_i(v))}\right],\ N=2^n \text{ or}$$

$$H(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)(-1)^{\sum_{i=0}^{n-1}(b_i(x)b_i(u)+b_i(y)b_i(v))}$$

Inverse

$$f(x,y)=\frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}H(u,v)\left[\prod_{i=0}^{n-1}(-1)^{(b_i(x)b_i(u)+b_i(y)b_i(v))}\right] \text{ etc.}$$

Properties of the Hadamard Transform

- The Hadamard transform differs from the Walsh transform only in the order of basis functions.
- The order of basis functions of the Hadamard transform **does not** allow the fast computation of it by using a straightforward modification of the FFT.
- Extended version of the Hadamard transform is the **Ordered Hadamard Transform** for which a fast algorithm called **Fast Hadamard Transform (FHT)** can be applied.
- An important property of Hadamard transform is that, letting H_N represent the matrix of order N, the recursive relationship is given by the expression

$$H_{2N}=\begin{bmatrix}H_N & H_N\\ H_N & -H_N\end{bmatrix}$$

KARHUNEN-LOEVE (KLT) or HOTELLING TRANSFORM

The Karhunen-Loeve Transform or KLT was originally introduced as a series expansion for continuous random processes by Karhunen and Loeve. For discrete signals Hotelling first studied what was called a method of principal components, which is the discrete equivalent of the KL series expansion. Consequently, the KL transform is also called the Hotelling transform or the method of principal components. **The term KLT is the most widely used**.

The case of many realisations of a signal or image (Gonzalez/Woods)

The concepts of **eigenvalue** and **eigevector** are necessary to understand the KL transform.

If C is a matrix of dimension $n \quad n$, then a scalar is called an eigenvalue of C if there is a nonzero vector e in R^n such that

$Ce \quad e$

The vector e is called an eigenvector of the matrix C corresponding to the eigenvalue .

Consider a population of random vectors of the form

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

The **mean vector** of the population is defined as

$$\underline{m}_x = E\{\underline{x}\}$$

The operator E refers to the expected value of the population calculated theoretically using the probability density functions (pdf) of the elements x_i and the joint probability density functions between the elements x_i and x_j . The covariance matrix of the population is defined as

$$\underline{C}_x = E\left\{(\underline{x} - \underline{m}_x)(\underline{x} - \underline{m}_x)^T\right\}$$

Because $\underline{x}$ is n-dimensional, $\underline{C}_x$ and $(\underline{x}-\underline{m}_x)(\underline{x}-\underline{m}_x)^T$ are matrices of order $n \times n$. The element c_{ii} of $\underline{C}_x$ is the variance of x_i, and the element c_{ij} of $\underline{C}_x$ is the covariance between the elements x_i and x_j. If the elements x_i and x_j are uncorrelated, their covariance is zero and, therefore, $c_{ij} = c_{ji} = 0$.

For M vectors from a random population, where M is large enough, the mean vector and covariance matrix can be approximately calculated from the vectors by using the following relationships where all the expected values are approximated by summations

$$\underline{m}_x = \frac{1}{M}\sum_{k=1}^{M}\underline{x}_k$$

$$\underline{C}_x = \frac{1}{M}\sum_{k=1}^{M}\underline{x}_k\underline{x}_k^T - \underline{m}_x\underline{m}_x^T$$

Very easily it can be seen that $\underline{C}_x$ is real and symmetric. In that case a set of n orthonormal (**at this point you are familiar with that term**) eigenvectors always exists. Let $\underline{e}_i$ and λ_i, $i = 1,2,\ldots,n$, be this set of eigenvectors and corresponding eigenvalues of $\underline{C}_x$, arranged in descending order so that $\lambda_i \geq \lambda_{i+1}$ for $i = 1,2,\ldots,n-1$. Let $\underline{A}$ be a matrix whose rows are formed from the eigenvectors of $\underline{C}_x$, ordered so that the first row of $\underline{A}$ is the eigenvector corresponding to the largest eigenvalue, and the last row the eigenvector corresponding to the smallest eigenvalue.

Suppose that $\underline{A}$ is a transformation matrix that maps the vectors $\underline{x}'s$ into vectors $\underline{y}'s$ by using the following transformation

$$\underline{y} = \underline{A}(\underline{x}-\underline{m}_x)$$

The above transform is called the Karhunen-Loeve or Hotelling transform.

The case of one realisation of a signal or image

The derivation of the KLT for the case of one image realisation assumes that the two dimensional signal (image) is **ergodic**. This assumption allows us to calculate the statistics of the image using only one realisation. Usually we divide the image into blocks and we apply the KLT in each block. This is reasonable because the 2-D field is likely to be ergodic within a small block since the nature of the signal changes within the whole image. Let's suppose that $\underline{f}$ is a vector obtained by lexicographic ordering of the pixels $f(x,y)$ within a block of size $M \times M$ (placing the rows of the block sequentially).
The mean vector of the random field inside the block is a scalar that is estimated by the approximate relationship

$$m_f = \frac{1}{M^2} \sum_{k=1}^{M^2} \underline{f}(k)$$

and the covariance matrix of the 2-D random field inside the block is $\underline{C}_f$ where

$$c_{ii} = \frac{1}{M^2} \sum_{k=1}^{M^2} \underline{f}(k)\underline{f}(k) - m_f^2$$

and

$$c_{ij} = c_{i-j} = \frac{1}{M^2} \sum_{k=1}^{M^2} \underline{f}(k)\underline{f}(k+i-j) - m_f^2$$

After knowing how to calculate the matrix $\underline{C}_f$, the KLT for the case of a single realisation is the same as described above.

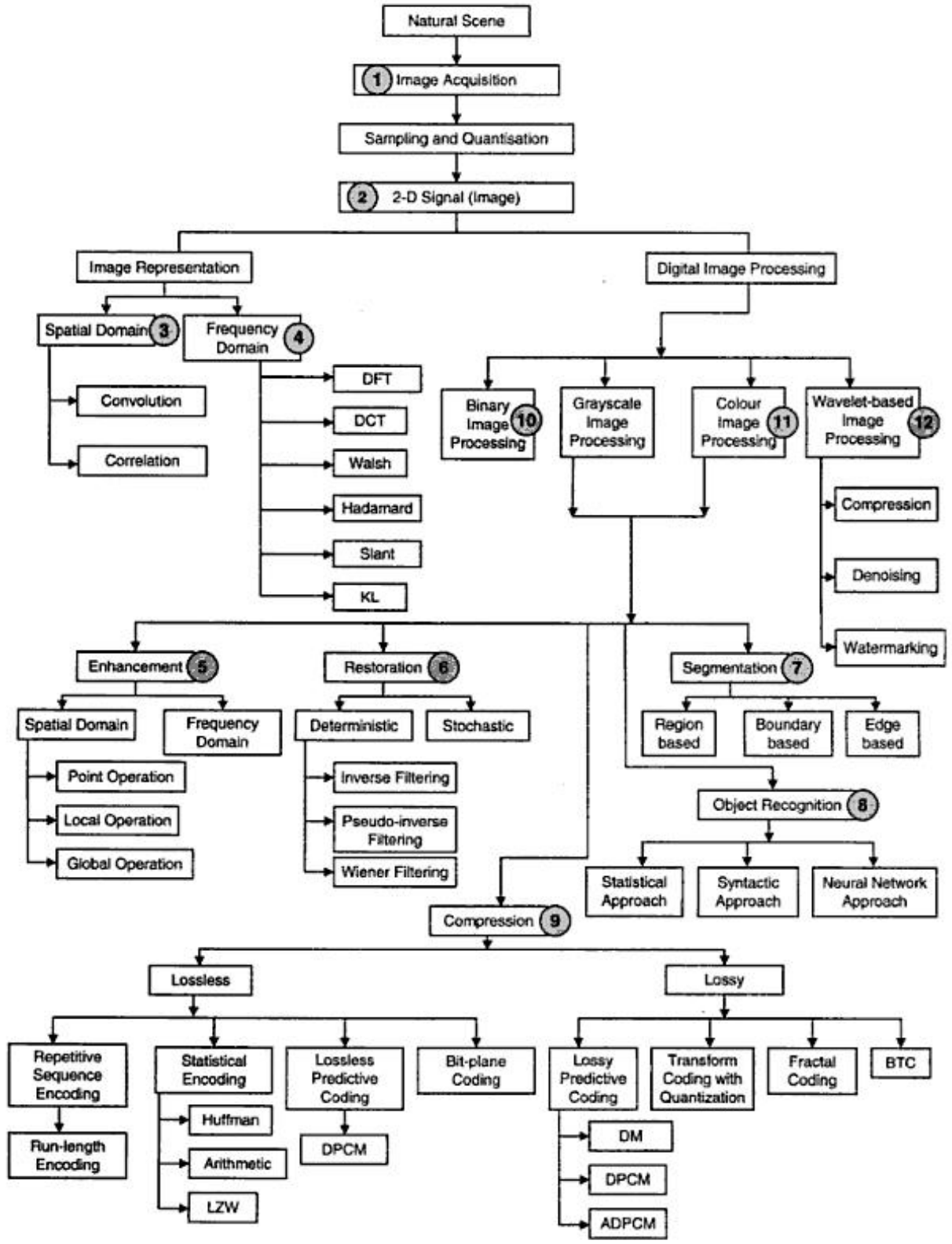
Properties of the Karhunen-Loeve transform

Despite its favourable theoretical properties, the KLT is not used in practice for the following reasons.

Its basis functions depend on the covariance matrix of the image, and hence they have to recomputed and transmitted for every image.

Perfect decorrelation is not possible, since images can rarely be modelled as realisations of ergodic fields. There are no fast computational algorithms for its implementation.

IMAGE-PROCESSING TAXONOMY



APPLICATIONS:

Digital image processing has a broad spectrum of applications, such as remote sensing via satellites and other spacecrafts, image transmission and storage for business applications, medical processing, radar, sonar, and acoustic image processing, robotics, and automated inspection of industrial parts.

Image acquired by satellites are useful in tracking of earth resources; geographical mapping; prediction of agricultural crops, urban growth, and weather; flood and fire control; and many other environmental applications. Space image application include recognition and analysis of objects contained in images obtained from deep space-probe missions. Image transmission and storage applications occur in broadcast television, teleconferencing, transmission of facsimile images(printed documents and graphics) for office automation, communication over computer networks, closed-circuit television based security monitoring systems, and in military communications. In medical applications one is concerned with processing of chest X rays, cineangiograms, projection images of transaxial tomography, and other medical images that occur in radiology, nuclear magnetic resonance(NMR), and ultrasonic scanning. These images may be used for patient screening and monitoring or for detection of tumors or other disease in patients. Radar and sonar images are used for detection and recognition or various types of targets or in guidance and maneuvering of aircraft or missile systems. Figure 1.3 shows examples of several different types of images. There are many other applications ranging from robot vision for industrial automation to image synthesis for cartoon making or fashion design. In other words, whenever a human or a machine or any other entity receives data of two or more dimensions, an image is processed.

IMAGE FORMATION:

What is an image?

A digital image can be considered as a discrete representation of data possessing both spatial (layout) and intensity (colour) information. We can also consider treating an image as a multidimensional signal.

The two-dimensional (2-D) discrete, digital image represents the response of some sensor (or simply a value of some interest) at a series of fixed positions (m ¼ 1; 2; . . . ;M; n ¼ 1; 2; . . . ;N) in 2-D Cartesian coordinates and is derived from the 2-D continuous spatial signal through a sampling process frequently referred to as discretization. Discretization occurs naturally with certain types of imaging sensor (such as CCD cameras) and basically effects a local averaging of the continuous signal over some small (typically square) region in the receiving domain.

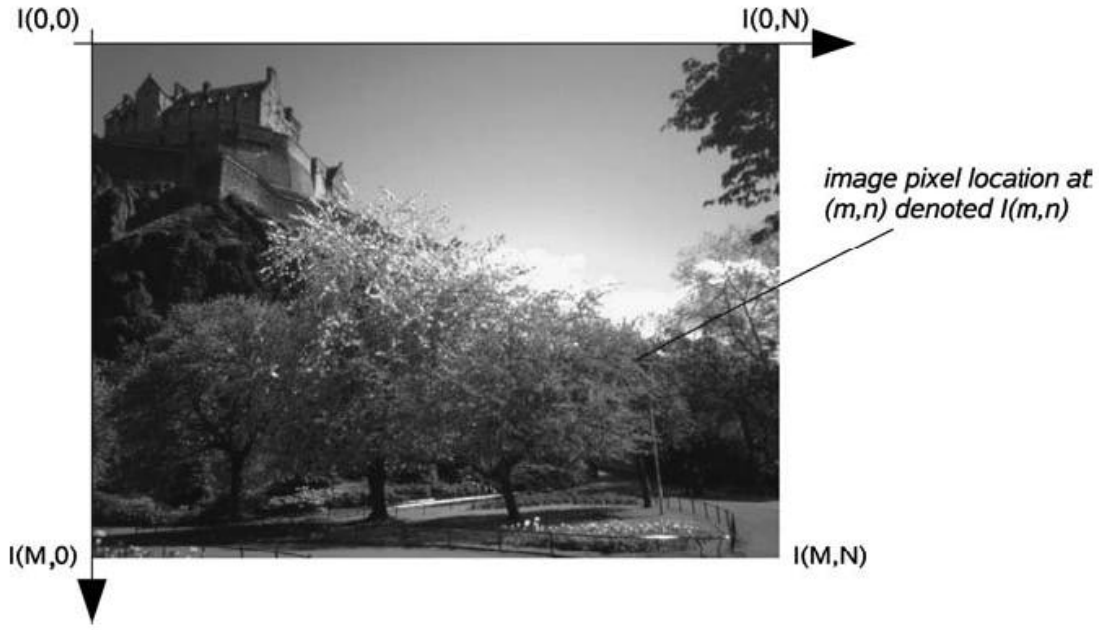


Figure 1.1 The 2-D Cartesian coordinate space of an M x N digital image

IMAGE FORMATION:

How is an image formed?

The image formation process can be summarized as a small number of key elements. In general, a digital image s can be formalized as a mathematical model comprising a functional representation of the scene (the object function o) and that of the capture process (the point spreadfunction (PSF) p).

Additionally, the image will contain additive noise n. These are essentially combined as follows to form an image:

Image ¼ PSF _ object function þ noise s ¼ p _ oþn ð2:1Þ

In this process we have several key elements:

- PSF this describes the way information on the object function is spread as a result of recording the data. It is a characteristic of the imaging instrument (i.e. camera) and is a deterministic function (that operates in the presence of noise).

- Object function This describes the object (or scene) that is being imaged (its surface or internal structure, for example) and the way light is reflected from that structure to the imaging instrument.

- Noise This is a nondeterministic function which can, at best, only be described in terms of some statistical noise distribution (e.g. Gaussian). Noise is a stochastic function which is a consequence of all the unwanted external disturbances that occur during the recording of the image data.

- Convolution operator _ A mathematical operation which ‘smears’ (i.e. convolves) one function with another.

Here, the function of the light reflected from the object/scene (object function) is transformed into the image data representation by convolution with the PSF. This function characterizes the image formation (or capture) process. The process is affected by noise.

The PSF is a characteristic of the imaging instrument (i.e. camera). It represents the response of the system to a point source in the object plane, as shown in Fig where we can also consider an imaging system as an input distribution (scene light) to output distribution (image pixels) mapping function consisting both of the PSF itself and additive noise

From this overview we will consider both the mathematical representation of the image formation process, useful as a basis for our later consideration of advanced image-processing representations, and from an engineering perspective in terms of physical camera imaging

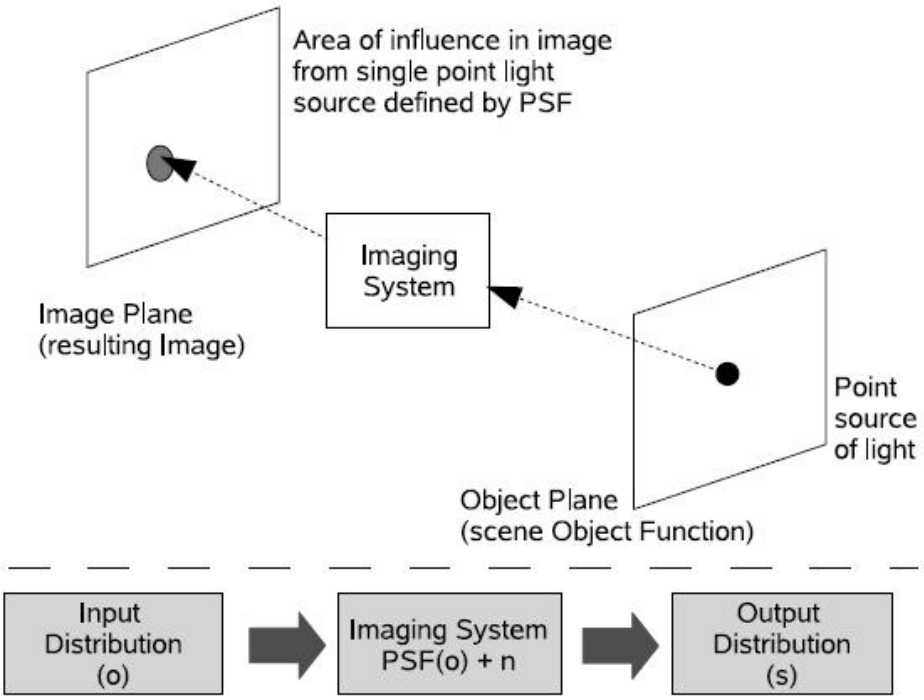


Figure 2.1 An overview of the image formation 'system' and the effect of the PSF

THE MATHEMATICS OF IMAGE FORMATION:

The formation process of an image can be represented mathematically. In our later consideration of various processing approaches this allows us to reason mathematically using knowledge of the conditions under which the image originates.

INTRODUCTION:

In a general mathematical sense, we may view image formation as a process which transforms an input distribution into an output distribution. Thus, a simple lens may be viewed as a 'system' that transforms a spatial distribution of light in one domain (the object plane) to a distribution in another (the image plane).

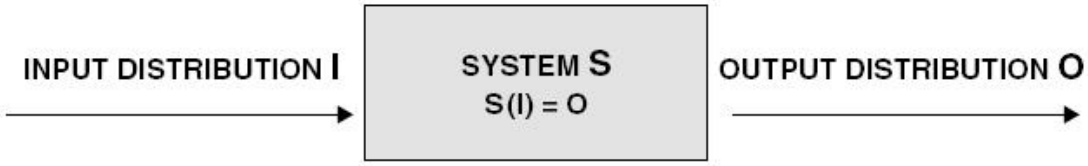


Figure 2.2 Systems approach to imaging. The imaging process is viewed as an operator *S* which acts on the input distribution *I* to produce the output *O*

Any imaging device is a system, or a 'black box', whose properties are defined by the way in which an input distribution is mapped to an output distribution. Figure 2.2 summarizes this concept.

The process by which an imaging system transforms the input into an output can be viewed from an alternative perspective, namely that of the Fourier or frequency domain.

From this perspective, images consist of a superposition of harmonic functions of different frequencies. Imaging systems then act upon the spatial frequency content of the input to produce an output with a modified spatial frequency content. Frequency-domain methods are powerful and important in image processing, and we will offer a discussion of such methods later. First however, we are going to devote some time to understanding the basic mathematics of image formation.

PONDICHERRY UNIVERSITY QUESTIONS

GRAPHICS AND IMAGE PROCESSING

1. Write about Hotelling Transform. **(APRIL / MAY 2012)**
2. Discuss about Walsh Transform. **(APR 2012), (APR 2013) (Pg.no.18)**
3. Discuss Discrete Cosine Transform in detail. **(NOVEMBER 2012) (APR 2014)**
4. Explain Hotelling transform and the principle of an image **(NOVEMBER 2012)**
5. Discuss about Hadamard Transform.**(APRIL 2013)**
6. What is Discrete Fourier Transform? Explain about its functionality in detail. **(NOVEMBER 2013)**
7. Discuss about the conversion schemes between data classes and image types in detail. **(NOVEMBER 2013)**
8. Describe the basic relationship and distance measure between pixels. **(APR 2014)**

UNIT-IV

Image Enhancement and Restoration: Image Quality and need for Enhancements – Point operations - Histogram Techniques – Spatial filtering concepts – Frequency Domain Filtering – Image Smoothening – Image Sharpening - Image degradation and Noise Models – Introduction to Restoration Techniques.

HISTOGRAM MODELING: (NOV 2012)

It is one of the powerful techniques in image enhancement. The histogram of an image represents the relative frequency of occurrence of the various gray levels in the image. Histogram may be modified to have a desired shape. This is useful in stretching the low-contrast levels of images with narrow histograms.

HISTOGRAM EQUALIZATION: (APR 2014)

The histogram equalization operation is shown in the figure. A cumulative histogram of the function is found and then uniform quantization is carried out.

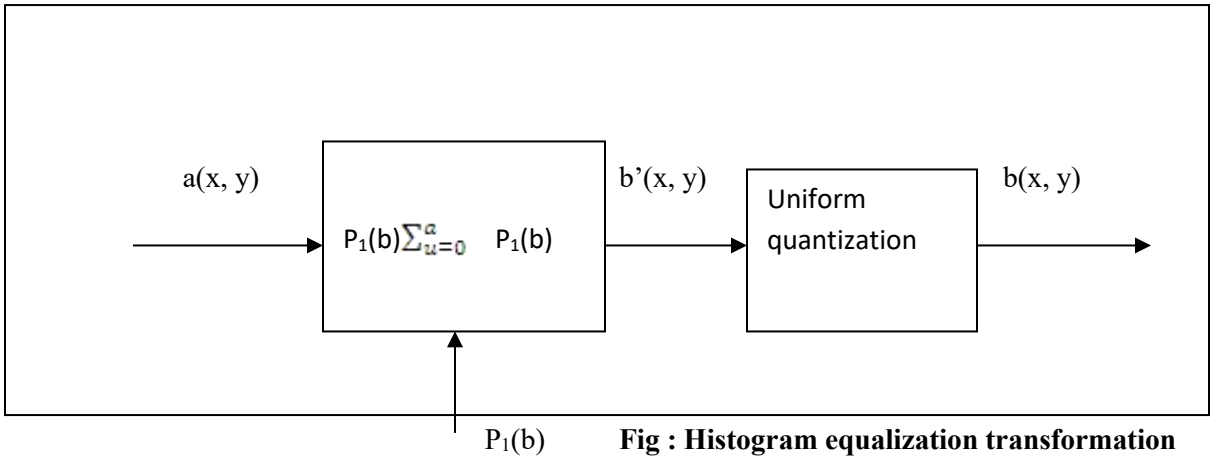


Fig : Histogram equalization transformation

This technique for equalizing the histogram gives the best possible dynamic range and strong contrast. An image is more visible if we use this technique.

Let the variable r represent the gray levels in the image to be enhanced. Initially assume that pixel values are continuous quantities that have been normalized .So they lie in the interval [0,1] with r=0 represent blank and r=1 represent white.

For any value r in the interval [0,1] the transformation is given by S=T (r)

Which produce a level for every pixel value r in the original image.

This equation satisfied the **conditions**.

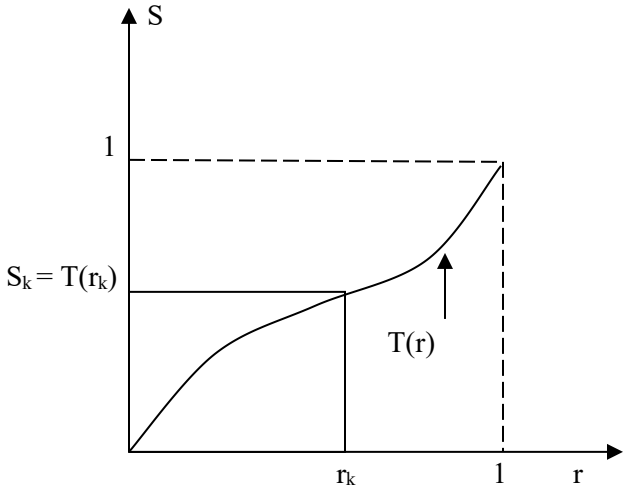
(i) T(r) is single valued and monotonically increased in the interval $0 \leq r \leq 1$.

(i.e. preserves the order from blank to white in the gray scale)

(ii) $0 \leq T(r) \leq 1$ for $0 \leq r \leq 1$

(i.e. guarantees a mapping that is consistent with the allowed range of pixel values)

FIGURE: Shows the transformation function satisfying these conditions.



A gray level transformation function. The inverse transformation is given by

$$\mathbf{r = T^{-1}(S) \qquad 0 \leq S \leq 1}$$

The gray levels in an image may be as random quantities in the interval [0,1]. If they are continuous variables, the original and transformed gray levels can be characterized by their probability density functions $P_r(r)$ and $P_s(s)$ respectively.

If $P_r(r)$ and T(r) are known and $T^{-1}(S)$ satisfies the conditions (i), the probability density function of the transformed gray level is

$$\mathbf{P_s(s) = [P_r(r) dr/ds]_{r = T^{-1}(S)}}$$

Consider the transformation

$$\mathbf{S = T(r)} = \int_0^r P_r(w)\,dw \qquad \mathbf{0 \leq r \leq 1}$$

Where w is the dummy variable of integration.

The right side value of equation represents the cumulative distribution function **(CDF)** if **r.** Conditions (i) and (ii) are satisfied by this transformed function because the CDF increases monotonically from 0 to 1 as a function of r.

From equation, the derivative of S with respect to r is

$ds/dr = P_r(r)$

substitute dr/ds value into equation

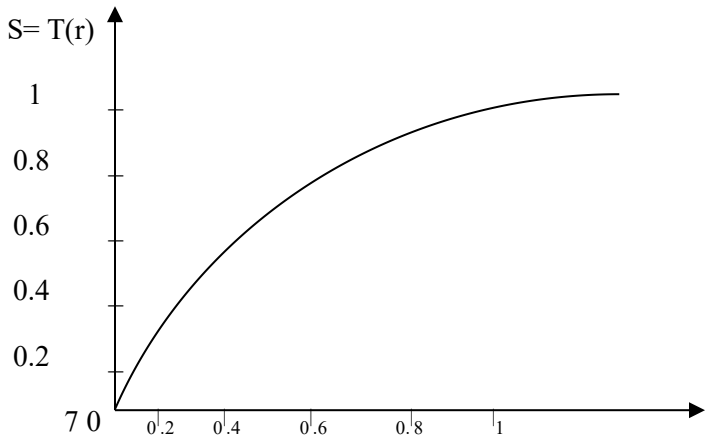
$$\mathbf{P_s(s) = [(P_r(r).1)/P_r(r)]_{r=T^{-1}(S)}}$$

$$\mathbf{= [1]\ r = T^{-1}(S)}$$

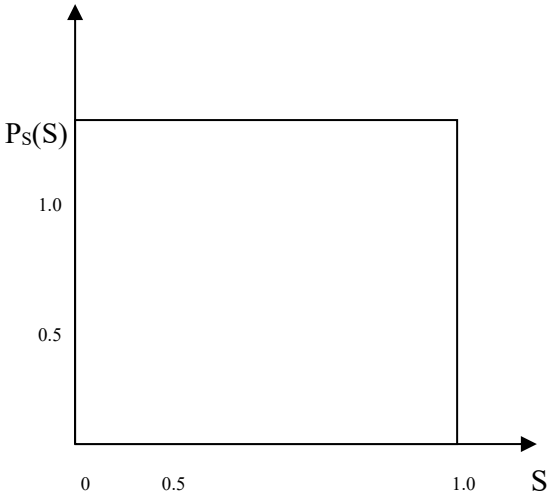
$$\mathbf{= 1 \qquad 0 \le S \le 1}$$

Which is a uniform density function.

FOR EXAMPLE:



Transformation function



Resulting uniform density

From the fig the levels r have the probability density function $P_r(r)$ which is given by

$$P_r(r) = \begin{cases} -2r + 2 & 0 \le r \le 1 \\ 0 & \text{Otherwise} \end{cases}$$

Substitute this value in equation which gives the transformation function.

$$S = T(r) = \int_0^r (-2w + 2)\,dw$$

$$= -r^2 + 2r$$

Inverse transform is

$$r = T^{-1}(S)$$

$$= 1 \pm \sqrt{1 - S}$$

Since r lies in the interval [0, 1]

$r = T^{-1}(S) = 1 - \sqrt{1 - S}$ is valid.

The probability density function is given by

$$P_s(S) = \left[P_r(r) \frac{dr}{ds} \right]_{r = T^{-1}(S)}$$

$$= \left[(-2r + 2) \frac{dr}{ds} \right]_{r = 1 - \sqrt{1 - S}}$$

$= [(-2[1-\sqrt{1 - S}]+2)d(1-\sqrt{1 - S}\ )/ds]$

$= [-2+2\sqrt{1 - S}+2].d(1-\sqrt{1 - S}\ )/ds$

$= 2\sqrt{1 - S}\ .d(1-\sqrt{1 - S}\ )/ds$

$=1 \quad 0 \le S \le 1$

Which is a uniform density in the desired range.

For gray levels that take on discrete values, then p.d.f. is

$P_r(r_k) = n_k/n$ $\quad 0 \leq nk \leq 1$ and K=0,1,2,…L-1

Where, L is the number of levels

$P_r(rk)$ is the probability of the Kth gray level.

n_k is the number of times this level appear in the image.

n is the total no. of pixels in the image.

A plot of $P_r(rk)$ Vs rk is called 'histogram' and the technique used for the obtaining a uniform histogram is known as histogram equalization or histogram linearization.

The discrete form of equation (2) is

$\mathbf{S_k = T(r_k) = \sum_{j=0}^{k} nj/n}$

$= \sum_{j=0}^{k} Pr(rj)$ $\quad$ **0≤ rk ≤1 and K=0,1,2,…L-1**

The inverse transformation is

$\mathbf{r_k = T^{-1}(S_k)}$ $\quad$ **0≤ Sk ≤1**

HISTOGRAM SPECIFICATION:

The histogram equalization does not used in image enhancement applications because it is capable of generating only one result i.e. uniform histogram.

Sometimes the ability to specify particular histogram shapes capable of highlighting certain gray level ranges in an image is desirable.

Let $P_r(r)$ and $p_d(d)$ be the original and desired p. d .f. respectively.

The histogram equalization of original image is

S=T(r) = $\int_0^r Pr(w)\, dw$

The histogram equalization of desired image is

V=G (d) = $\int_0^a Pd(w)\, dw$

The inverse process is d=G^-1(V), which gives the d of the desired image.

$P_s(S)$ and $P_v(V)$ are the identical uniform densities, because, the final result is independent of the density inside the integral.

The procedure is

(i)To equalize the levels of the original image using transformation equation i.e.(2)

(ii)Specify the desired density function and get the transformation function G(d) using equation (2.3.3) .

(iii)Apply the inverse transformation function , d=G^-1(S) to the level got in step(i).

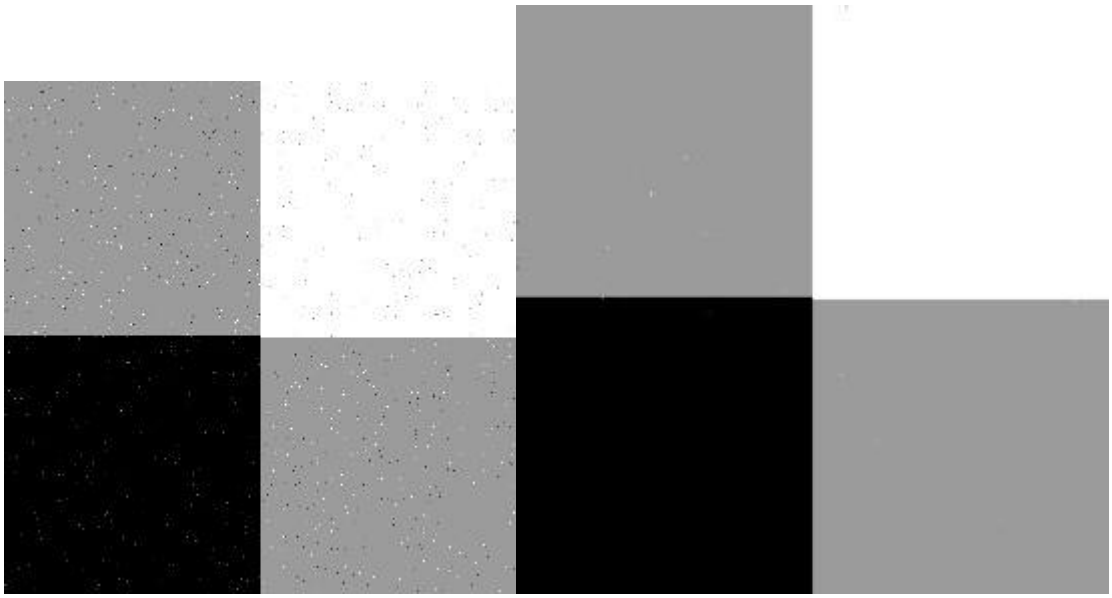
This procedure gives a processed value of original image with the new gray levels characterized by the specified density P_d (d).

IMAGE SMOOTHING:

Smoothing may be distinguished from the related and partially overlapping concept of curve fitting in the following ways. Curve fitting often involves the use of an explicit function form for the result, whereas the immediate results from smoothing are the "smoothed" values with no later use made of a functional form if there is one. The aim of smoothing is to give a general idea of relatively slow changes of value with little attention paid to the close matching of data values, while curve fitting concentrates on achieving as close a match as possible. Smoothing methods often have an associated tuning parameter which is used to control the extent of smoothing.

MEDIAN FILTERING:

Median filtering is needed to remove impulse noise from an image. Pixel intensity is replaced with the median of pixel intensities within a window centered at that pixel. If a part of the window falls outside the image, intensities within the portion of the window inside the image is used. Circular windows are used to make smoothing independent of the image orientation. An example of median filtering b y this software is given below.

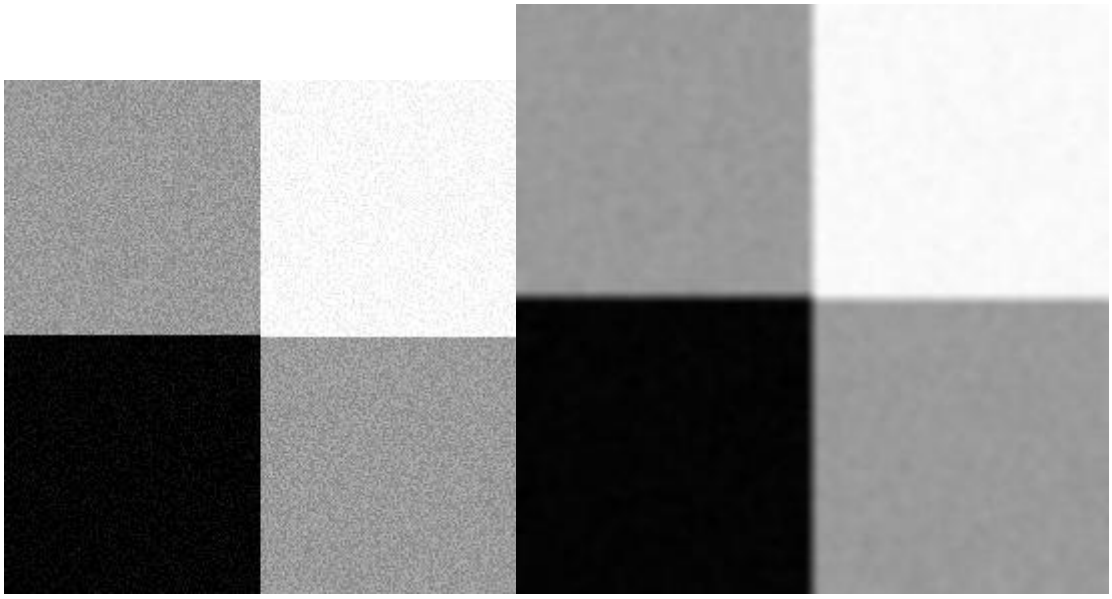


(a) (b)

Fig. 1. (a) An image containing impulse noise.
(b) Smoothing the image by median filtering using a filter of radius 1 pixel.

MEAN FILTERING:

Mean filtering is image averaging and the value at a pixel in the output is equal to the average of values within a circular window centered at the pixel in the input. The window size determines the neighborhood size where averaging is performed. As the window size is increased, more noise is removed, but that at the same time blurs the image more. An example of mean filtering by this method is given below.



(a) (b)

Fig. 2. (a) An image containing zero-mean noise.

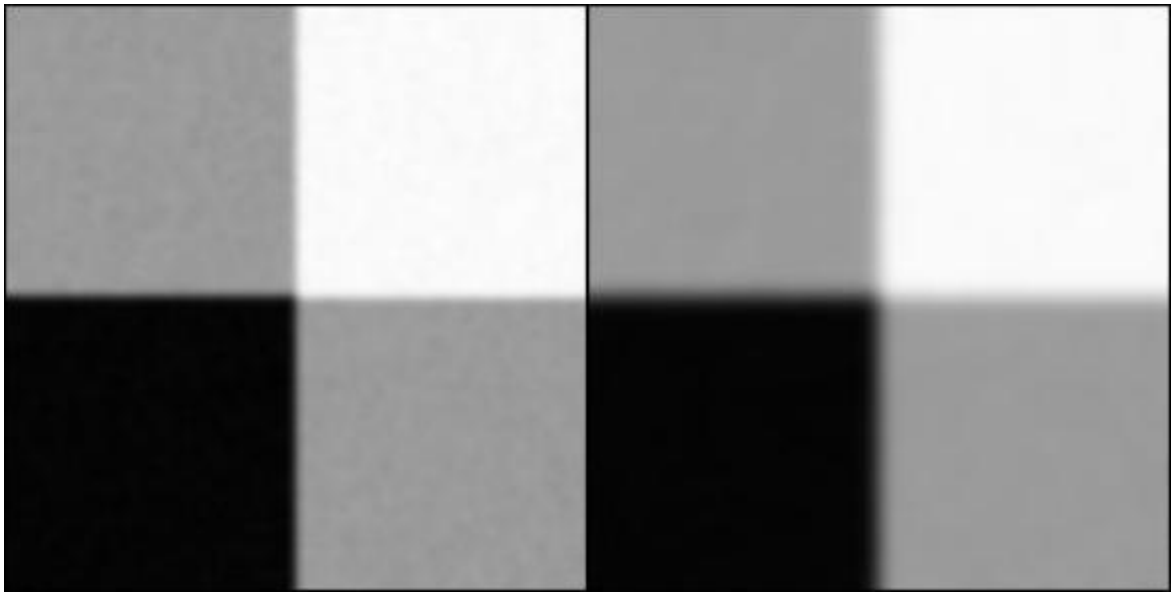
(b) Image obtained by mean filtering using a filter of radius 4 pixels.

GAUSSIAN FILTERING:

A Gaussian is an ideal filter in the sense that it reduces the magnitude of high spatial frequencies in an image proportional to their frequencies. That is, it reduces magnitude of higher frequencies more. This will be at the cost of more computation time when compared to mean filtering.

A Gaussian extends to infinity in all directions, but because it approaches zero exponentially, it can be truncated three or four standard deviation away from its center without affecting the result noticeably. Speed up is achieved by splitting a 2-D Gaussian into two 1-D Gaussians, G(x,y) = G(x) G(y), and carrying out filtering in 1-D, first row by row and then column by column.

An example of Gaussian filtering is given below.



(a) (b)

Fig. 3. (a) Image that containing noise.
(b) Smoothing the image shown in Fig. 2a by Gaussian filters of standard deviations 2 and 4 pixels, respectively.

CONSERVATIVE SMOOTHING:

Brief Description

Conservative smoothing is a noise reduction technique that derives its name from the fact that it employs a simple, fast filtering algorithm that sacrifices noise suppression power in order to preserve the high spatial frequency detail (e.g. sharp edges) in an image. It is explicitly designed to remove noise spikes --- i.e. isolated pixels of exceptionally low or high pixel intensity (e.g. salt and pepper noise) and is, therefore, less effective at removing additive noise (e.g. Gaussian noise) from an image.

How It Works?

Like most noise filters, conservative smoothing operates on the assumption that noise has a high spatial frequency and, therefore, can be attenuated by a local operation which makes each pixel's intensity roughly consistent with those of its nearest neighbors.

However, whereas mean filtering accomplishes this by averaging local intensities and median filtering by a non-linear rank selection technique, conservative smoothing simply ensures that each pixel's intensity is bounded within the range of intensities defined by its neighbors.

This is accomplished by a procedure which first finds the minimum and maximum intensity values of all the pixels within a windowed region around the pixel in question. If the intensity of the central pixel lies within the intensity range spread of its neighbors, it is passed on to the output image unchanged.

However, if the central pixel intensity is greater than the maximum value, it is set equal to the maximum value; if the central pixel intensity is less than the minimum value, it is set equal to the minimum value. Figure 1 illustrates this idea.

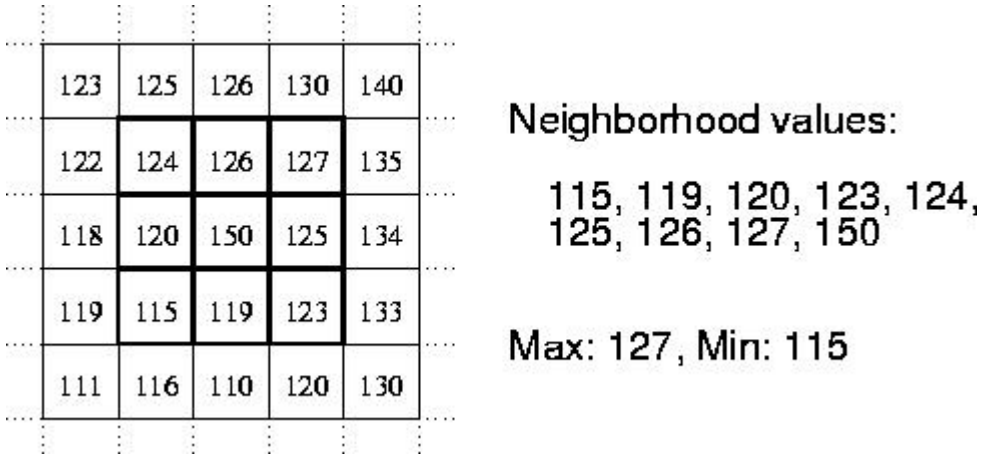


Figure 1 Conservatively smoothing a local pixel neighborhood. The central pixel of this figure contains an intensity spike (intensity value 150). In this case, conservative smoothing replaces it with the maximum intensity value (127) selected amongst those of its 8 nearest neighbors.

If we compare the result of conservative smoothing on the image segment of Figure 1 with the result obtained by mean filtering and median filtering, we see that it produces a more subtle effect than both the former (whose central pixel value would become 125) and the latter (124). Furthermore, conservative smoothing is less corrupting at image edges than either of these noise suppression filters.

IMAGE SHARPENING:

Sharpening is one of the most impressive transformations you can apply to an image since it seems to bring out image detail that was not there before. What it actually does, however, is to emphasize edges in the image and make them easier for the eye to pick out while the visual effect is to make the image seem sharper, no new details are actually created. Paradoxically, the first step in sharpening an image is to blur it slightly. Next, the original image and the blurred version are compared one pixel at a time.

If a pixel is brighter than the blurred version it is lightened further; if a pixel is darker than the blurred version, it is darkened. The result is to increase the contrast between each pixel and its neighbors.

The nature of the sharpening is influenced by the blurring radius used and the extent to which the differences between each pixel and its neighbor are exaggerated.

SHARPENING METHODS:

Picture Window offers three sharpening techniques: Sharpen, Heavy Sharpen, and Un sharp Mask. Sharpen and Heavy Sharpen are the most straightforward methods.

They are computed similarly, but Sharpen blurs the pixels in the image based on only its four closest neighbors and exaggerates the differences moderately while heavy Sharpen blurs by using the eight nearest neighbors and exaggerates the differences more. In either case, you can moderate the effect by setting the Amount slider back from its 100% maximum value.

The effects of Sharpen and Heavy Sharpen are illustrated below:

Sharpen Heavy Sharpen

OVERSHARPENING:

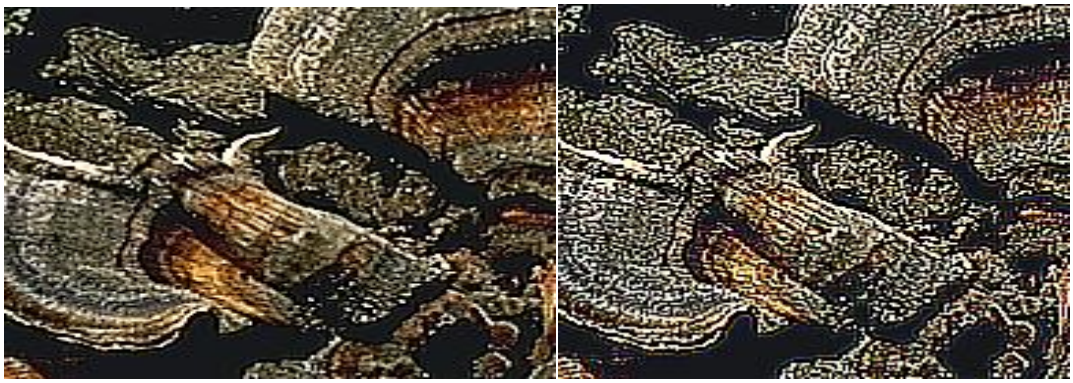
If you continue to sharpen an image, several things happen:

1. Edges become unnaturally pronounced—dark objects are outlined with light halos and light objects are outlined with dark halos.

2. Normally invisible noise in the image is amplified and starts to show up as texture in areas that looked smooth in the original images. This can create an undesirable graininess in parts of photographs that should be smooth like clouds and skies.

3. Extreme sharpening causes the image to break up as each individual pixel stands out more and more from its neighbors.

Thus as you progressively sharpen a slightly blurry image, it first starts to look better, but then as you begin to over sharpen it, it starts to look worse again. Once you know what over sharpening looks like, you will be able to spot it on the screen and back off a little.



Heavy Sharpen Twice Heavy Sharpen Three Times

USING THE SHARPEN TRANSFORMATION:

The best way to set up Picture Window's Sharpen Transformation is to zoom the input image in to a magnification factor of 1:1 or even 2:1 and the zoom the Preview window in to the same magnification factor and scroll the two windows as necessary to display the same part of the image.

Resize and reposition the two windows so they are next to each other across the top of the screen. This lets you see every detail at the individual pixel level in both the original image and the sharpened Version.

As you preview the results of various settings, you will then be able to compare the original and sharpened versions easily. If necessary, you can scroll both windows to see the effect of the transformation on other parts of the image.

For slightly blurred images, the Sharpen or Heavy Sharpen options often work well.

If the effect is too strong, you can always move the Amount slider to the left to reduce the amount of sharpening. If you want to sharpen just part of an image, leaving the rest alone, you will need to create a mask that isolates the region of interest and then select this mask into the Amount control. Making and using masks is covered in the Picture Window manual and help file.

UNSHARP MASKING:

Unsharp masking is the most powerful sharpening method Picture Window supports, however it is a little more complicated to use. When you select Unsharp Masking, the Sharpen dialog box expands to add two additional sliders for Radius and Threshold.

The Radius slider lets you control the amount of blurring. Generally you should set the radius to correspond to the degree to which the original image is blurred. The blurrier the image, the higher the radius you need to select. Choosing too large a radius creates a sort of ghosting effect around the edges of objects; if the radius is too small, the sharpening effect is minimized.

The Threshold setting lets you restrict to sharpening action to only those pixels whose difference from their neighbors exceeds a specified threshold value.

The idea behind setting the threshold value is to select a value that still brings out edge detail without creating unwanted texture in smooth areas like clouds or clear blue skies. In the image detail below, you can see how Unsharp Mask with a threshold of zero sharpens the tree silhouette, but also brings out the film grain and scanning noise in the sky area.

Increasing the threshold to 20 leaves the sky mostly untouched but still makes the tree stand out against its background.

Unsharp Mask
Threshold 0

Unsharp Mask
Threshold 20

Note: this example has been deliberately over sharpened to emphasize the effects of the threshold setting.

IMAGE RESTORATION: (NOV 2012)

Image restoration is similar to the process of image enhancement. Image restoration is the process that reconstructs an image that has been degraded by using some prior knowledge of degradation phenomenon.

Restoration is the process that restores the image. The restoration techniques use modeling the degradation and applying the inverse process to recover the original image.

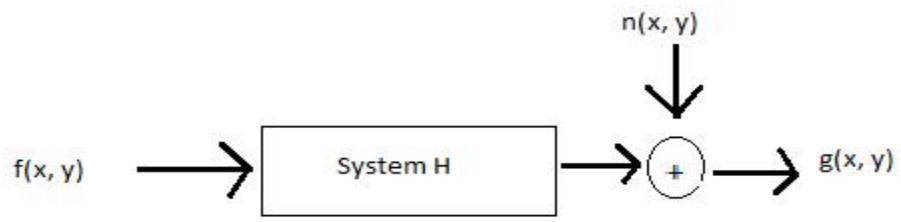
Digital images can be degraded in many ways,

For example,

i) In analog to digital converter (ADC), linear noise may arise.
ii) Computer can cause random point degradations.
iii) Interference from radio source, electric motor of such frequency will produce lines at regular intervals on the screen. If the object moves, as it is being exposed, the image will be blurred.

The algorithms for detecting and removing degradation in the spatial domain are convolution, modeling, matrix and re-sampling techniques. In frequency domain, Fourier reconstruction algorithm is used.

MODEL OF IMAGE DEGRADATION: (APR 2012) (APR 2013)



The input-image f(x, y) is applied to the system 'H'. The additive noise n(x, y) is also added to the system output to produce a degraded image g(x, y).

The input-output relationship is expressed as

$$g(x, y) = H[f(x, y)] + n(x, y) \quad \rightarrow (1)$$

Properties of degradation model:

i) Linearity:

Assume n(x, y) = 0 in eqn 1

$$g(x, y) = H[f(x, y)] \quad \rightarrow (2)$$

H is linear if

$$H[k_1 f_1(x, y) + k_2 f_2(x, y)] = k_1 H[f_1(x, y)] + k_2 H[f_2(x, y)] \quad \rightarrow (3)$$

Where k_1 and k_2 are constants, and $f_1(x, y)$ and $f_2(x, y)$ are any two input images. The linear operator posses both additive and homogeneity property.

ii) **Additive:**

If $k_1 = k_{2=1}$ in eqn 3. It is called additive property.

$$H[f_1(x, y) + f_2(x, y)] = H[f_1(x, y)] + H[f_2(x, y)] \quad \rightarrow (4)$$

It states that, if H is a linear operator, the response to a sum of two inputs is equal to the sum of the two responses.

iii) Homogeneity:

It states that the response to a constant multiple of any input is equal to the response to that input multiplied by the same constant.

i.e., $f_2(x, y) = 0$ in eqn (3)

$$H[k_1 f_1(x, y)] = k_1 H[f_1(x, y)] \quad \rightarrow (5)$$

Which is called the homogeneity property

iv) Position invariant (or) Space invariant:

An operation having the input-output relationship g(x,y) = H[f(x, y)] is said to be "position invariant" if

Sub x= x-α, y= y-β

H[f(x-α, y-β)] = g(x-α, y-β) → **(6)**

For any f(x, y) and any α and β.

Position invariant indicates that, the response at any point in the image depends only on the value of the input at that point and not on the position of the point.

DEGRADATION MODEL FOR CONTINUOUS FUNCTION:

When the image f(x, y) is convolved with the two dimensional impulse function δ (x, y), then it can be represented by

f(x, y) = $\int_{-\infty}^{\infty} f(x,y)\ \delta(x - x\square, y - y\square)dxdy$ → **(7)**

We use the dummy variable α and β and the above equation can be written as

f(x, y) = $\int_{-\infty}^{\infty} f(\alpha,\beta)\ \delta(x - \alpha, y - \beta)d\alpha d\beta$ →**(8)**

sub equation (8) in (1)

g(x, y) = H[f(x, y)] +n(x, y)

f(x, y) = H $[\int_{-\infty}^{\infty} f(\alpha,\beta)\ \delta(x - \alpha, y - \beta)d\alpha d\beta]$ **+ n(x,y)** →**(9)**

If n(x, y) = 0 and if H is a linear operator & we extend the additively property to the integrals, then

g(x, y) = $\int_{-\infty}^{\infty} H[f(\alpha,\beta)\ \delta(x - \alpha, y - \beta)]d\alpha d\beta$ →**(10)**

Since f(α, β) is independent of x and y and from the homogeneity property,

g(x, y) = $\int_{-\infty}^{\infty} f(\alpha,\beta)H[\ \delta(x - \alpha, y - \beta)]d\alpha d\beta$ →**(11)**

We know the "impulse response" H can be represented as

h(x, α, y, β) = H[δ (x-α, y-β)] →**(12)**

Then g(x, y) can be represented by

g(x, y) = $\int_{-\infty}^{\infty} f(\alpha,\beta)h(x,\alpha,y,$ **β) d**α**dβ** →**(13)**

This equation is called as "superposition (or Freehold) integral of the first kind".

If H is a position invariant, then,

$$H[\delta(x-\alpha, y-\beta)] = h(x-\alpha, y-\beta) \quad \rightarrow (14)$$

Then equation (13) becomes,

$$g(x,y) = \int_{-\infty} \int^{\infty} f(\alpha,\beta) h(x-\alpha, y-\beta)\, d\alpha d\beta \quad \rightarrow (15)$$

In the presence of additive noise, the expression describing a linear degradation model becomes

$$g(x,y) = \int_{-\infty} \int^{\infty} f(\alpha,\beta) h(x,\alpha,y,\beta)\, d\alpha d\beta + n(x,y) \quad \rightarrow (16)$$

This is the equation for degradation model for continuous image function f(x, y).

DISCRETE DEGRADATION MODEL:

Digital images are used in most computer based applications and therefore it is needed to formulate discrete degradation model.

Let two functions f(x) and h(x) are sampled uniformly to form arrays of dimensions A and B respectively.

The discrete variable ranges 1, 1, 2…. A-1 for f(x) and 0, 1, 2… B-1 for h(x).

To avoid overlapping in the individual periods, a common period "M ≥ A + B- 1" is selected and in the individual functions zeros are appended after the interval A and B in the functions f(x) and h(x) and they are represented as $f_e(x)$ and $h_e(x)$ i.e., extended functions.

The convolution of $f_e(x)$ and $h_e(x)$ is given by

$$g_e(x) = \sum_{m=0}^{M-1} fe(m)\, h_e(x-m) \quad \rightarrow (17)$$

for x = 0, 1, 2,….. M-1

Both $f_e(x)$ and $h_e(x)$ are assumed to have a period equal to M. $g_e(x)$ also has this period. Then, the above equation is expressed in the matrix notation as

$$g = Hf \quad \rightarrow (18)$$

where f and g are M dimensional column vectors.

$$f = \begin{pmatrix} f_e(0) \\ f_e(1) \\ \\ f_e(M-1) \end{pmatrix} \quad \rightarrow (19)$$

and

$$\begin{pmatrix} g_e(0) \\ \\ \\ \end{pmatrix} \quad \rightarrow (20)$$

$$g = \begin{matrix} g_e(1) \\ \dots \\ g_e(M-1) \end{matrix}$$

and H is the M x M matrix

$$H = \begin{pmatrix} h_e(0) & h_e(-1) & h_e(-2) & \dots & h_e(-M+1) \\ h_e(1) & h_e(0) & h_e(-1) & \dots & h_e(-M+2) \\ h_e(2) & h_e(1) & h_e(0) & \dots & h_e(-M+3) \\ \dots & \dots & \dots & \dots & \dots \\ h_e(M-1) & h_e(M-2) & h_e(M-3) & \dots & h_e(0) \end{pmatrix} \rightarrow (21)$$

Assume the function $h_e(x)$ has the priority property, then $h_e(x) = h_e(M + x)$. Then the matrix "H" is written as

$$H = \begin{pmatrix} h_e(0) & h_e(M-1) & h_e(M-2) & \dots & h_e(1) \\ h_e(1) & h_e(0) & h_e(M-1) & \dots & h_e(2) \\ \dots & \dots & \dots & \dots & \dots \\ h_e(M-1) & h_e(M-2) & h_e(M-3) & \dots & h_e(0) \end{pmatrix} \rightarrow (22)$$

In this equation, the rows are related by a circular shift to the right. i.e., the i^{th} row elements can be obtained from the $i\text{-}1^{th}$ row by shifting its elements circularly right by one position.

A square matrix in which each row is a circular shift of the preceding row and the first row is a circular shift of the last row. It is called as **"circulator matrix".**

For example,

Assume A= 3 and B= 2

$M \geq A + B - 1$

$M \geq 3 + 2 - 1 = 4$ and then append one zero to the samples of f(x) and two zeros to samples of h(x)

In this, f and g are 4D vectors and H is 4 x 4 matrix.

$$H = \begin{pmatrix} h_e(0) & h_e(3) & h_e(2) & h_e(1) \\ h_e(1) & h_e(0) & h_e(3) & h_e(2) \\ h_e(2) & h_e(1) & h_e(0) & h_e(3) \\ h_e(3) & h_e(2) & h_e(1) & h_e(0) \end{pmatrix}$$

However, as $h_e(x) = 0$ for x=2, 3 and $h_e(x) = h(x)$ for x =0, 1

$$H = \begin{pmatrix} h_e(0) & & & h_e(1) \\ h_e(1) & h_e(0) & & \\ & h_e(1) & h_e(0) & \\ & h_e(1) & h_e(0) & \end{pmatrix}$$

where, all the elements not indicated in the matrix are zero.

The extension of the concepts for 2-D discrete degradation model is straight forward.

For two digitized images f(x, y) and h(x, y) of sizes A x B and C x D respectively, extended images of size M x N may be formed by padding with zeros and they may be given by

$$f_e(x, y) = \begin{cases} f(x, y) & 0 \le x \le A-1 \text{ and } 0 \le y \le B-1 \\ 0 & A0 \le x \le M-1 \text{ and } B \le y \le N-1 \end{cases} \rightarrow (23)$$

And

$$h_e(x, y) = \begin{cases} f(x, y) & 0 \le x \le C-1 \text{ and } 0 \le y \le D-1 \\ 0 & C \le x \le M-1 \text{ and } D \le y \le N-1 \end{cases} \rightarrow (24)$$

for x =0 to M – 1

y = 0 to N – 1

The complete degradation model is obtained by adding the noise term $n_e(x, y)$. It is given as

$$g(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f_e(m - n)\, h_e(x-m, y-n) + n_e(x, y)$$

for x = 0, 1 ,2 ,3 …M-1 &

y = 0, 1 ,2 ,3 …M-1

Let f, g & n represent MN-dimensional column vectors formed by the rows of M x N functions $h_e(x, y)$, $g_e(x, y)$ and $n_e(x, y)$.

The elements in the first row of $f_e(x, y)$, the next N elements are from the second row and so on for all M rows of $f_e(x, y)$.

This allows the equation, **g = Hf + n** →**(25)**

Where f, g and n are of dimension of MN x 1 and H is of dimension MN x MN. This matrix consists of M^2 partitions, each partition being of size M x N and ordered according to

$$H = \begin{pmatrix} H_0 & H_{M-2} & H_{M-2} & \dots & H_1 \\ H_1 & H_0 & H_{M-1} & \dots & H_2 \\ H_2 & H_1 & H_0 & \dots & H_3 \\ \dots & \dots & \dots & \dots & \dots \\ H_{M-1} & H_{M-2} & H_{M-3} & \dots & H_0 \end{pmatrix} \rightarrow (26)$$

Each partition h_j is constructed from the j^{th} row of the extended function $h_e(x, y)$

As follows,

$$H = \begin{pmatrix} h_e(j, 0) & h_e(j, N-1) & \dots & h_e(j, 1) \\ h_e(j, 1) & h_e(j, 0) & \dots & h_e(j, 2) \\ \dots & \dots & \dots & \dots \\ h_e(j, N-1) & h_e(j, N-2) & \dots & h_e(j, 0) \end{pmatrix} \rightarrow (27)$$

IMAGE NOISE MODELS: (NOV 2013) (APR 2014)

Error or uncertainty of an image arising from such sources as sensor noise, film grain irregularities, and atmospheric light fluctuations. These all such effects are called as **"noise".**

Noise Modeling :

During image transmission, the noise can be occurs in the digital images. The performance of image is affected by various factors such as environmental conditions and quality of the sensing elements themselves.

For example, an image transmitted through a wireless network may be corrupted as a result of lightning or other atmospheric disturbances.

Spatial and Frequency Properties of Noise:

Frequency properties refer to the frequency content of noise in the Fourier sense (i.e. as opposed to frequencies of electromagnetic spectrum). For example, when the Fourier Spectrum of noise is constant, the noise is called **"white noise".**

Spatial properties refer to the noise is independent of spatial coordinates and it is uncorrelated with respect to the image itself (i.e. there is no correlation between pixel values and the values of noise components)

Some important Noise Probability Density Function:

i) Gaussian Noise:

The PDF of a Gaussian random variable 'z' , is given by

$$P(z) = \frac{1}{\sqrt{2\pi\sigma}} e^{-(Z - \bar{Z})^2/2\sigma^2}$$

where, Z=Intensity

$\bar{Z}$= Mean (average) value of Z

σ = Its standard deviation

σ^2 = Variance of Z

ii) Rayleigh Noise:

The PDF of a Rayleigh noise is given by

$$P(z) = \begin{cases} \frac{2}{b}(z-a)e^{-(z-a)^2/b} & \text{for } z \geq a \\ 0 & \text{for } z < a \end{cases}$$

The mean and variance of this density are given by

$$\overline{z} = a + \sqrt{\frac{\pi b}{4}}$$

And $\sigma^2 = \frac{b(4-\pi)}{4}$

iii) Erlang (Gamma) Noise:

The PDF of Erlang noise is given by

$$P(z) = \begin{cases} a^b z^{b-1} / (b-1)!\, e^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$

where, the parameters are such that $a > 0$, b is a positive integer. The mean and variance are given by

$$\overline{z} = \frac{b}{a}$$

and $\sigma^2 = b/a^2$

iv) Exponential Noise:

The PDF of exponential noise is given by

$$P(z) = \begin{cases} a e^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}$$

where, $a > 0$

The mean and variance are

$$\overline{z} = \frac{1}{a} \quad \text{and} \quad \sigma^2 = 1/a^2$$

v) Uniform Noise:

The PDF of uniform noise if given by

$$P(z) = \begin{cases} \frac{1}{b-a} & \text{if } a \le z \le b \\ 0 & \text{otherwise} \end{cases}$$

The mean and variance are

$$\bar{z} = \frac{1}{b-a}$$

and $\sigma^2 = (b-a)^2/12$

vi) Impulse (salt and pepper) Noise:

The PDF of impulse noise is given by

$$P(z) = \begin{cases} p_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$

If b> a, intensity 'b' will appear as a light dot in the image otherwise 'a' will appear like a dark dot. If either p_a or p_b is zero, the impulse noise is called **'unipolar'**. It nether probability is zero and if they are approximately equal, impulse noise values will resemble salt and pepper granules randomly distributed over the image. For this reason, bipolar noise is called salt and pepper noise.

Periodic Noise:

Periodic noise in an image arises from electrical or electromechanical interference during image acquisition. This noise is the spatially dependent noise. Periodic noise can be reduced significantly through frequency domain filtering. For example if the image is severely computed by (spatial) sinusoidal noise of various frequencies. The Fourier transform of a pure sinusoidal noise of various frequencies. The Fourier transform of a pure sinusoid is a pair of conjugate impulses located at the conjugate frequencies of the sine wave.

$\sin(2\pi u_0 x + 2\pi v_0 x) \Leftrightarrow j\ 1/2[\delta(u+Mu_0, V+ NV_0) - \delta(u-Mu_0, V-NV_0)]$

Thus if the amplitude of a sine wave in the spatial domain is strong, we would expect the spectrum of the image is a pair of impulses for each sine wave in the image.

PHOTO-DETECTOR NOISE :

Types of noises:

1. Photo-detector noise
2. Film grain noise

We can see these noises one by one.

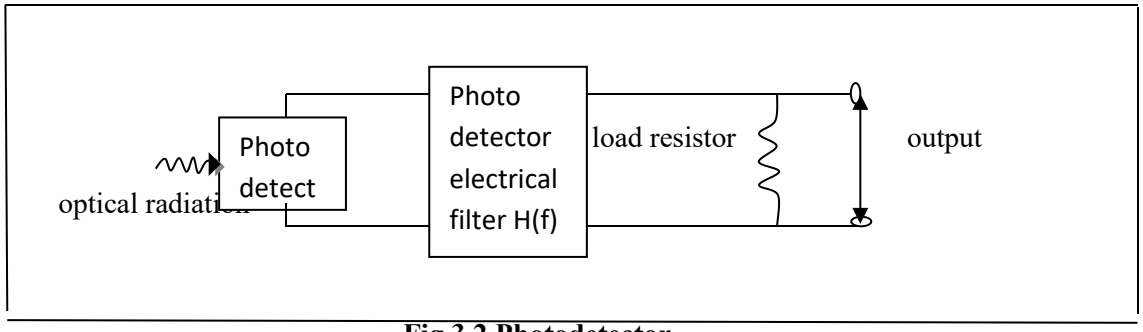


Fig.3.2.Photodetector

Fig.3.2. shows the photo-detector that contains photo-detector imaging sensor and an associated electrical filter that will serve as a basis for the discussion of sensor noise sources.

Ideally, the photo-detector acts as a converter of optical intensity to detector current or equivalently, as a converter of incident photons to electrons emitted by the detector.

The detector produces a signal current that passes through an electrical filter and creates a signal voltage across a load.

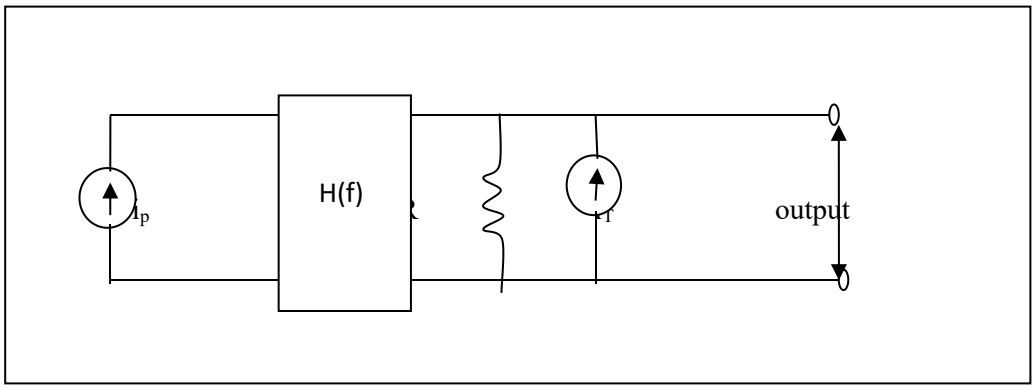


Fig.3.3.Photo-detector signal and thermal noise model

Thermal noise is the most common noise source associated with electronic imaging systems, resulting from random electron fluctuation in resistive elements of photo-detectors or resistors with sensor amplifies.

Thermal noise can be modeled as an additive Gaussian stochastic process independent of the sensed image field.

As an illustration of thermal noise, calculations consider the photo-detector model of fig. in which the load resister is assumed to be the only significant resistive element in the unit.

In the figure the photo-detector is replaced by an ideal current generator whose current is linearly proportional to the incident light intensity at some image point.

The thermal noise contribute can be modeled as a thermal noise current source of value i_T in parallel with the noise resister.

The thermal noise current represents a zero mean Gaussian random process with variance $\sigma^2_{iT} = N_T / R$

where N_T is the thermal noise power at the system output

$N_T = 2kT \int_{-\infty}^{\infty} |H_E(f)|^2 df$

Where k is the Boltzmann's constant

$k = 1.38 \times 10^{-23}$ Joules / degree Kelvin

T is the temperature in degrees Kelvin and $H_E(f)$ is the equivalent transfer function of the photo-detector electrical filter and load resister.

By simply , the filter is a capacitor of value C placed in parallel with the detector and the load resistor, the thermal noise power becomes,

$N_T = kT/RC$

If the detector current is constant and the equivalent filter is linear, the probability density of the current output of the detector can be written as

$p(i) = (2 \pi \sigma^2_{iT})^{-1/2} \exp[-(i-i_s)^2 / 2 \sigma^2_{iT}]$

Where i_s is the equivalent average signal current at the output produced by the photo-detector current i_p passing through the filter.

Photo-detector current is not truly constant valued even when the incident light intensity is constant. Photoelectric sensors produces uncertainty resulting from the quantum mechanical nature of light.

At low light levels, the number of electrons emitted by a photo-detector is governed by a poison probability density. In an observation time period "T " , the probability distribution of detector current pulses is given by

$P_R [i_P = j_q / T] = (u_S + u_H)^j \exp\{-(u_S + u_H)\} / j!$

Where j is a positive integer

q is the charge of an electron

$q = 1.6 \times 10^{19}$ coulombs

u_S is the average number of electrons emitted front the detector as a result of the incident illumination and u_H is the average number of electrons emission caused by detector dark current and extraneous background radiation.

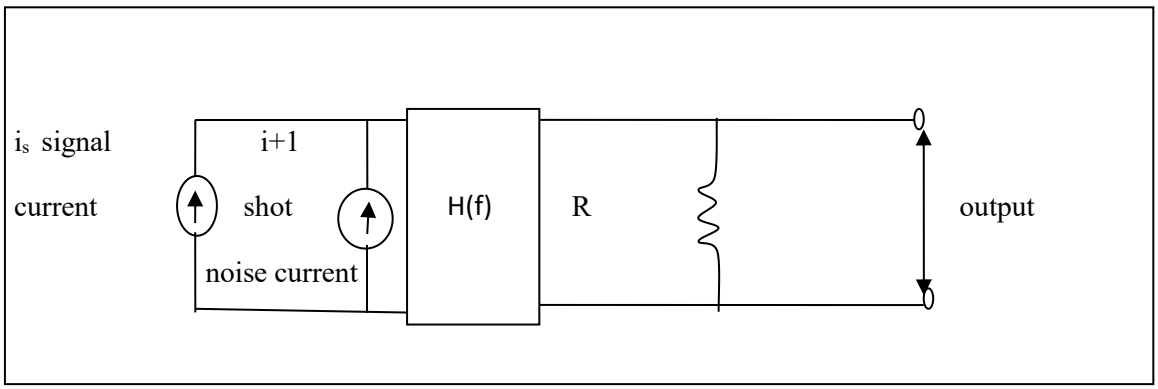


Photo-detector signal and short noise model

The resultant uncertainty in knowledge of the detector current is called short Boise in photo-emissive detectors and generation –recombination noise in photo-conductive and photovoltaic detectors.

If the photo-detector possesses sufficient internal amplification, for example, a photo-multiple tube, the shot noise will normally dominate any subsequent thermal noise sources otherwise considered constant.

In either case, if the average number of signal electron emissions become large, the poison distribution can be approximated by the Gaussian distribution.

$p(i_p)=[2\pi q^2 u_{S/} T^2]^{-1/2} \exp\{-i_p - q u_S / T)/2q u_S / T \}^2$For most imaging sensors, the associated sensor can be modeled either as a Gaussian distributed or Poisson distribution random process.

Poisson distribution noise only occurs when the image light level is extremely small and the photo-detector possesses a large internal electron amplification.

Film Grain Noise :

Silver halide grain are used in the exposure and development of a photographic film, that are exposed to a sufficient quantity of light are converted to grains of metallic silver.

This process is not entirely deterministic, silver halide grains experiencing an equivalent exposure are not necessarily converted to the same size and shape silver grains, furthermore silver grains are randomly distributed over the surface of the film.

This inherent randomness is silver grain formation is called "film – grain noise", leads to a randomness or uncertainty in the amount of light passing through a transparency or reflected from a print.

The classical model for film grain noise fluctuations assumes a Gaussian distribution of the exposed film density D(x, y)

$$P[D(x, y)] = [\, 2\pi\sigma^2_D(x, y)]^{-1/2} \exp \frac{-[D(x,y) - u\,(x,y)]2}{2\sigma 2\ (x,y)}$$

About a mean density $u_D(x, y)$ with a variance $\sigma^2(x, y)$

The mean density is determined by a spatial average about the point (x, y) over some window including many individual grains and the standard deviation is modeled as

$$\sigma_D(x, y) = \alpha[u_D\ (x, y)]^{\beta}$$

where β is a constant

α is empirically set at

$$\alpha = 0.66 \left(\frac{a}{A}\right)^{1/2}$$

where 'a' is the average film grain area and 'A' is the examination area of the film Falconer has suggested $\beta = 1/2$ while Huang has setting $\beta = 1/3$.

The density D(x , y) described by equation can be expressed as the sum of a mean density components $u_D(x, y)$ and a zero mean Gaussian random process component N(x, y) according to the relation.

$D(x,y) = u_D(x,y) + \sigma_D(x,y)N(x,y)$

or $D(x,y) = u_D(x,y) + \alpha [\sigma_D(x,y)]^{\beta} N(x,y)$

The film grain noise is signal dependent, it is additive in the density domain. The transmittance of the exposed film at a fixed wavelength may be written as

$$\tau(x,y) = 10^{-D(x,y)\,D(\lambda)}$$

Where $D(\lambda)$ is the characteristic density of the film as a function of wavelength.

Then it is easily seen that the transmittance function may be expressed as the product of the average transmittance without grain noise $\tau_0(x,y)$ and a noise factor proportional to $\sigma_D(x,y)$.

$$\tau(x,y) = \tau_0(x,y)\, 10^{-\sigma\, D(x,y)\, N(x,y)\, D(\lambda)}$$

Thus in the intensity domain, film grain noise is a multiplicative noise process.

SPATIAL FILTERING : (APR 2012) (APR 2013)

The use of a spatial mark for image processing is called spatial filtering.

Two types of spatial filtering

i) Linear filters,
ii) Non Linear filters.

i) LINEAR SPATIAL FILTER:

Linear filter means that the transfer function and the impulse or point spread function of a linear system are inverse Fourier transforms of each other.

Three types of linear filters are,

a) Lowpass filters,

b) Highpass filters, and

c) Bandpass filter.

a) LOWPASS FILTER :

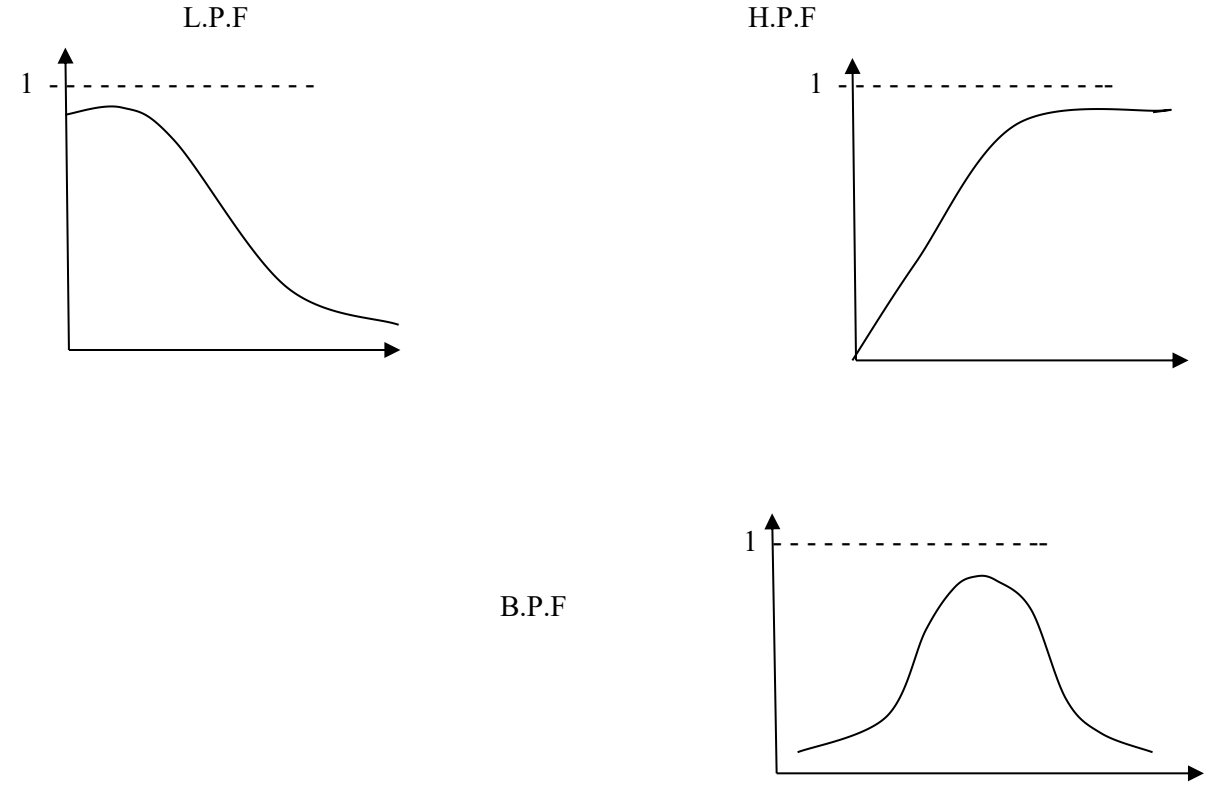
It attenuate or eliminate high frequency components in the Fourier domain. High frequency components characterize edges and others sharp details in an images, so the low pass filter causes image blurring.

b) HIGHPASS FILTERS:

It attenuate or eliminate low frequency components. These components are responsible for slowly varying characteristics of an image, such as overall contrast and average intensity.

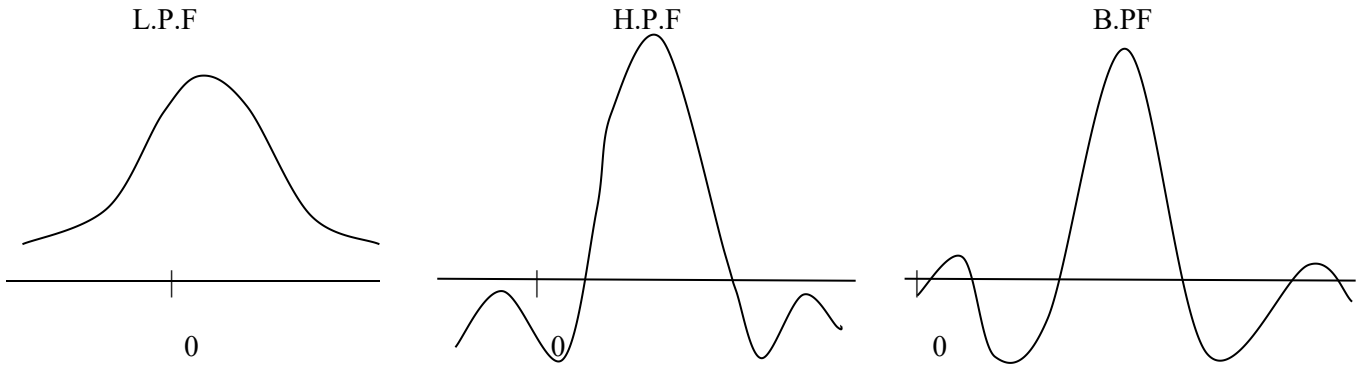
c) BANDPASS FILTER :

It removes selected frequency regions between low and high frequencies. It is used for image restoration.



(A) Frequency Domain Filters

Fig.(a) represents the cross sections of circularly symmetric low pass, high pass and band pass filters in the frequency domain and their corresponding spatial filters are shown below in **fig**(b).



(a) Spatial Domain Filters

The linear mask is the sum of the products between the mask coefficients and the intensities of the pixels under the mask at a specific location in the image. It is denoted by the gray levels of pixels under the mask at any location by $z_1, z_2, \ldots..z_9$. The response of a linear mask is $R=w_1z_1, w_2z_1,\ldots. w_9z_9$.

3x3 mask coefficient is

w_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

ii) NONLINEAR SPATIAL FILTER:

This filter is based directly on the values of the pixels in the neighborhood and they did not use coefficients.Using nonlinear filter noise reduction can be achieved effectively.

The basic function of the nonlinear filter is to compute the median gray level value in the neighborhood in which the filter is located.

Other examples of nonlinear filters are max and min filter.

- Max filter is used to find the brightest points in an image.
- Min filter is the opposite of max filter.

SMOOTHING FILTERS:

Smoothing filters are used for blurring and noise reduction. Blurring is used in preprocessing steps, such as removal of small details from large image and bridging of small gaps in lines and curves.

Noise reduction can be achieved by blurring with a linear filter and by nonlinear filter.

Types of Smoothing filters:

Lowpass Spatial Filtering

It shows that, the LP spatial filter have all positive coefficients.

From this equation,

$R=w_1z_1, w_2z_1,\ldots. w_9z_9$.

The response will be the sum of gray levels of nine pixels which could cause R to be out of the valid gray level range.

Therefore the solution is to scale the sum by dividing R by 9.

Generally, the response R would simply be the average of all the pixels in the area of mask. Spatial lowpass filters of various sizes are shown in fig.

1	1	1
1	1	1

1	1	1

1/9 x

(a)

1/25 x

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

(b)

It is referred to as neighborhood averaging.

MEDIAN FILTERING :

If Q=0, this filter is reduced to the arithmetic mean filter. If Q= -1, this filter is reduced to harmonic filter. In this filter, the value of the pixel is replaced by the intensity level of the neighborhood of that pixel.

The restored function is given as,

F(x,y) = Median[g(u,v)]

Median filters are used to reduce certain types of random noise excellently. These are effective in the presence of unipolar and bipolar impulse noises. The median operator is a nonlinear filter. It has the following properties.

i) The median filter reduces the variance of the intensities in the image.

ii) Median filters can change the image intensity mean value, if the spatial noise distribution in the image is not symmetrical within the window.

iii) Median filters preserve certain edge shapes.

iv) Given a symmetrical window shape the median filter preserves the location of edges.

v) After applying a median filter, new gray values are not generated.

Computation of median filter requires sorting of a list of numbers. For large window sizes, computation time is large. One approach for neighboring window locations is based on modification of the sorted list at the previous location (x,y). As we move the "n x m" window, we discard "n" points and add 'n' new points and leave the remaining "mn-2n" points unchanged.

This approach is efficient only, if the amount of common image intensity data is significantly large from one position to the next. The second approach which is based on list sorting is to calculate the median iteratively, by bit position, starting with MSB. After determining the median bit for a bit position. It is identified that, only intensity values with this bit are retained for subsequent calculations.

This procedure terminates for 'n' bit data after n iterations, regardless of the size of the window.

Steps:

i) Input the values in decimal numbers.

ii) Convert them into binary form.

iii) Find the median value of all the MSBs.

iv) Find the median value of the next LSB.

v) Continue the procedure of finding the median value of the next LSB bit, till all the bits are exhausted.

vi) Write the median bits, starting from that of MSB to LSB. This is the overall median value.

DIRECTIONAL SMOOTHING

A directional averaging filter is used to protect the edges from blurring while smoothing.

Spatial averages V(m, n: ɵ) are calculated in various directions. Its equation is given below.

$$V(m,n:\square) = 1/N \sum_{\square(k,l)} \sum_{\epsilon w\square} y(m-k,n-l)$$

The direction ɵ* at which |y(m , n) – V(m, n : ɵ*)| is minimum is noted.
V(m, n) = V(m, n : ɵ*) can give the desired result.

Sharpening Filters

Sharpening is used to highlight fine detail in an image.

HIGH PASS SPATIAL FILTERING :

This highpass spatial filter indicates that, the impulse response of the filter should have positive coefficients near its center and negative coefficients in the outer periphery.

For example consider, 3x3 mask

1/9 x

-1	-1	-1
-1	8	-1
-1	-1	-1

HIGHPASS SPATIAL FILTER :

Here, sum of coefficients is **'0'.** Thus , when the mask is over an area of constant or slowly varying gray level, the output of the mask is zero or very small. Reducing the average value of an image to zero implies that the image must have some negative gray levels. If we use positive gray levels, the result of highpass filtering involve some form of scaling or clipping, so the gray levels range of [0,L-1] is used.

Therefore, positive values are not good. Negative value is better.

HIGH- BOOST FILTERING :

To obtain the better result, rather than highpass spatial filtering, we use high-boost filtering.

The highpass filter image is

Highpass =Original image - Lowpass filter image.

The definition of a highboost or high-frequency emphasis filter is given by multiplying the original image by an amplification factor denoted by 'A'.

Highboost = (A)(Original image)-Lowpass filter image

= (A-1)(Original image)+original image-Lowpass filter image.

=(A-1)(original)+(Highpass filter image.

When A=1, It gives the standard highpass result.

When A>1, It gives cthe highboost image which looks more like the original image.

The general process of subtracting a blurred image from an original is given below.

High= (A) (original)- Lowpass

This is known as "Unsharp masking".

The center weight of the mask is

w=9A-1

A>=1 which determines the nature of the filter.

1/9 x

-1	-1	-1
-1	w	-1
-1	-1	-1

Mask for highboost spatial filter

DERIVATIVE FILTERS :

Averaging of pixels produces blurred image. Averaging is analogous to integration. But differentiation produces the opposite effect. i.e. Sharpen the image. The most common method of differentiation in image processing application is the 'gradient'. For a function f(x,y), the gradient of f at coordinates (x,y) is defined as the vector

$$\nabla f = \begin{pmatrix} \partial f / \partial x \\ \partial f / \partial y \end{pmatrix}$$

The magnitude of this vector is

$$\nabla f = \text{mag}(\nabla f) = [(\partial f/\partial x)^2 + \partial(\partial f/\partial y)^2]^{1/2}$$

Consider the image region

Z_1	Z_2	Z_3
Z_4	Z_5	Z_6
Z_7	Z_8	Z_9

Where, Z's denote the values of gray level equation above can be approximated at point Z_5 in a number of ways. The simplest approach is to use the difference $(Z_5 - Z_8)$ in the x direction and $(Z_5 - Z_6)$ in the y direction. These are combined as,

$$\nabla f \approx [(Z_5 - Z_8)^2 + (Z_5 - Z_6)^2]^{1/2}$$

Take magnitude

$$\nabla f \approx |Z_5 - Z_8| + |Z_5 - Z_6|$$

Another method is to use cross differences in equation

$$\nabla f \approx [(Z_5 - Z_9)^2 + (Z_6 - Z_8)^2$$

$$\nabla f \approx |Z_5 - Z_9| + |Z_6 - Z_8|$$

The above equations can be implemented by using the masks of size 2x2.

For example , these equation can be implemented by taking the absolute value of the response of the two masks shown below and summing the results. These masks are called "**Roberts cross-gradient operators**".

1	0
0	-1

0	1
-1	0

If mask is 3x3 , the neighborhood is

$$\nabla f \approx |(Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3)| + |(Z_3 + Z_6 + Z_9) - (Z_1 + Z_4 + Z_7)|$$

The difference between the 3rd and 1st row of the 32x23 region approximates the derivative in the x direction and difference between the 3rd and 1st column approximates the derivative in the y direction called **"prewitt operators "** shown in the figure below.

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

Prewitt operators

For approximating the magnitude of the gradient, another pair of masks can be used which are known as **sobel operators.**

-1	-2	-1
0	0	0

1	2	1

-1	0	1
-2	0	2
-1	0	1

Sobel operators

LAPLACIAN FILTERS :

Laplacian filters is used to enhance discontinuities.

The laplacian of two dimensional image f(x,y) is defined as,

$$\nabla^2 f(x,y) = [\partial^2 f/\partial x^2 + \partial^2 f/\partial y^2]$$

For a 3x3 sub image , the digital form equivalent to the Laplacian operator is given as

$$\nabla^2 f(x,y) = 4P_5 - (P_2 + P_4 + P_6 + P_8)$$

From this equation, it is possible to define the digital laplacian mask. So , the coefficient associated with center pixels should be positive and that associated with the other pixels should be negative.

Moreover, the sum of the coefficients should be zero.

The Laplacian response is sensitive to noise and is rarely used for edge detection.

The 3x3 kernel is

(a)

P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

(b)

0	-1	0
-1	4	-1
0	-1	0

The mask is used to compute Laplacian.

FREQUENCY DOMAIN FILTERS :

Enhancement in frequency domain is achieved by computing the Fourier transform of the image to be enhanced. Then the result is multiplied by a filter transfer function and the inverse transform to produce the enhanced image.

There are three types of frequency domain filters.

i) Smoothing Filters
ii) Sharpening Filters
iii) Homomorphic Filters

Smoothing filters :

Lowpass filters are used for noise smoothging and interpolation.

Highpass filters are used for extracting edges and in sharpening images.

Bandpass filters are used for enhancement of edges and other high pass characterisitics in the presence of noise.

Lowpass filtering :

Smoothing effect is achieved in the frequency domain by attenuating a specified range of high frequency components in the transform of a given image.

The low pass filter can be represented as G(u,v)= H(u,v) F(u,v)

Where,

F(u,v) is the Fourier transform of an image to be smoothed.

H(u,v) is the filter transfer function and

G(u,v) is the enhanced image in the frequency domain.

In order to get the enhanced image in the spatial domain , the inverse transform is applied and is given by,

g(x,y)= Inverse Fourier transform of[H(u,v) F(u,v)]\In the equation G(u,v) we have considered the transform function H(u,v) that gives Z(u,v) by attenuating the high frequency components of F(u,v).

In general, most of the filter transfer functions affect the real and imaginary parts of F(u,v) in the same manner.

These filters are called as **"Zero phase shift filters"**. Because, they do not change the phase of the transform.

Ideal low pass filter :

The two dimensional ideal pass filter treansfer function can be given by

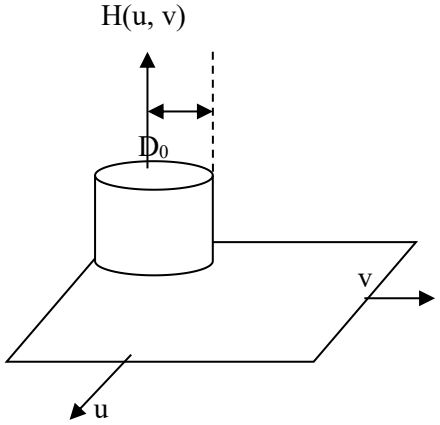
$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \leq D_o \\ 0 & \text{if } D(u,v) > D_o \end{cases}$$

Where, D_o is a specified non negative quantity and

D(u,v) is the distance from point (u,v) to the origin of the frequency plance

i.e. D(u,v)= $\sqrt{u^2 + v^2}$

(a) Perspective plot of an ideal L.P.F transfer function.



(b) Filter Cross section

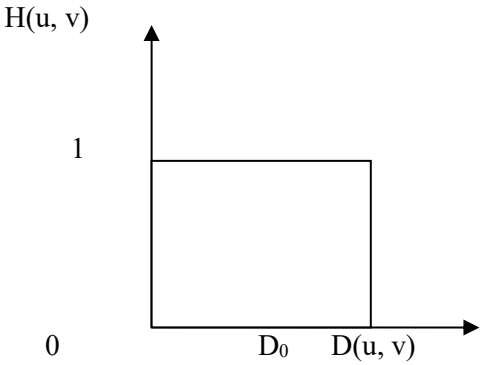


Figure shows that, the filter passes all the frequencies inside the circle of radius D_o where it attenuates all the frequencies outside this circle. Hence , this filter is called "Ideal low pass filter".

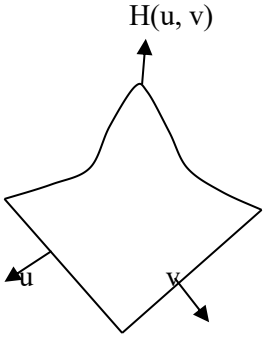
Butterworth lowpass filter :

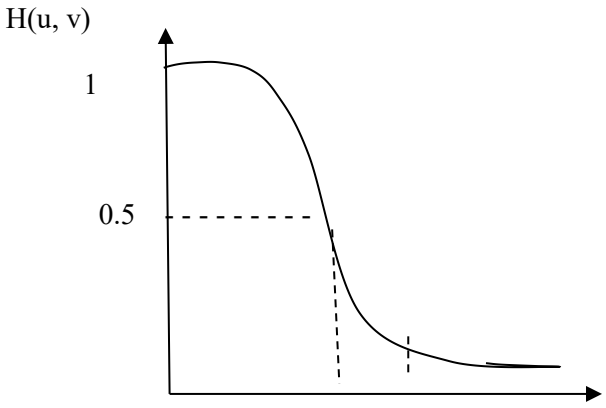
The response of the butterworth low pass filter of order n is defined by the equation

$H(u,v) = 1/1+[D(u,v)/ D_o]^{2n}$

Where ,D(u,v) is the distance from the point (u,v) to the origin and is given by $\sqrt{u^2 + v^2}$

The 3-D and cross sectional views of the butterworth low pass filter response are shown below.





0 D_0 2 3 D(u, v)

When D(u,v)= D_o , H(u,v)=0.5. This indicates that, at cutoff frequency the response is half of its maximum value.

In the most of the cases, at cutoff frequency, the response will be equal to $1/\sqrt{2}$ times the maximum value of H(u,v). To have this effect, the equation should be modified as

$$H(u,v)= 1/1+[\sqrt{2}-1][D(u,v)/ D_o]^{2n}$$

$$H(u,v)= 1/1+0.414[D(u,v)/ D_o]^{2n}$$

SHARPENING FILTER :

The highpass filter is useful in extracting edges and in sharpening images.

The highpass filter passes the high frequency components and it attenuates low frequency components corresponding to slow varying details of the image.

The ideal highpass filter which has sharp or abrupt transition is given by the equation

$$H(uv) = \begin{cases} 0 \text{ if } D(u,v) \le D_o \\ 1 \text{ if } D(u,v) > D_o \end{cases}$$

Where, D_o is the cutoff frequency which is measured from the origin and

D(u,v) is the distance from the origin and is given by

$$D(u,v) = \sqrt{u^2 + v^2}$$

The ideal low pass filter which has the abrupt transition at the cutoff frequency cannot be realized using the electronic circuit components. However, it can be realized with smooth transition frequency and such filters are called as Butterworth filters. The transfer function of the high pass Butterworth filter of order n with a cutoff frequency D_o from the origin is given by

$$H(u, v) = \frac{1}{1 + [D_o / D(u,v)]^{2n}}$$

HOMOMORPHIC FILTERS :

We know that, the **'illumination' and 'reflectance'** components of the image light intensity function f(x, y) are denoted by i(x, y) and r(x, y) respectively.

In frequency domain, it is useful for improving the appearance of an image by using simultaneous brightness range compression and contrast enhancement.

f(x, y) is given as,

$$\mathbf{f(x, y) = i(x, y) \cdot r(x, y)} \qquad \rightarrow \mathbf{(1)}$$

where, $0 < i(x, y) < \infty$ and

$$0 < r(x, y) < 1$$

The equation cannot be used directly, in order to operate separately on the frequency components of illumination and reflectance. Because, in frequency domain, the Fourier transform of the product of two functions is not separable.

$$\text{i.e. } F\{ f(x, y)\} \neq F\{ i(x, y) \} \cdot F\{ r(x, y) \}$$

but we can define

$$z(x, y) = \ln f(x, y)$$

$$= \ln i(x, y) + \ln r(x,y)$$

Then, we can write

$$F\{ z(x, y)\} = F\{\ln f(x, y)\}$$

$$= F\{\ln i(x, y)\} + F\{ \ln r(x,y)\}$$

Otherwise

$$Z(u, v) = I(u, v) + R(u, v)$$

Where I(u, v) and R(u, v) are the Fourier transforms of ln i(x, y) and ln r(x, y) respectively.

We already know that, in frequency domain the convolution of an image f(x, y) and linear position invariant operator h(x, y) is given in Fourier transform as

$$G(u, v) = H(u, v) \cdot F(u, v)$$

H(u, v) = Homomorphic filter function

Write z(u, v) by means of a filter function H(u, v) from above equation.

$$S(u, v) = H(u, v)\, S(u, v)$$

$$S(u, v) = H(u, v)\, I(u, v) + H(u, v)\, R(u, v)$$

Where S(u, v) is the Fourier transform of the result.

In spatial domain representation

$$S(x ,y) = F^{-1} \{ S(u, v)\}$$

$$= F^{-1} \{ H(u, v) I(u, v)\} + F^{-1} \{ H(u, v) R(u, v)\} \quad \rightarrow (2)$$

Let $i'(x, y) = F^{-1} \{ H(u, v) I(u, v)\}$

And $r'(x, y) = F^{-1} \{ H(u, v) R(u, v)\}$

Then equation 2 becomes

$$s(x, y) = i'(x, y) + r'(x, y)$$

Z(x, y) is formed by taking the logarithm of the original image f(z, y). The inverse operation yields the desired enhancement image g(x, y).

So, take the exponential function

$$\therefore \ g (x, y) = \exp [s(x, y)]$$

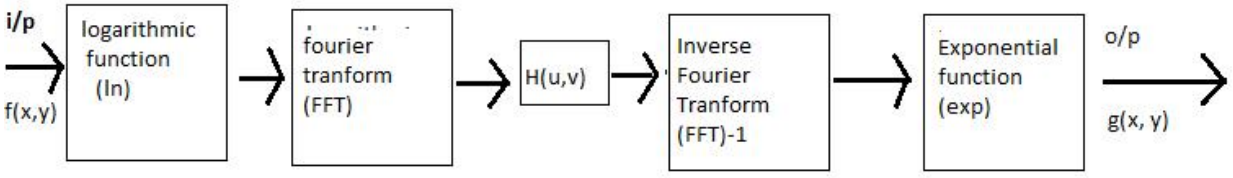
$$= \exp[i'(x, y)] . \exp[r'(x, y)]$$

$$g(x, y) = i_o(x, y) . r_o(x, y)$$

where, $i_o(x, y) = \exp[r'(x, y)]$ and

$$r_o(x, y) = \exp[r'(x, y)]$$

The illumination and reflectance components of the output image.



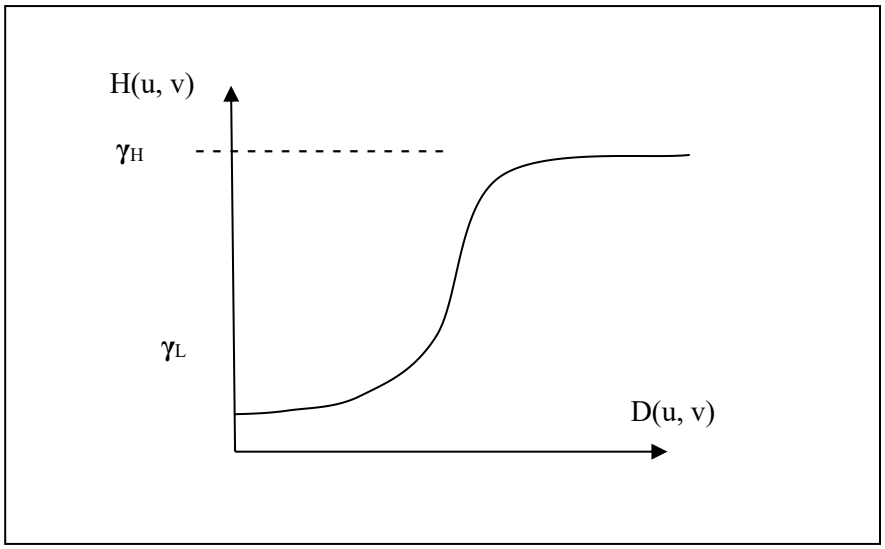
Homomorphic filtering approach for image Enhancement

The diagram shows the above said enhancement techniques. This method is called as 'homomorphic systems'.

In this system, the main thing is that separation of illumination and reflectance components. Then, homomorphic filter function H(u, v) can operate on these components separately.

The filter function H(u, v) that affects the low frequency components of the Fourier transform with illumination and the high frequencies with reflectance in different ways.

So H(u, v) is rotated by 360° cross section about vertical axis.



Cross section of circulatory symmetric filter function

D(u, v) → distance from the origin

The parameters γ_H and γ_L should be

$\gamma_L < 1$ and $\gamma_H > 1$

The filter function shown in fig. is used to decrease the low frequencies and amplify the high frequencies. Finally, are simultaneously done dynamic range compression and contrast enhancement.

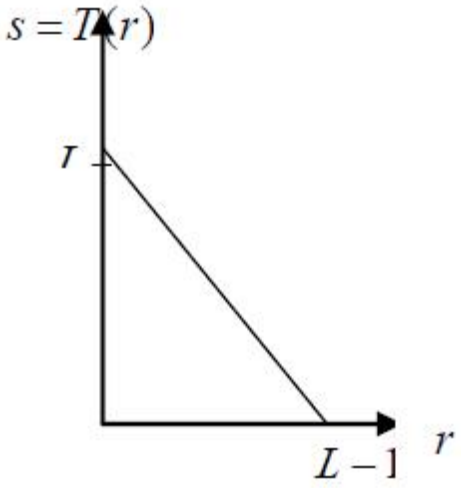
SPATIAL DOMAIN: ENHANCEMENT BY POINT PROCESSING

.

Intensity transformations

Image Negatives

The negative of a digital image is obtained by the transformation function s= T(r)= L-1-r shown in the following figure, where L the number of grey levels. The idea is that the intensity of the output image decreases as the intensity of the input increases. This is useful in numerous applications such as displaying medical images.



Contrast Stretching

Low contrast images occur often due to poor or non uniform lighting conditions, or due to nonlinearity, or small dynamic range of the imaging sensor. In the figure of Example 1 above you have seen a typical contrast stretching transformation.

HISTOGRAM PROCESSING. DEFINITION OF THE HISTOGRAM OF AN IMAGE

.

By processing (modifying) the histogram of an image we can create a new image with specific desired properties. Suppose we have a digital image of size N x N with grey levels in the range [0, L-1]. The histogram of the image is defined as the following discrete function:

$$p(r_k) = \frac{n_k}{N^2}$$

where r is the k_{th} grey level, n_k is the number of pixels in the image with grey level, N^2 is the total number of pixels in the image The histogram represents the frequency of occurrence of the various grey levels in the image. A plot of this function for all values of k provides a global description of the appearance of the image.

GLOBAL HISTOGRAM EQUALIZATION

In this section we will assume that the image to be processed has a continuous intensity that lies within the interval $[0, L-1]$. Suppose we divide the image intensity with its maximum value $L-1$. Let the variable r represent the new grey levels (image intensity) in the image, where now $0 \le r \le 1$ and let $p_r(r)$ denote the probability density function (pdf) of the variable r. We now apply the following transformation function to the intensity

$$s = T(r) = \int_0^r p_r(w)dw, \qquad 0 \le r \le 1$$

(1) By observing the transformation of equation (1) we immediately see that it possesses the following properties:

(i) $0 \le s \le 1$.

(ii) $r_2 > r_1 \Rightarrow T(r_2) \ge T(r_1)$, i.e., the function $T(r)$ is increase ng with r.

(iii) $s = T(0) = \int_0^0 p_r(w)dw = 0$ and $s = T(1) = \int_0^1 p_r(w)dw = 1$. Moreover, if the original image has intensities **only** within a certain range $[r_{\min}, r_{\max}]$ then $s = T(r_{\min}) = \int_0^{r_{\min}} p_r(w)dw = 0$ and $s = T(r_{\max}) = \int_0^{r_{\max}} p_r(w)dw = 1$ since $p_r(r) = 0, r < r_{\min}$ and $r > r_{\max}$. Therefore, the new intensity s takes always all values within the available range [0 1].

Suppose that $P_r(r)$, $P_s(s)$ are the probability distribution functions (PDF's) of the variables r and s respectively.

Let us assume that the original intensity lies within the values r and $r+dr$ with dr a small quantity. dr can be assumed small enough so as to be able to consider the function $p_r(w)$ constant within the interval $[r, r+dr]$ and equal to $p_r(r)$. Therefore,

$$P_r[r, r+dr] = \int_r^{r+dr} p_r(w)dw \cong p_r(r) \int_r^{r+dr} dw = p_r(r)dr.$$

Now suppose that $s = T(r)$ and $s_1 = T(r+dr)$. The quantity dr can be assumed small enough so as to be able to consider that $s_1 = s + ds$ with ds small enough so as to be able to consider the function $p_s(w)$ constant within the interval $[s, s+ds]$ and equal to $p_s(s)$. Therefore,

$$P_s[s, s+ds] = \int_s^{s+ds} p_s(w)dw \cong p_s(s) \int_s^{s+ds} dw = p_s(s)ds$$

Since $s = T(r)$, $s + ds = T(r+dr)$ and the function of equation (1) is increasing with r, all and only the values within the interval $[r, r+dr]$ will be mapped within the interval $[s, s+ds]$. Therefore,

$$P_r[r, r+dr] = P_s[s, s+ds] \Rightarrow p_r(r)dr \overset{r=T^{-1}(s)}{=} p_s(s)ds \Rightarrow p_s(s) = p_r(r)\left.\frac{dr}{ds}\right|_{r=T^{-1}(s)}$$

From equation (1) we see that

$$\frac{ds}{dr} = p_r(r)$$

and hence,

$$p_s(s) = \left[p_r(r)\frac{1}{p_r(r)} \right]_{r=T^{-1}(s)} = 1, \quad 0 \le s \le 1$$

LOCAL HISTOGRAM EQUALISATION

* Global histogram equalisation is suitable for overall enhancement. It is often necessary to enhance details over small areas. The number of pixels in these areas may have negligible influence on the computation of a global transformation, so the use of this type of transformation does not necessarily guarantee the desired local enhancement.

* The solution is to devise transformation functions based on the grey level distribution – or other properties – in the neighbourhood of every pixel in the image. The histogram processing technique previously described is easily adaptable to local enhancement.

* The procedure is to define a square or rectangular neighbourhood and move the centre of this area from pixel to pixel. At each location the histogram of the points in the neighbourhood is computed and a histogram equalisation transformation function is obtained.

* This function is finally used to map the grey level of the pixel centred in the neighbourhood. The centre of the neighbourhood region is then moved to an adjacent pixel location and the procedure is repeated. Since only one new row or column of the neighbourhood changes during a pixel-to-pixel translation of the region, updating the histogram obtained in the previous location with the new data introduced at each motion step is possible quite easily.

* This approach has obvious advantages over repeatedly computing the histogram over all pixels in the neighbourhood region each time the region is moved one pixel location. Another approach often used to reduce computation is to utilise non overlapping regions, but this methods usually produces an undesirable checkerboard effect.

HISTOGRAM SPECIFICATION

Suppose we want to specify a particular histogram shape (not necessarily uniform) which is capable of highlighting certain grey levels in the image.

Let us suppose that:

p (r) is the original probability density function

p(z) is the desired probability density function

Suppose that histogram equalisation is first applied on the original image r

$$s = T(r) = \int_0^r p_r(w)dw$$

- Suppose that the desired image z is available and histogram equalisation is applied as well

$$v = G(z) = \int_0^z p_z(w)dw$$

$p_s(s)$ and $p_v(v)$ are both uniform densities and they can be considered as identical. Note that the final result of histogram equalisation is independent of the density inside the integral. So in equation $v = G(z) = \int_0^z p_z(w)dw$ we can use the symbol s instead of v.

The inverse process $z = G^{-1}(s)$ will have the desired probability density function. Therefore, the process of histogram specification can be summarised in the following steps.

(i) We take the original image and equalise its intensity using the relation $s = T(r) = \int_0^r p_r(w)dw$.

(ii) From the given probability density function $p_z(z)$ we specify the probability distribution function $G(z)$.

(iii) We apply the inverse transformation function $z = G^{-1}(s) = G^{-1}[T(r)]$

Spatial domain: Enhancement in the case of many realizations of an image of interest available
Image averaging

- Suppose that we have an image $f(x,y)$ of size $M \times N$ pixels corrupted by noise $n(x,y)$, so we obtain a noisy image as follows.

$$g(x,y) = f(x,y) + n(x,y)$$

For the noise process $n(x,y)$ the following assumptions are made.

(i) The noise process $n(x,y)$ is ergodic.

(ii) It is zero mean, i.e., $E\{n(x,y)\} = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} n(x,y) = 0$

(ii) It is white, i.e., the autocorrelation function of the noise process defined as $R[k,l] = E\{n(x,y)n(x+k,y+l)\} = \frac{1}{(M-k)(N-l)}\sum_{x=0}^{M-1-k}\sum_{y=0}^{N-1-l} n(x,y)n(x+k,y+l)$ is zero, apart for the pair $[k,l] = [0,0]$.

Therefore, $R[k,l] = \frac{1}{(M-k)(N-l)}\sum_{x=0}^{M-1-k}\sum_{y=0}^{N-1-l} n(x,y)n(x+k,y+l) = \sigma^2_{n(x,y)}\delta(k,l)$ where $\sigma^2_{n(x,y)}$ is the variance of noise.

- Suppose now that we have L different noisy realisations of the same image $f(x,y)$ as $g_i(x,y)=f(x,y)+n_i(x,y)$, $i=0,1,\ldots,L$. Each noise process $n_i(x,y)$ satisfies the properties (i)-(iii) given above. Moreover, $\sigma^2_{n_i(x,y)}=\sigma^2$. We form the image $\bar{g}(x,y)$ by averaging these L noisy images as follows:

$$\bar{g}(x,y)=\frac{1}{L}\sum_{i=1}^{L}g_i(x,y)=\frac{1}{L}\sum_{i=1}^{L}(f(x,y)+n_i(x,y))=f(x,y)+\frac{1}{L}\sum_{i=1}^{L}n_i(x,y)$$

Therefore, the new image is again a noisy realisation of the original image $f(x,y)$ with noise $n(x,y)=\frac{1}{L}\sum_{i=1}^{L}n_i(x,y)$.

The mean value of the noise $n(x,y)$ is found below.

$$E\{n(x,y)\}=E\{\frac{1}{L}\sum_{i=1}^{L}n_i(x,y)\}=\frac{1}{L}\sum_{i=1}^{L}E\{n_i(x,y)\}=0$$

The variance of the noise $n(x,y)$ is now found below.

$$\sigma^2_{n(x,y)}=E\{n^2(x,y)\}=E\left\{\left(\frac{1}{L}\sum_{i=1}^{L}n_i(x,y)\right)^2\right\}=\frac{1}{L^2}E\left\{\left(\sum_{i=1}^{L}n_i(x,y)\right)^2\right\}$$

$$=\frac{1}{L^2}E\{(\sum_{i=1}^{L}{n_i}^2(x,y))\}+\frac{1}{L^2}E\{(\sum_{\substack{i=1\\ i\neq j}}^{L}\sum_{j=1}^{L}(n_i(x,y)n_j(x,y))\}=\frac{1}{L^2}\sum_{i=1}^{L}E\{{n_i}^2(x,y)\}+\frac{1}{L^2}\sum_{\substack{i=1\\ i\neq j}}^{L}\sum_{j=1}^{L}E\{n_i(x,y)n_j(x,y)\}$$

$$=\frac{1}{L^2}\sum_{i=1}^{L}\sigma^2+0=\frac{1}{L}\sigma^2$$

Spatial domain: Enhancement in the case of a single image
Spatial masks

Many image enhancement techniques are based on spatial operations performed on local neighborhoods of input pixels. The image is usually convolved with a finite impulse response filter called spatial mask. The use of spatial masks on a digital image is called spatial filtering. Suppose that we have an image f(x,y) of size N^2 and we define a neighbourhood around each pixel. For example let this neighbourhood to be a rectangular window of size 3 x 3

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Low pass and high pass spatial filtering

A 3 x 3 spatial mask operating on an image can produce (a) a smoothed version of the image (which contains the low frequencies) or (b) it can enhance the edges and suppress essentially the constant background information. The behaviour is basically dictated by the signs of the elements of the mask. Let us suppose that the mask has the following form

a	b	c
d	1	e
f	g	h

To be able to estimate the effects of the above mask with relation to the sign of the coefficients a,b,c,d,e,f,g,h, we will consider the equivalent one dimensional mask

d	1	e

Let us suppose that the above mask is applied to a signal $x(n)$. The output of this operation will be a signal $y(n)$ as

$y(n) = dx(n-1) + x(n) + ex(n+1) \Rightarrow Y(z) = dz^{-1}X(z) + X(z) + ezX(z) \Rightarrow$

$Y(z) = (dz^{-1} + 1 + ez)X(z) \Rightarrow \frac{Y(z)}{X(z)} = H(z) = dz^{-1} + 1 + ez$. This is the transfer function of a system that produces the above input-output relationship. In the frequency domain we have $H(e^{j\omega}) = d\exp(-j\omega) + 1 + e\exp(j\omega)$.

The values of this transfer function at frequencies $\omega = 0$ and $\omega = \pi$ are:

$$H(e^{j\omega})\Big|_{\omega=0} = d + 1 + e$$

$$H(e^{j\omega})\Big|_{\omega=\pi} = -d + 1 - e$$

If a lowpass filtering (smoothing) effect is required then the following condition must hold

$$H(e^{j\omega})\Big|_{\omega=0} \geq H(e^{j\omega})\Big|_{\omega=\pi} = d + e \geq 0$$

If a highpass filtering effect is required then

$$H(e^{j\omega})\Big|_{\omega=0} \leq H(e^{j\omega})\Big|_{\omega=\pi} = d + e \leq 0$$

Popular techniques for low pass spatial filtering

Uniform filtering The most popular masks for low pass filtering are masks with all their coefficients positive and equal to each other as for example the mask shown below. Moreover, they sum up to 1 in order to maintain the mean of the image.

$$\frac{1}{9}\times\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Gaussian filtering

The two dimensional Gaussian mask has values that attempts to approximate the continuous function

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{\frac{x^2+y^2}{\sigma^2}}$$

In theory, the Gaussian distribution is non-zero everywhere, which would require an infinitely large convolution kernel, but in practice it is effectively zero more than about three standard deviations from the mean, and so we can truncate the kernel at this point. The following shows a suitable integer-valued convolution kernel that approximates a Gaussian with a σ of 1.0

Median filtering

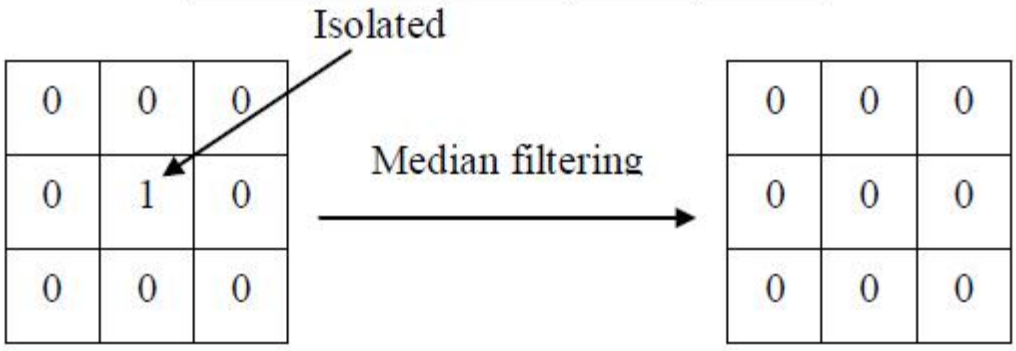
The median m of a set of values is the value that possesses the property that half the values in the set are less than m and half are greater than m. Median filtering is the operation that replaces each pixel by the median of the grey level in the neighbourhood of that pixel. Median filters are non linear filters because for two sequences x(n) and y(n)

$$\text{median}\{x(n)+y(n)\}\neq\text{median}\{x(n)\}+\text{median}\{y(n)\}$$

Median filters are useful for removing isolated lines or points (pixels) while preserving spatial resolutions. They perform very well on images containing binary (**salt and pepper**) noise but perform poorly when the noise is Gaussian. Their performance is also poor when the number of noise pixels in the window is greater than or half the number of pixels in the window

$\frac{1}{273}\times$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1



Directional smoothing

To protect the edges from blurring while smoothing, a directional averaging filter can be useful. Spatial averages g(x,y: θ) are calculated in several selected directions (for example could be horizontal, vertical, main diagonals)

$$g(x,y:\theta)=\frac{1}{N_\theta}\sum_{(k,l)\in W_\theta}\sum f(x-k,y-l)$$

and a direction θ^* is found such that $\left|f(x,y)-g(x,y:\theta^*)\right|$ is minimum. (Note that W_θ is the neighbourhood along the direction θ and N_θ is the number of pixels within this neighbourhood).

Then by replacing $g(x,y:\theta)$ with $g(x,y:\theta^*)$ we get the desired result.

High Boost Filtering

A high pass filtered image may be computed as the difference between the original image and a lowpass filtered version of that image as follows:

(Highpass part of image)=(Original)-(Lowpass part of image)

Multiplying the original image by an amplification factor denoted by A, yields the so called high boost filter:

(Highboost image)= (A) (Original)-(Lowpass)= (A-1) (Original)+(Original)- (Lowpass)

=(A-1) (Original)+(Highpass)

The general process of subtracting a blurred image from an original as given in the first line is called ***unsharp masking***. A possible mask that implements the above procedure could be the one illustrated below.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & A & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{1}{9} \times \begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\frac{1}{9} \times \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9A-1 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Noise models

The general noise model

$$n(i,j) = g\{r[w(i,j)]\}n_1(i,j) + n_2(i,j)$$

is applicable in many situations. Example, in photoelectronic systems we may have $g(x) = \sqrt{x}$. Therefore,

$$n(i,j) = \sqrt{\alpha w(i,j)^{\beta}}\, n_1(i,j) + n_2(i,j)$$

where n_1 and n_2 are zero-mean, mutually independent, Gaussian white noise fields. The term $n_2(i,j)$ may be referred as thermal noise. In the case of films there is no thermal noise and the noise model is

$$n(i,j) = \sqrt{\gamma \log_{10} w(i,j) - r_o}\, n_1(i,j)$$

Because of the signal-dependent term in the noise model, restoration algorithms are quite difficult. Often $w(i,j)$ is replaced by its spatial average, μ_w, giving

$$n(i,j) = g[r[\mu_w]]n_1(i,j) + n_2(i,j)$$

which makes $n(i,j)$ a Gaussian white noise random field. A lineal observation model for photoelectronic devices is

$$y(i,j) = w(i,j) + \sqrt{\mu_w}\, n_1(i,j) + n_2(i,j)$$

For photographic films with $\gamma = -1$

$$y(i,j) = -\log_{10} w(i,j) - r_0 + a n_1(x,y)$$

where r_0, a are constants and r_0 can be ignored.

The light intensity associated with the observed optical density $y(i,j)$ is

$$I(i,j) = 10^{-y(i,j)} = w(i,j)10^{-a n_1(i,j)} = w(i,j)n(i,j)$$

where $n(i,j) \hat{=} 10^{-an_1(i,j)}$ now appears as multiplicative noise having a log-normal distribution.

Linear position invariant degradation models

We again consider the general degradation model

$$y(i,j) = H\left[f(i,j)\right] + n(i,j)$$

If we ignore the presence of the external noise $n(i,j)$ we get

$$y(i,j) = H\left[f(i,j)\right]$$

H is ***linear*** if

$$H\left[k_1 f_1(i,j) + k_2 f_2(i,j)\right] = k_1 H\left[f_1(i,j)\right] + k_2 H\left[f_2(i,j)\right]$$

H is ***position*** (or ***space***) ***invariant*** if

From now on we will deal with linear, space invariant type of degradations.

SOME CHARACTERISTIC METRICS FOR DEGRADATION MODELS

- **Blurred Signal-to-Noise Ratio (BSNR)**: a metric that describes the degradation model.

$$\text{BSNR} = 10\log_{10}\left\{\frac{\frac{1}{MN}\sum_i\sum_j\left[g(i,j) - \bar{g}(i,j)\right]^2}{\sigma_n^2}\right\}$$

$g(i,j) = y(i,j) - n(i,j)$

$\bar{g}(i,j) = E\{g(i,j)\}$

σ_n^2 : variance of additive noise

- **Improvement in SNR (ISNR)**: validates the performance of the image restoration algorithm.

$$\text{ISNR} = 10\log_{10}\left\{\frac{\sum_i\sum_j\left[f(i,j) - y(i,j)\right]^2}{\sum_i\sum_j\left[f(i,j) - \hat{f}(i,j)\right]^2}\right\}$$

where $\hat{f}(i,j)$ is the restored image.

ONE DIMENSIONAL DISCRETE DEGRADATION MODEL. CIRCULAR CONVOLUTION

Suppose we have a one-dimensional discrete signal $f(i)$ of size A samples $f(0), f(1),\ldots, f(A-1)$, which is due to a degradation process. The degradation can be modeled by a one-dimensional discrete impulse response $h(i)$ of size B samples. If we assume that the degradation is a causal function we have the samples $h(0), h(1),\ldots, h(B-1)$. We form the extended versions of $f(i)$ and $h(i)$, both of size $M \geq A+B-1$ and periodic with period M. These can be denoted as $f_e(i)$ and $h_e(i)$. For a time invariant degradation process we obtain the discrete convolution formulation as follows

$$y_e(i) = \sum_{m=0}^{M-1} f_e(m) h_e(i-m) + n_e(i)$$

Using matrix notation we can write the following form

$$\mathbf{y} = \mathbf{H}\mathbf{f} + \mathbf{n}$$

$$\mathbf{f} = \begin{bmatrix} f_e(0) \\ f_e(1) \\ \vdots \\ f_e(M-1) \end{bmatrix},$$

$$\underset{(M\times M)}{\mathbf{H}} = \begin{bmatrix} h_e(0) & h_e(-1) & \ldots & h_e(-M+1) \\ h_e(1) & h_e(0) & \ldots & h_e(-M+2) \\ \vdots & \vdots & \ddots & \vdots \\ h_e(M-1) & h_e(M-2) & \ldots & h_e(0) \end{bmatrix}$$

At the moment we decide to ignore the external noise $\mathbf{n}$. Because h is periodic with period M we have that

$$\underset{(M\times M)}{\mathbf{H}} = \begin{bmatrix} h_e(0) & h_e(M-1) & \ldots & h_e(1) \\ h_e(1) & h_e(0) & \ldots & h_e(2) \\ \vdots & \vdots & \ddots & \vdots \\ h_e(M-1) & h_e(M-2) & \ldots & h_e(0) \end{bmatrix}$$

We define $\lambda(k)$ to be

$$\lambda(k) = h_e(0) + h_e(M-1)\exp(j\frac{2\pi}{M}k) + h_e(M-2)\exp(j\frac{2\pi}{M}2k) + \ldots$$

$$+ h_e(1)\exp[j\frac{2\pi}{M}(M-1)k], \quad k = 0,1,\ldots,M-1$$

Because $\exp[j\frac{2\pi}{M}(M-i)k] = \exp(-j\frac{2\pi}{M}ik)$ we have that

$$\lambda(k) = MH(k)$$

$H(k)$ is the discrete Fourier transform of $h_e(i)$.

I define $\mathbf{w}(k)$ to be

$$\mathbf{w}(k) = \begin{bmatrix} 1 \\ \exp(j\frac{2\pi}{M}k) \\ \vdots \\ \exp[j\frac{2\pi}{M}(M-1)k] \end{bmatrix}$$

It can be seen that

$$\mathbf{H}\mathbf{w}(k) = \lambda(k)\mathbf{w}(k)$$

This implies that $\lambda(k)$ is an eigenvalue of the matrix $\mathbf{H}$ and $\mathbf{w}(k)$ is its corresponding eigenvector.

We form a matrix $\mathbf{w}$ whose columns are the eigenvectors of the matrix $\mathbf{H}$, that is to say

$$\mathbf{W} = [\mathbf{w}(0) \quad \mathbf{w}(1) \quad \dots \quad \mathbf{w}(M-1)]$$

$$w(k,i) = \exp\left[j\frac{2\pi}{M}ki\right] \text{ and } w^{-1}(k,i) = \frac{1}{M}\exp\left[-j\frac{2\pi}{M}ki\right]$$

We can then diagonalize the matrix $\mathbf{H}$ as follows

$$\mathbf{H} = \mathbf{W}\mathbf{D}\mathbf{W}^{-1} \Rightarrow \mathbf{D} = \mathbf{W}^{-1}\mathbf{H}\mathbf{W}$$

where

$$\mathbf{D} = \begin{bmatrix} \lambda(0) & & & \mathbf{0} \\ & \lambda(1) & & \\ & & \ddots & \\ \mathbf{0} & & & \lambda(M-1) \end{bmatrix}$$

Obviously $\mathbf{D}$ is a diagonal matrix and

$$D(k,k) = \lambda(k) = MH(k)$$

If we go back to the degradation model we can write

$$\mathbf{y} = \mathbf{H}\mathbf{f} \Rightarrow \mathbf{y} = \mathbf{W}\mathbf{D}\mathbf{W}^{-1}\mathbf{f} \Rightarrow \mathbf{W}^{-1}\mathbf{y} = \mathbf{D}\mathbf{W}^{-1}\mathbf{f} \Rightarrow$$

$$Y(k) = MH(k)F(k), k = 0,1,\dots,M-1$$

$Y(k), H(k), F(k), k = 0,1,\dots,M-1$ are the $M-$sample discrete Fourier transforms of $y(i), h(i), f(i)$, respectively. So by choosing $\lambda(k)$ and $\mathbf{w}(k)$ as above and assuming that $h_e(i)$ is periodic, we start with a matrix problem and end up with M scalar problems.

TWO DIMENSIONAL DISCRETE DEGRADATION MODEL. CIRCULAR CONVOLUTION

Suppose we have a two-dimensional discrete signal $f(i,j)$ of size $A \times B$ samples which is due to a degradation process. The degradation can now be modeled by a two dimensional discrete impulse response $h(i,j)$ of size $C \times D$ samples. We form the extended versions of $f(i,j)$ and $h(i,j)$, both of size $M \times N$, where $M \geq A + C - 1$ and $N \geq B + D - 1$, and periodic

with period $M \times N$. These can be denoted as $f_e(i,j)$ and $h_e(i,j)$. For a space invariant degradation process we obtain

$$y_e(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f_e(m,n) h_e(i-m, j-n) + n_e(i,j)$$

Using matrix notation we can write the following form

$$\mathbf{y} = \mathbf{H}\mathbf{f} + \mathbf{n}$$

where $\mathbf{f}$ and $\mathbf{y}$ are MN – dimensional column vectors that represent the lexicographic ordering of images $f_e(i,j)$ and $h_e(i,j)$ respectively.

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_0 & \mathbf{H}_{M-1} & \cdots & \mathbf{H}_1 \\ \mathbf{H}_1 & \mathbf{H}_0 & \cdots & \mathbf{H}_2 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{H}_{M-1} & \mathbf{H}_{M-2} & \cdots & \mathbf{H}_0 \end{bmatrix}$$

$$\mathbf{H}_j = \begin{bmatrix} h_e(j,0) & h_e(j,N-1) & \cdots & h_e(j,1) \\ h_e(j,1) & h_e(j,0) & \cdots & h_e(j,2) \\ \vdots & \vdots & \ddots & \vdots \\ h_e(j,N-1) & h_e(j,N-2) & \cdots & h_e(j,0) \end{bmatrix}$$

The analysis of the diagonalisation of $\mathbf{H}$ is a straightforward extension of the one-dimensional case.

In that case we end up with the following set of $M \times N$ scalar problems.

$$Y(u,v) = MNH(u,v)F(u,v)(+N(u,v))$$
$$u = 0,1,\ldots,M-1, v = 0,1,\ldots,N-1$$

IMAGE RESTORATION

Restoration refers to a class of methods that aim to remove or reduce the degradations that have occurred while the digital image was being obtained. All natural images when displayed have gone through some sort of degradation:

- during display mode
- during acquisition mode, or
- during processing mode

The degradations may be due to

- sensor noise
- blur due to camera misfocus
- relative object-camera motion
- random atmospheric turbulence
- others

In most of the existing image restoration methods we assume that the degradation process can be described using a mathematical model.

- Image restoration differs from image enhancement in that the latter is concerned more with accentuation or extraction of image features rather than restoration of degradations.
- Image restoration problems can be quantified precisely, whereas enhancement criteria are difficult to represent mathematically.

Possible classification of restoration methods

Restoration methods could be classified as follows:

- **deterministic:** we work with sample by sample processing of the observed (degraded) image
- **stochastic:** we work with the statistics of the images involved in the process
- **non-blind:** the degradation process H is known
- **blind:** the degradation process H is unknown
- **semi-blind:** the degradation process H could be considered partly known

From the viewpoint of implementation:

- **direct**
- **iterative**
- **recursive**

PONDICHERRY UNIVERSITY QUESTIONS

GRAPHICS AND IMAGE PROCESSING

1. Discuss about Lowpass Filtering and Highpass Filtering. **(APR 2012) (Pg.no. 26)**
2. Explain Degradation Model. **(APR 2012) (APRIL 2013) (Pg.no. 14)**
3. What is histogram? Explain the use of histogram statistics for image enhancement. **(NOV 2012) (Pg.no. 1)**
4. What are three principal ways to estimate degradation function for use in image restoration Explain. **(NOV 2012) (Pg.no. 14)**
5. List out and discuss the different type of Spatial Filtering. **(APR 2013) (Pg.no. 26)**
6. What is Periodic Noise? How Periodic noises can be eliminated? Explain. **(NOV 2013) (Pg.no. 22)**
7. What is "imnoice"? Explain its various functions with an example for each.**(NOV 2013)**
8. Explain how histogram equalization techniques is useful for image enhancement. **(APR 2014) (Pg.no. 01)**
9. Explain about image noise model. **(APR 2014) (Pg.no. 20)**

UNIT-V

Image Compression: Compression Models and measures – coding types – Types of Redundancy - Lossless compression algorithms – Lossy compression algorithms – Introduction to compression standards.

Image Segmentation: Detection of Discontinuities – Edge Detection – Thresholding – Region Based Segmentation.

Introduction to Color Image Processing. Introduction to Morphological operations.

Image Compression Models:

Three general techniques for reducing or compressing the amount of data required to represent an image. However, these techniques typically are combined to form practical image compression systems.

- A compression system consists of two distinct structural blocks: an encoder and a decoder. An input image f(x, y) is fed into the encoder, which creates a set of symbols from the input data.
- After transmission over the channel, the encoded representation is fed to the decoder, where a reconstructed output image j(x, y) is generated. In general(x, y) may or may not be an exact replica of f(x, y). If it is, the system is error free or information preserving; if not, some level of distortion is present in the reconstructed image.
- It consists of two relatively in-dependent functions or sub blocks. The encoder is made up of a source encoder, which removes input redundancies, and a channel encoder, which increases the noise immunity of the source encoder's output.
- As would be expected, the de-coder includes a channel decoder followed by a source decoder. If the channel between the encoder and decoder is noise free (not prone to error), the channel encoder and decoder are omitted, and the general encoder and decoder become the source encoder and decoder, respectively.

Source Encoder and Decoder:

- The source encoder is responsible for reducing or eliminating any coding, inters pixel, or psycho visual redundancies in the input image.
- The specific application and associated fidelity requirements dictate the best encoding approach to use in any given situation. Normally, the approach can be modeled by a series of three independent operations.
- Each operation is designed to reduce one of the three redundancies depicts the corresponding source decoder. In the first stage of the source encoding process, the mapper transforms the input data into a (usually nonvisual) format designed to reduce inter pixel redundancies in the input image. This operation generally is reversible and may or may not reduce directly the amount of data required to represent the image.

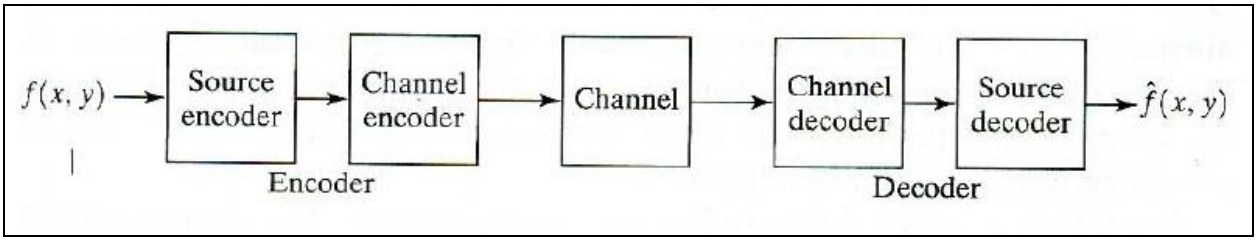


Fig :Source encoder model

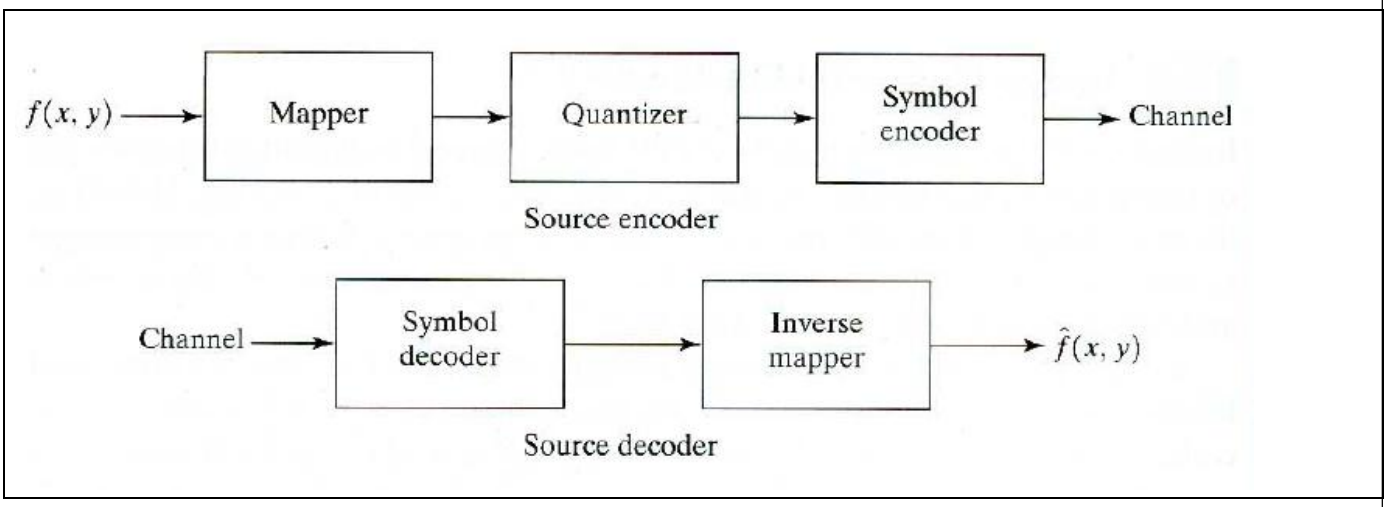


Fig: Source decoder model

- Run-length coding is an example of a mapping that directly results in data compression in this initial stage of the overall source en-coding process. The representation of an image by a set of transform coefficients is an example of the opposite case. Here, the mapper transforms the image into an array of coefficients, making its inter pixel redundancies more accessible for compression in later stages of the encoding process.
- The second stage, or quantize, block reduces the accuracy of the mapper's output in accordance with some pre-established fidelity criterion. This stage reduces the psycho visual redundancies of the input image. This operation is irreversible. Thus it must be omitted when error free compression is desired.
- In the third and final stage of the source encoding process, the symbol coder creates a fixed-or variable-length code to represent the quantizer output and maps the output in accordance with the code.
- The term symbol coder distinguishes this coding operation from the overall source encoding process. In most cases, a variable-length code is used to represent the mapped and quantized data set; it assigns the shortest code words to the most frequently occurring out- put values and thus reduces coding redundancy. The operation, of course, is reversible. Upon completion of the symbol coding step, the input image has been processed to remove each of the three redundancies.
- The source encoding process as three successive operations, but all three operations are not necessarily included in every compression system.

- The source decoder contains only two components: a symbol decoder and an inverse mapper. These blocks perform, in reverse order, the inverse operations of the source encoder's symbol encoder and mapper blocks because quantization results in irreversible information loss, an inverse quantizer block is not included in the general source decoder model.
- The channel encoder and decoder play an important role in the overall encoding-decoding process when the channel is noisy or prone to error.
- They are designed to reduce the impact of channel noise by inserting a con- trolled form of redundancy into the source encoded data. As the output of the source encoder contains little redundancy, it would be highly sensitive to trans- mission noise without the addition of this "controlled redundancy."
- One of the most useful channel encoding techniques was devised by R. W. Hamming (Hamming [1950]). It is based on appending enough bits to the data being encoded to ensure that some minimum number of bits must change be- tween valid code words. Hamming showed, for example, that if 3 bits of redundancy are added to a 4-bit word, so that the distance' between any two valid code words is 3, all single-bit errors can be detected and corrected. (By appending additional bits of redundancy, multiple-bit errors can be detected and corrected.) The 7-bit Hamming (7, 4) code word h, h, ... h,h6h, associated with a 4-bit binary number $b_3b_2b_1b_0$is

$$h_1 = b_3 \oplus b_2 \oplus b_0 \qquad h_3 = b_3$$
$$h_2 = b_3 \oplus b_1 \oplus b_0 \qquad h_5 = b_2$$
$$h_4 = b_2 \oplus b_1 \oplus b_0 \qquad h_6 = b_1$$
$$h_7 = b_0$$

Where⊕ denotes the exclusive OR operation. Note that bits h" hz, and h4 are even- parity bits for the bit fields $b_3b_2b_0$, $b_3b_1b_0$, and $b_2b_1b_0$", respectively. (Recall that a string of binary bits has even parity if the number of bits with a value of is even.)

To decode a Hamming encoded result, the channel decoder must check the encoded value for odd parity over the bit fields in which even parity was previously established. A single-bit error is indicated by a nonzero parity word $c_4c_2c_1$, Where

$$c_1 = h_1 \oplus h_3 \oplus h_5 \oplus h_7$$
$$c_2 = h_2 \oplus h_3 \oplus h_6 \oplus h_7$$
$$c_4 = h_4 \oplus h_5 \oplus h_6 \oplus h_7$$

If a nonzero value is found, the decoder simply complements the code word bit position indicated by the parity word. The decoded binary value is then extracted from the corrected code word as $h_3h_5h_6h_7$.

Elements of Information Theory:

Several ways to reduce the amount of data used to represent an image.

Measuring Information:

- The fundamental premise of information theory is that the generation of in-formation can be modeled as a probabilistic process that can be measured in a manner that agrees with intuition.
- In accordance with this supposition, a random event E that occurs with probability p (E) is said to contain

$$I(E) = \log\frac{1}{P(E)} = -\log P(E)$$

Units of information. The quantity I(E) often is called the self-information of E. Generally speaking, the amount of self-information attributed to event E is inversely related to the probability of E. If P(E) = 1 (that is, the event always occurs), I(E) = 0 and no information is attributed to it.

The Information Channel:

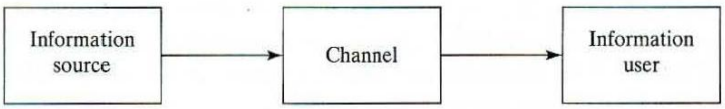
- When self-information is transferred between an information source and a user of the information, the source of information is said to be-connected to the user of information by an information channel.
- The information channel is the physical medium that links the source to the user. It may be a telephone line, an electromagnetic energy propagation path, or a wire in a digital computer. A simple mathematical model for a discrete information system. Here, the parameter of particular interest is the system's capacity, defined as its ability to transfer information.
- Generates a random sequence of symbols from a finite or countable infinite set of possible symbols. That is, the output of the source is a discrete random variable. The set of source symbols $\{a_1,a_2,..,a_J\}$ is referred to as the source alphabet A, and the elements of the set, denoted a_j, are called symbols or letters.
- The probability of the event that the source will produce symbol a_j is $P(a_j)$, and

$$\sum_{j=1}^{J} P(a_j) = 1.$$

A J X 1 vector $z = [P(a_1), p(a_2), \ldots, p(a_J)]^T$ customarily is used to represent the set of all source symbol probabilities $\{P(a_1), p(a_2), \ldots, p(a_J)\}$ The finite ensemble (A, z) describes the information source completely.

The average information per source output, denoted H(z), is

$$H(\mathbf{z}) = -\sum_{j=1}^{J} P(a_j) \log P(a_j).$$



- This quantity is called the uncertainty or entropy of the source. It defines the average amount of information obtained by observing a single source output. As its magnitude increases, more uncertainty and thus more information is associated with the source.
- Having modeled the information source, we can develop the input-output characteristics of the information channel rather easily. Because we modeled the input to the channel in as a discrete random variable, the information transferred to the output of the channel is also a discrete random variable. Like the source random variable, it takes on values from a finite or countable infinite set of symbols $\{b_1, b_2, \ldots, b_K\}$ called the channel alphabet, B. The probability of the event that symbol b_k is presented to the information user is $P(b_k)$. The finite ensemble (B,v), where $v = [P(b_1), p(b_2), \ldots, P(b_K)]^T$, describes the channel output completely and thus the information received by the user.
- The probability $P(b_k)$ of a given channel output and the probability distribution of the source z are related by the expression

$$P(b_k) = \sum_{j=1}^{J} P(b_k | a_j) P(a_j)$$

Where $P(b_K | a_j)$ is the conditional probability that output symbol b_K is received, given that source symbol a_j was generated. If the conditional probabilities referenced in are arranged in a matrix K X J matrix **Q**, such that

$$\mathbf{Q} = \begin{bmatrix} P(b_1|a_1) & P(b_1|a_2) & \cdots & P(b_1|a_J) \\ P(b_2|a_1) & \cdot & \cdots & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & & & \\ P(b_K|a_1) & P(b_K|a_2) & \cdots & P(b_K|a_J) \end{bmatrix}$$

Then the probability distribution of the complete output alphabet can be computed from

$$\mathbf{v = Qz}$$

Matrix **Q**, with elements $q_{kj} = P(b_K | a_j)$, is referred to as the forward channel *transition matrix* or by the abbreviated term channel matrix.

$$H(\mathbf{z} | b_k) = -\sum_{j=1}^{J} P(a_j | b_k) \log P(a_j | b_k)$$

Where $P(a_j \mid b_k)$, is the probability that symbol a_j was transmitted by the source, given that the user received bk• The expected (average) value of this expression over all bk is

$$H(\mathbf{z}\|\mathbf{v}) = \sum_{k=1}^{K} H(\mathbf{z}\,|\,b_k)P(b_k),$$

which, after substitution of Eg. (8.3·7) for $H(\mathbf{z}\mathrm{I}\ bk\ )$ and some minor rearrangement, can be written as

$$H(\mathbf{z}\,|\,\mathbf{v}) = -\sum_{j=1}^{J}\sum_{k=1}^{K} P(a_j, b_k)\log P(a_j\,|\,b_k).$$

Here, $P(a_j, b_k)$ is the joint probability of a_j and b_k. That is, $p(a_j, b_k)$ is the probability that a_j is transmitted *and* b_k is received.

- The term H(**z**| **v**) is called the equivocation of z with respect to v. It represents the average information of one source symbol, assuming observation of the output symbol that resulted from its generation.
- Because H(**z**) is the average information of one source symbol, assuming no knowledge of the resulting output symbol, the difference between H(z) and H(z lv) is the average information received upon observing a single output symbol. This difference, denoted I(**z**, **v**) and called the mutual information of **z** and **v**, is

$$I(\mathbf{z}, \mathbf{v}) = H(\mathbf{z}) - H(\mathbf{z}\,|\,\mathbf{v})$$

for $H(z)$ and H (**z**|**v**), and recalling that
$P(a_j) = P(a_j, \mathrm{b}_1) + P(a_j' b_2,) + \ldots + P(a_j, b_k),$ yields

$$I(\mathbf{z}, \mathbf{v}) = \sum_{j=1}^{J}\sum_{k=1}^{K} P(a_j, b_k)\log\frac{P(a_j, b_k)}{P(a_j)P(b_k)}$$

This, after further manipulation, can be written as

$$I(\mathbf{z}, \mathbf{v}) = \sum_{j=1}^{J}\sum_{k=1}^{K} P(a_j)q_{kj}\log\frac{q_{kj}}{\sum_{i=1}^{J} P(a_i)q_{ki}}$$

Thus the average information received upon observing a single output of the information channel is a function of the input or source symbol probability vector **z** and channel matrix **Q**.

Error-Free Compression:

- In numerous applications error-free compression is the only acceptable means of data reduction. One such application is the archival of medical or business documents, where lossy compression usually is prohibited for legal reasons.
- Another is the processing of satellite imagery, where both the use and cost of collecting the data makes any loss undesirable. Yet another is digital radiography, where the loss of information can compromise diagnostic accuracy. In these another cases, the need for error-free compression is motivated by the intended use or nature of the images under consideration.

- We focus on the principal error-free compression strategies currently in use. They normally provide compression ratios of 2 to 10. Moreover, they are equally applicable to both binary and gray-scale images.
- Error-free compression techniques generally are composed of two relatively independent operations:

  (1) Devising an alternative representation of the image in which its inter pixel redundancies are reduced;

  (2) Coding the representation to eliminate coding redundancies. These steps correspond to the mapping and symbol coding operations of the source coding model discussed in connection

Variable-Length Coding:

- The simplest approach to error-free image compression is to reduce only coding redundancy. Coding redundancy normally is present in any natural binary encoding of the gray levels in an image. It can be eliminated by coding the gray levels so is minimized.

- To do so requires construction of a variable-length code that assigns the shortest possible code words to the most probable gray levels. Here. We examine several optimal and near optimal techniques for constructing such a code.
- These techniques are formulated in the language of information theory. The source symbols may be either the gray levels of an image or the output of a gray-level mapping operation (pixel differences, run lengths, and so on).

Huffman coding:

- The most popular technique for removing coding redundancy is due to Huffman (Huffman [1952]).
- When coding the symbols of an information source individually, Huffman coding yields the smallest possible number of code symbols resource symbol. In terms of the noiseless coding theorem, the resulting code is optimal for a fixed value of n, subject to the constraint that the source symbols be coded one at a time.
- The first step in Huffman's approach is to create a series of source reductions by ordering the probabilities of the symbols under consideration and combining the lowest probability symbols into a single symbol that replaces them in the next source reduction.
- At the far left, a hypothetical set of source symbols and their probabilities are ordered from top to bottom in terms of decreasing probability values. To form the first source reduction, the bottom two probabilities, 0.06 and 0.04, are combined to form a "compound symbol" with probability 0.1.
- This compound symbol and its associated probability are placed in the first source reduction column so that the probabilities of the reduced source are also ordered from the most to the least probable. This process is then repeated until a reduced source with two symbols is reached.
- The second step in Huffman's procedure is to code each reduced source, starting with the smallest source and working back to the original source. The length binary code for a two-symbol source,

of course, is the symbols0and 1, these symbols are assigned to the two symbols on the right (the assignment is arbitrary; reversing the order of the 0 and 1 would work just as well). As the reduced source symbol with probability 0.6 was generated by combining two symbols in the reduced source to its left, the 0 used to code it is now assigned to *both* of these symbols, and a 0 and 1 are arbitrarily

Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

Original source			Source reduction							
Sym.	Prob.	Code	1		2		3		4	
a_2	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
a_6	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
a_1	0.1	011	0.1	011	0.2	010	0.3	01		
a_4	0.1	0100	0.1	0100	0.1	011				
a_3	0.06	01010	0.1	0101						
a_5	0.04	01011								

Appended to each to distinguish them from each other. This operation is then repeated for each reduced source until the original source is reached. The final code appears at the far. The average length of this code is

$$L_{avg} = (0.4)(1) + (0.3)(2) + (0.1)(3) + (0.1)(4) + (0.06)(5) + (0.04)(5)$$
$$= 2.2 \text{ bits/symbol}$$

And the entropy of the source is 2.14 bits/ symbol. In accordance with the resulting Huffman code efficiency is 0.973.Huffman's procedure creates the optimal code for a set of symbols and probabilities subject t*o* the constraint that the symbols be coded one at a time. After the code has been created, coding and/or decoding is accomplished in a simple lookup table manner.

Other near optimal variable length codes:

When a large number of symbols are to be coded, the construction of the optimal binary Huffman code is a nontrivial task. For the general case of J source symbols, J - 2 source reductions must be performed and J - 2 code assignments made. Thus construction of the optimal Huffman code for an image with 256 gray levels requires 254 source reductions and 254code assignments. In view of the computational complexity of this task, sacrificing coding efficiency for simplicity in code construction sometimes is necessary.

A truncated Huffman code is generated by Huffman coding only the most probable ψ symbols of the source, for some positive integer less than second, near optimal and variable-length code known as a B-code. It is close to optimal when the source symbol probabilities obey a power law of the form

$$P(a_j) = cj^{-\beta}$$

Forsome positive constant β and normalizing constant

$$c = 1/\sum_{j=0}^{J} j^{-\beta}.$$

For example, the distribution of run lengths in a binary representation of a typical type written text document is nearly exponential.

Arithmetic coding:

- In arithmetic coding, which can be traced to the work of, a one-to-one correspondence between source symbols and code words does not exist. Instead, an entire sequence of source symbols (or message) is assigned a single arithmetic code word.
- The code word itself defines an interval of real numbers between 0 and 1. As the number of symbols in the message increases, the interval used to represent it becomes smaller and the number of information units (say, bits) required to represent the interval becomes larger.
- Each symbol of the message reduces the size of the interval in accordance with its probability of occurrence. Because the technique does not require, as does Huffman's approach, that each source symbol translate into an integral number of code symbols (that is, that the symbols be coded one at time), it achieves (but only in theory) the bound established by the noise less coding theorem
- Here, a five-symbol sequence or message, a1a2a3a3a4, from a four-symbol source is coded. At the start of the coding process, the message is assumed to occupy the entire half open interval [0,1).

LZW Coding:

- The technique, called Lempel-Ziv-Welch (LZW) coding, assigns fixed-length code words to variable length sequences of source symbols but requires no a priori knowledge of the probability of occurrence of the symbols to be encoded.
- **LZW** coding is conceptually very simple. At the onset of the Coding process, a code book or "dictionary" containing the source symbols to be coded is constructed.
- For 8-bit monochrome images, the first 256 words of the dictionary are assigned to the gray values 0, 1, 2, ..., 255.
- As the encoder sequentially examines the image's pixels, gray-level sequences that are not in the dictionary are placed in algorithmically determined (e.g., the next unused) locations. If the first two pixels of the image are white, for instance, sequence"255-255" might be assigned to location 256, the address following the locations reserved for gray levels 0 through 255.
- The next time that two consecutive white pixels are encountered, code word 256, the address of the location containing sequence 255-255, is used to represent them. If a 9-bit, 512-word dictionary is

employed in the coding process, the original (8 + 8) bits that were used to represent the two pixels are replaced by a single 9-bit code word.

- Cleary, the size of the dictionary is an imp9rtant system parameter. If it is too small, the detection of matching gray-level sequences will be less likely; if it is too large, the size of the code words will adversely affect compression performance.

Consider the following 4 × 4, 8-bit image of a vertical edge:

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

The steps involved in coding its 16 pixels. A 512-word dictionary with the following starting content is assumed:

Dictionary Location	Entry
0	0
1	1
⋮	⋮
255	255
256	—
⋮	⋮
511	—

Locations 256 through 511 are initially unused.

- A unique feature of the LZW coding just demonstrated is that the coding dictionary or code book is created while the data are being encoded; most practical applications require a strategy for handling dictionary overflow.
- A simple solution is to flush or reinitialize the dictionary when it becomes full and continue coding with a new initialized dictionary.
- A more complex option is to monitor compression performance and flush the dictionary when it becomes poor or unacceptable. Alternately, the least used dictionary entries can be tracked an replaced when necessary.

Bit-Plane Coding:

- Another effective technique for reducing an image's inter pixel redundancies is to process the image's bit planes individually. The technique, called bit-plane coding, is based on the concept of decomposing a multilevel (monochrome or color) image into a series of binary images and compressing each binary image via one of several well-known binary compression methods.

Bit-plane decomposition:

The gray levels of an m-bit gray-scale image can be represented in the form of the base 2 polynomial

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \cdots + a_1 2^1 + a_0 2^0.$$

Based on this property, a simple method of decomposing the image into a collection of binary images is to separate the m coefficients of the polynomial in to m I -bit *bit planes.* As noted in Chapter 3, the zeroth-order bit plane is generated by collecting the *ao* bits of each pixel, while the *(m* - 1)st-order bit plane contains the $a_{m-1,}$ bits or coefficients. In general, each bit plane is numbered from0 to *m* - 1 and is constructed by setting its pixels equal to the values of the appropriate bits or polynomial coefficients from each pixel in the original image. The inherent disadvantage of this approach is that small changes in gray level can have a significant impact on the complexity of the bit planes. If a pixel of intensity 127 (01111111) is adjacent to a pixel of intensity 128 (10000000), for instance, every bit plane will contain a corresponding 0 to 1 (or 1 to 0) transition.

For example, as the most significant bits of the two binary codes for 127 and 128 are different, bit plane 7 will contain a zero-valued pixel next to a pixel of value l, creating a 0 to 1 (or 1 to 0) transition at that point.

An alternative decomposition approach (which reduces the effect of small gray-level variations) is to first represent the image by an In-bit *Gray code.* The m-bit Gray code $g_{m-1}... g_2 g_1 g_0$that corresponds to the polynomial can be computed from

$$g_i = a_i \oplus a_{i+1} \qquad 0 \le i \le m-2$$
$$g_{m-1} = a_{m-1}.$$

Here, ⊕denotes the exclusive OR operation. This code has the unique property that successive code words differ in only one bit position. Thus, small changes in gray level are less likely to affect all *m* bit planes.

Constant area coding:

- A simple but effective method of compressing a binary image or bit plane is to use special code words to identify large areas of contiguous l's or 0's.
- In one such approach, called constant area coding (CAC), the image is divided into blocks of size p X q pixels, which are classified as all white, all black or mixed intensity. The most probable or frequently occurring category is then assigned the I-bit code word 0, and the other two categories are assigned the 2-bit codes 10 and 11.
- When predominantly white text documents are being compressed, a slightly simpler approach is to code the solid white areas as 0 and all other blocks (including the solid black blocks) by a 1 followed by the bit pattern of the block. This approach, called white block skipping.

One-dimensional run-length coding:

An effective alternative to constant area coding is to represent each row of an image or bit plane by a sequence of lengths that describe successive runs of black and white pixels. This technique, referred to as run-length coding.

Two-dimensional run-length coding:

One-dimensional run-length coding concepts are easily extended to create a variety of 2-D coding procedures. One of the better known results is relative address coding (RAC), which is based on the principle of tracking the binary transitions that begin and end each black and white run.

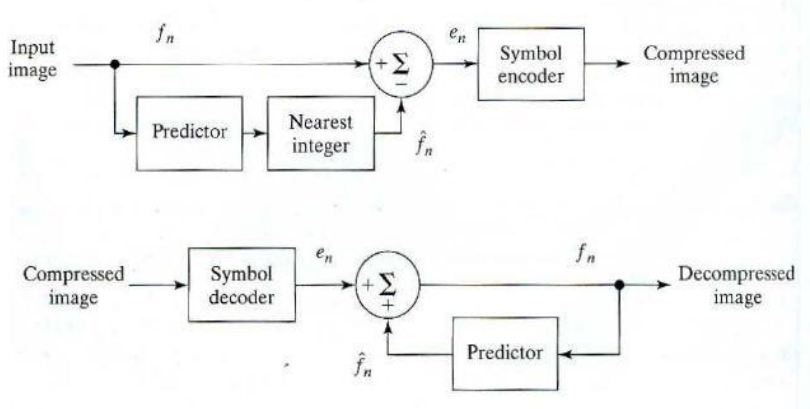
Contour tracing and coding:

Relative address coding is one approach for representing intensity transitions that make up the contours in a binary image. Another approach is to represent each contour by a set of boundary points or by a single boundary point and a set of directional. The latter sometimes is referred to as direct contour tracing. In this section, we describe yet another method, called predictive differential *quantizing* (PDQ), which demonstrates the essential characteristics of both approaches. It is a scan-line-oriented contour tracing procedure.

Lossless Predictive Coding:

- An error-free compression approach that does not require decomposition of an image into a collection of bit planes. The approach, commonly referred to as *lossless predictive coding,* is based on eliminating the inter pixel redundancies of closely spaced pixels by extracting and coding *only* the new information in each pixel.
- The *new information* of a pixel is defined as the difference between the actual and predicted value of that pixel. The basic components of a lossless predictive coding system. The system consists of an encoder and a decoder, each containing an identical *predictor.*
- As each successive pixel of the input image, denoted f_n, is introduced to the encoder, the predictor generates the anticipated value of that pixel based on some number of past inputs. The output of the predictor is then rounded to the nearest integer, denoted f_n ,and used to form the difference or *prediction error.*

$$e_n = f_n - \hat{f}_n,$$



Which is coded using a variable-length code (by the symbol encode r) to generate the next element of the compressed data stream. The decoder of reconstructs *en* from the received variable-length code words and performs the inverse operation

$$f_n = e_n + \hat{f}_n.$$

Various local, global, and adaptive methods can be used to generate f_n. In most cases, however. The prediction is formed by a linear combination of m previous pixels. That is,

$$\hat{f}_n = \text{round}\left[\sum_{i=1}^{m} \alpha_i f_{n-i}\right]$$

Where *m* is the order of the linear predictor, round is a function used to denote the rounding or nearest integer operation, and the α_i for i = 1, 2, ... ,*m* are prediction coefficients.. In I-D linear predictive coding, for example

$$\hat{f}_n(x, y) = \text{round}\left[\sum_{i=1}^{m} \alpha_i f(x, y - i)\right]$$

A similar comment applies to the higher-dimensional cases. Consider encoding the monochrome image of using the simple first-order linear predictor

$$\hat{f}(x, y) = \text{round}[\alpha f(x, y - 1)].$$

- A predictor of this general form commonly is called a previous pixel predictor, and the corresponding predictive coding procedure is referred to as differential coding or previous pixel *coding.*
- The preceding example emphasizes that the amount of compression achieved in loss less predictive coding is related directly to the entropy reduction that results from mapping the input image into the prediction error sequence.
- Because a great deal of inter pixel redundancy is removed by the prediction and differencing process, the probability density function of the prediction error is, in general, highly peaked at zero and characterized by a relatively small (in comparison to the input gray-level distribution) variance. In fact, the density function of the prediction error often is modeled by the zero mean uncorrelated Laplacian pdf

$$p_e(e) = \frac{1}{\sqrt{2}\sigma_e} e^{\frac{-\sqrt{2}|e|}{\sigma_e}}$$

Where (f f' is the standard deviation of *e)*

Detection of Discontinuities:

Several techniques for detecting the three basic types of gray-level discontinuities in a digital image: points, lines, and edges. The most common way to look for discontinuities is to run a mask through the image in the manner described in Section 3.5. For the 3 X 3 mask shown in Fig. 10.1, this procedure involves computing the sum of products of the coefficients with the gray

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Levels contained in the region encompassed by the mask. That is, with reference to the response of the mask at any point in the image is given by

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9$$
$$= \sum_{i=1}^{9} w_i z_i$$

Where *Zi* is the gray level of the pixel associated with mask coefficient *Wi* .As usual, the response of the mask is defined with respect to its center location. The details for implementing mask operations.

Point Detection:

The detection of isolated points in an image is straightforward in principle. Using the mask shown We say that a point has been detected at the location on which the mask is centered if

$$|R| \geq T$$

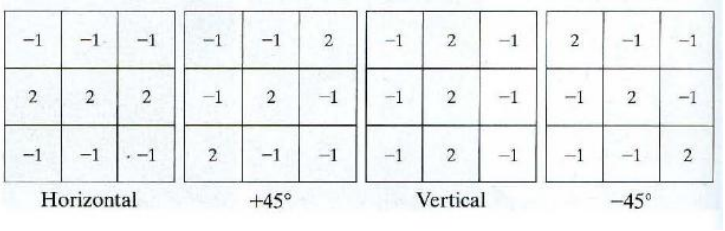
Where*T* is a nonnegative threshold and *R* is given .Basically this formulation measures the weighted differences between the center point and its neighbors. The idea is that an *isolated* point (a point whose gray level is significantly different from its background and which is located in a homogeneous or nearly homogeneous area) will be quite different from its surroundings, and thus be easily detectable by this type of mask. Only differences that are considered of interest are those

−1	−1	−1
−1	8	−1
−1	−1	−1

Large enough (as determined by *T)* to be considered isolated points.

Line Detection:

- The next level of complexity is line detection. Consider the masks. If the first mask were moved around an image, it would respond more strongly to lines (one pixel thick) oriented horizontally.
- With a constant background, the maximum response would result when the line passed through the middle row of the mask. This is easily verified by sketching a simple array of l's with a line of a different gray level (say, 5's) running horizontally through the array. A similar experiment would reveal that the second mask .responds best to lines oriented at +45°; the third mask to vertical lines; and the fourth mask to lines in the -45° direction. These directions can be established also by noting that the preferred direction of each mask is weighted with a larger coefficient (i.e., 2) than other possible directions.
- Note that the coefficients in each mask sum to zero, indicating a zero response from the masks in areas of constant gray level.
- Let *R" R" R3* ,and *R4* denote the responses of the masks. If, at a certain point in the image,$|R_i| > |R_j|$, for all $j \neq i$, that point is said to be more likely associated with a line in the direction of mask i. For example, if at a point in the image, $|R_1| > |R_j|$, for



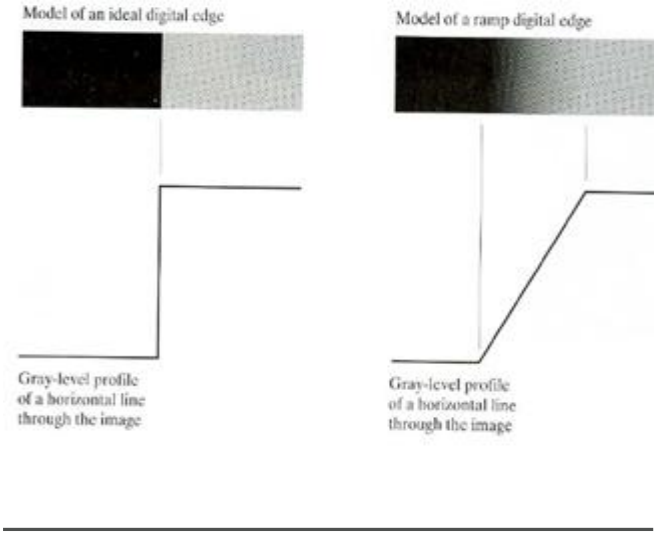
$j = 2, 3, 4$. That particular point is said to be more likely associated with a horizontal line.

Edge Detection:

Although point and line detection certainly are important in any discussion on segmentation, edge detection is by far the most common approach for detecting meaningful discontinuities in gray level.

Basic formulation:

An edge is a set of connected pixels that lie on the boundary between two regions. However, we already went through some length .We start by modeling an edge intuitively. This will lead us to a formalism in which "meaningful" transitions in gray levels can be measured. Intuitively, an ideal edge has the properties of the model.



Gradient operators:

First-order derivatives of a digital image are based on various approximations of the 2-D gradient. The gradient of an image *f(x, y)* at location *(x , y)*is defined as the *vector*

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}.$$

It is well known from vector analysis that the gradient vector points in the direction of maximum rate of change of **f** at coordinates *(x, y).*An important quantity in edge detection is the magnitude of this vector, denoted V *f,* where

$$\nabla f = \text{mag}(\nabla \mathbf{f}) = \left[G_x^2 + G_y^2\right]^{1/2}.$$

This quantity gives the maximum rate of increase of *f(x, y)* per unit distance in the direction of ∇f .It is a common (although not strictly correct) practice to refer to ∇f also as the *gradient.* The *direction* of the gradient vector also is an important quantity. Let *a(x ,y)* represent the direction angle of the vector ∇f at *(x , y).* Then, from vector analysis,

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

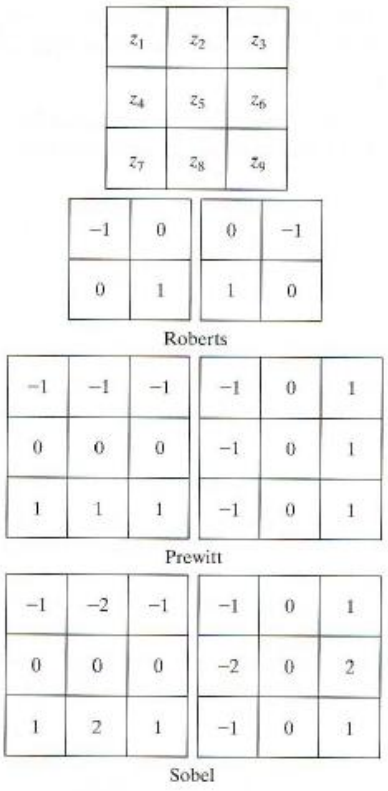
One of the simplest ways to implement a first-order partial derivative at point $Z_5$is to use the following Roberts cross-gradient operators

$$G_x = (z_9 - z_5)$$

$$G_y = (z_8 - z_6).$$

Masks of size 2 X 2 are awkward to implement because they do not have a clear center. An approach using masks of size 3 X 3 is given by

$$G_x = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$



The difference between the first and third rows of the 3 X 3 image region approximates the derivative in the x-direction, and the difference between the third and first columns approximates the derivative in the y-direction. The masks called the Prewitt operators, can be used to implement these two' equations.
A slight variation of these two equations uses a weight of 2 in the center coefficient:

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7).$$

An approach used frequently is to approximate the gradient by absolute values:

$$\nabla f \approx |G_x| + |G_y|.$$

The Laplacian

The Laplacian ofa 2-D function *f(x, y)* is a second-order derivative defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}.$$

Digital approximations to the Laplacian were introduced in Section 3.7.2. For a 3 X 3 region, one of the two forms encountered most frequently in practice is

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8)$$

0	−1	0	−1	−1	−1
−1	4	−1	−1	8	−1
0	−1	0	−1	−1	−1

Where the z's are defined .A digital approximation including the diagonal neighbors is given by

$$\nabla^2 f = 8z5 - (z, + z, + Z3 + z, + <6 + <7 + Z8 + z,).$$

In the first category, the Laplacian is combined with smoothing as a precursor to finding edges via zero-crossings. Consider the function

$$h(r) = -e^{-\frac{r^2}{2\sigma^2}}$$

Where $r' = x' + y'$ and *(J* is the standard deviation. Convolving this function with an image blurs the image, with the degree of blurring being determined by the value of *(J.* The Laplacian of *h* (the second derivative of *h* with respect to *r)* is

$$\nabla^2 h(r) = -\left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}.$$

Edge Linking and Boundary Detection:

This set of pixels seldom characterizes an edge completely because of noise, breaks in the edge from non-uniform illumination, and other effects that introduce spurious intensity discontinuities. Thus edge detection algorithms typically are followed by linking procedures to assemble edge pixels into meaningful edges.

Local Processing:

- One of the simplest approaches for linking edge points is to analyze the characteristics of pixels in a small neighborhood (say, 3 X 3 or 5 X 5) about every point *(x, y)* in an image that has been labeled an edge point by one of the techniques.
- All points that is similar according to a set of Predefined criteria are linked, forming an edge of pixels that share those criteria.
- The two principal properties used for establishing similarity of edge pixels in this kind of analysis are

  (1) The strength of the response of the gradient operator used to produce the edge pixel; (2) the direction of the gradient vector. The first property is given by the value of ∇f , as defined Thus an edge pixel with coordinates *(xo, yo)* in a predefined neighborhood of*(x, y)* , is similar in magnitude to the pixel at *(x, y)* if

$$|\nabla f(x, y) - \nabla f(x_0, y_0)| \leq E$$

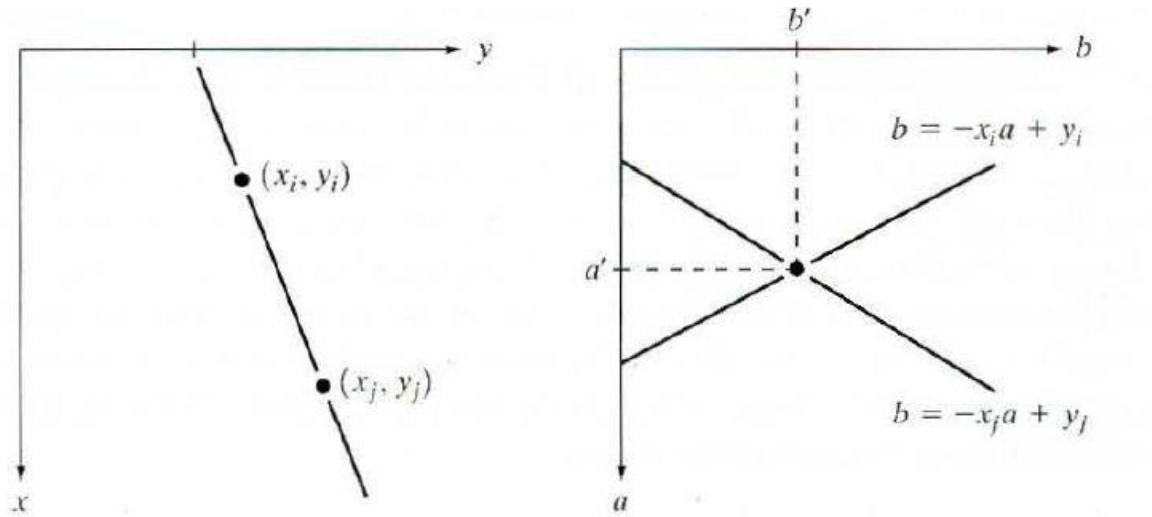
Where *E* is a non negative Threshold. The direction (angle) of the gradient vector is given. An edge pixel at (xo, *yo)* in the predefined neighborhood of *(x, y)* has an angle similar to the pixel at *(x. y)* if

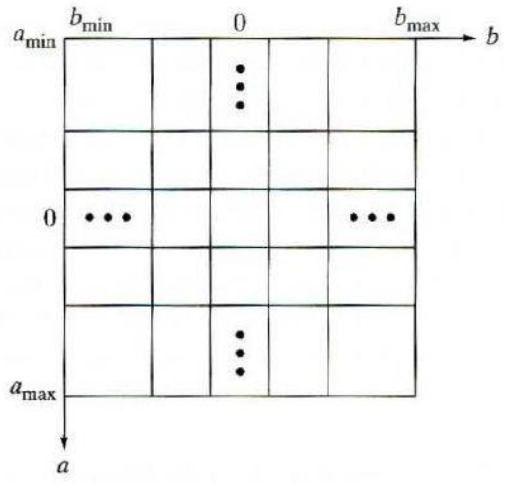
$$|\alpha(x, y) - \alpha(x_0, y_0)| < A$$

Where A is a non negative angle threshold. The direction of the edge at *(x, y)* is *perpendicular* to the direction of the gradient vector at that point.

Global Processing via the Hough Transform

- Points are linked by determining first if they lie on a curve of specified shape. Unlike the local analysis method we now consider global relationships between pixels.
- One possible solution is to first find all lines determined by every pair of points and then find all subsets of points that are close to particular lines.
- The problem with this procedure is that it involves finding *n (n* - 1)/2 - n' lines and then performing *(n) (n(n* - 1))/2 - *n3* comparisons of every point to all lines.
- This approach is computationally prohibitive in all but the most trivial applications. Consider a point $x_i, y_i)$ and the general equation of a straight line in slope-intercept form, $y_i = ax_i + b$. In finitely many lines pass through (x_i, y_i), but they all satisfy the equation $yi = ax_i + b$ for varying values of *a* and *b.* However, writing this equation as $b = -x_i a + y_i$ and considering the ab-plane (also called *parameter space)* yields the equation of a *single* line for a fixed pair$(x_i. y_i)$· Furthermore second point $(xj. y_i)$ also has a line in parameter space associated with it, and this line intersects the line associated with (xj,y_i) at (a', b'),where a' is the slope and b' the intercept of the line containing both $(xj. y_i)$ and$(xj. y_i)$ in the xy-plane. In fact, all points contained on this line have lines in parameter space that intersect at (a', b').



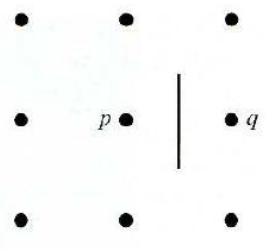


The computational attractiveness of the Hough transform arises from subdividing the parameter space into so-called *accumulator cells,* as illustrated, where (a_{max},a_{min}) and (b_{max},b_{min}) are the expected ranges of slope and intercept values. The cell at coordinates *(i, j),* with accumulator value *A(i, j),*corresponds to the square associated with parameter space coordinate *(a_i,b_j)*.Initially, these cells are set to zero. A problem with using the equation $Y = ax + b$ to represent a line is that the slope approaches infinity as the line approaches the vertical. One way around this difficulty is to use the normal representation of a line:

$$x\cos\theta + y\sin\theta = p.$$

Global Processing via Graph-Theoretic Techniques

A global approach for edge detection and linking based on representing edge segments in the form of a graph and searching the graph for low-cost paths that correspond to significant edges. This representation providesa rugged approach that performs well in the presence of noise. As might be expected, the procedure is considerably more complicated and requires more processing time than

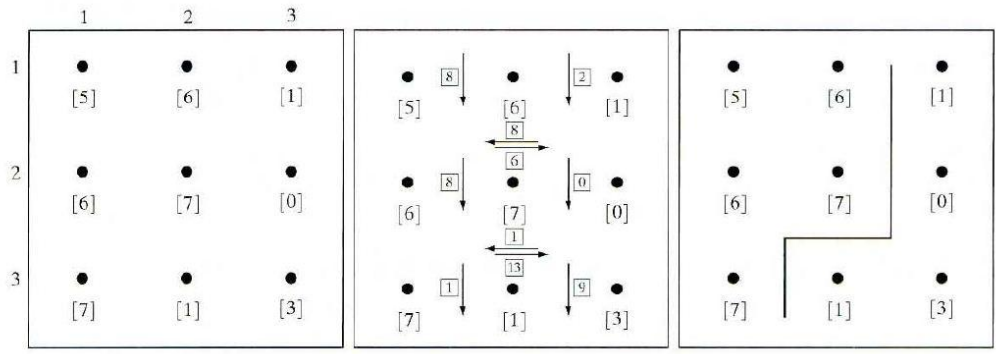


We begin the development with some basic definitions. A *graph* G= *(N , U)* is a finite, nonempty set of nodes *N,* together with a set *U* of unordered pairs of distinct elements of *N.* Each pair *(n_i,n_J) ofU* is called an *arc. A* graph in which the arcs are directed is called a *directed graph*

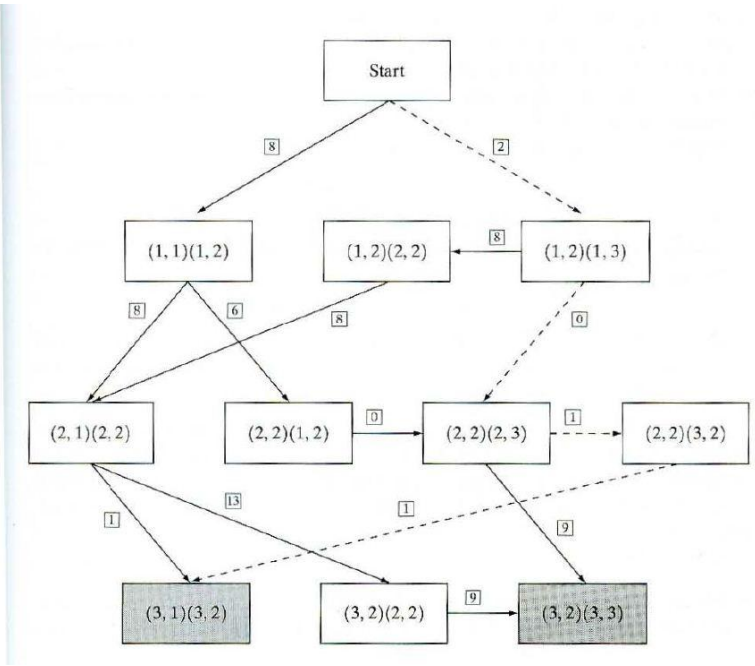
$$c = \sum_{i=2}^{k} c(n_{i-1}, n_i).$$

The following discussion is simplified if we define an edge element as the boundary between two pixels *p* and *q,* such that *p* and *q* are 4·neighbors, Edge elements are identified by the xy-coordinates of points *p*

and *q*. In other words, the edge element in Fig. 10.22 is defined by the pairs$(x_p, Y_p)(X_q, Y_q)$. Consistent with the definition given in Section 10.1.3, an *edge*is a sequence of connected edge elements.



- Where H is the highest gray-level value in the image (7 in this case), and *f(p)*and *f(q)* are the gray-level values of *p* and *q,* respectively. By convention, the point *p* is on the right-hand side of the direction of travel along edge elements. For example, the edge segment (1,2)(2, 2) is between points (1,2) and (2,2) If the direction of travel is to the right, then *p* is the point with coordinates (2, 2) and *q* is point with coordinates (1 , 2); therefore, $c(p, q) = 7 - [7 - 6] = 6$.
- This cost is shown in the box below the edge segment. If, on the other hand, we are traveling to the *left* between the same two points, then *p* is point (1, 2) and *q* is (2,2). In this case the cost is 8, as shown above the edge segment.
- To simplify the discussion, we assume that edges start in the top row and terminate in the last row, so that the first element of an edge can be only between points (1 ,1), (1, 2) or (1 , 2),(1, 3). Similarly, the last edge element has to be between points (3, 1), (3,2)or (3,2), (3,3). Keep in mind that *p* and *q* are 4-neighbors, as noted earlier.



Step 1: Mark the start node OPEN and set g(s) = 0.

Step 2: If no node is OPEN exit with failure; otherwise, continue.

Step 3: Mark CLOSED the OPEN node n whose estimate r(n) computed from Eg. (10.2·7) is smallest. (Ties for minimum r values are resolved arbitrarily, but always in favor of a goal node.)

Step 4: If n is a goal node, exit with the solution path obtained by tracing back through the pointers; otherwise, continue.

Step 5: Expand node n, generating all of its successors. (If there are no successors go to step 2.)

Step 6: If a successor n; is not marked, set mark it OPEN, and direct pointers from it back to n.

$$r(n_i) = g(n) + c(n, n_i),$$

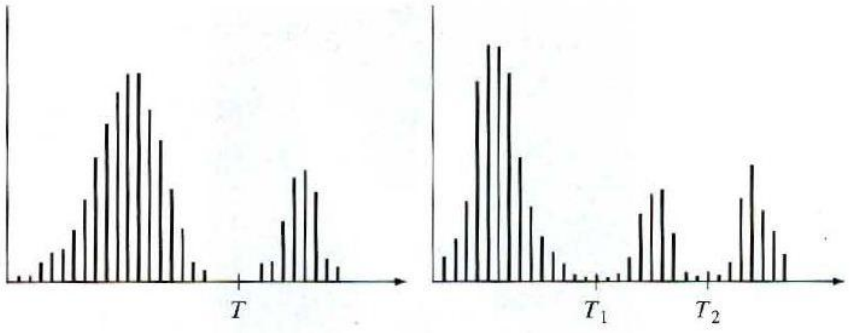
Step 7: If a successor 11; is marked CLOSED or OPEN, update its value by Letting

$$g'(n_i) = \min[g(n_i), g(n) + c(n, n_i)].$$

Mark OPEN those CLOSED successors whose g' values were thus lowered and redirect to n the pointers from all nodes whose g' values were lowered. Go to step 2.

Thresholding: (APR 2012)

- Thresholding in a more formal way and extend it to techniques that are considerably more general Suppose that the gray- level histogram corresponds to an image ,f(x, y), composed of light objects on a dark background, in such a way that object and background pixels have gray levels grouped into two dominant modes.
- One obvious way to extract the objects from the background is 10 select a threshold that separates these modes. Then any point (x, y) for which f(x, y) >T is called an object point; otherwise, the point is called a background point. This is the type of thresholding



Based on the preceding discussion, thresholding may be viewed as an operation that involves tests against a function T of the form

$$T = T[x, y, p(x, y), f(x, y)]$$

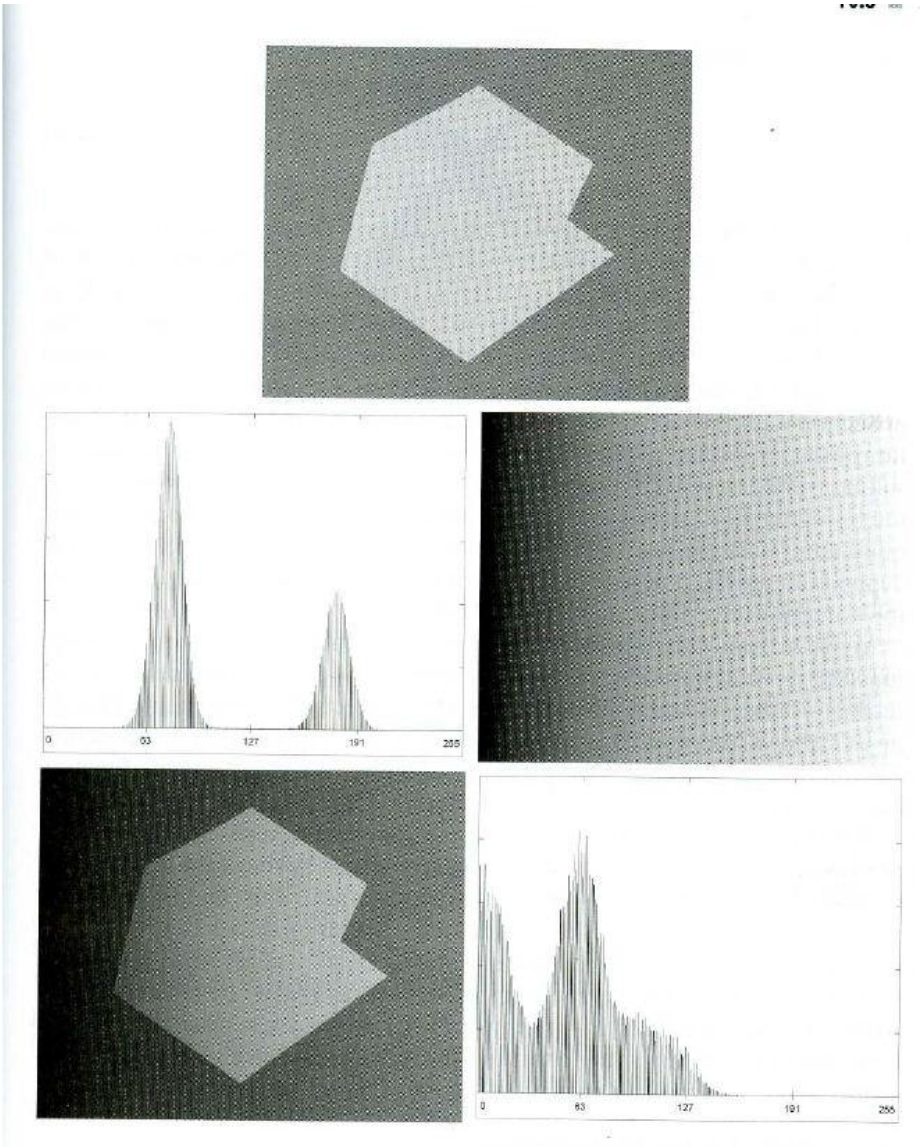
where J(x, y) is the gray level of point (x, y) and p(x, y) denotes some local property of this point- for example, the average gray level of a neighborhood centered on (x, y). A thresholded image g(x, y) is defined as

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T. \end{cases}$$

Thus, pixels labeled 1 (or any other convenient gray level) correspond to objects, whereas pixels labeled 0 (or any other gray level not assigned to Objects) correspond to the back ground. When T depends only on J(x,y) (that is, only on gray-level values) the threshold is called global. If T depends on both *J(x, y)* and *p(x, y),*the thresholdiing called *local.* If, in addition, *T* depends on the spatial coordinates *x* and *y,* the threshold is called *dynamic* or *adaptive.*

The Role of Illumination

A simple model in which an image *J(x ,y)* is formed as the product of a reflectance component *r(x, y)* and an illumination component *i(x, y).* The purpose of this section is to use this model to discuss briefly the effect of illumination on thresholding, especially on global thresholding.



$$f(x, y) = i(x, y)r(x, y).$$

Taking the natural logarithm of this equation yields a sum:

$$\begin{aligned} z(x, y) &= \ln f(x, y) \\ &= \ln i(x, y) + \ln r(x, y) \\ &= i'(x, y) + r'(x, y). \end{aligned}$$

Basic Global Thresholding:

- The simplest of all thresholding techniques is to partition the image histogram by using a single global threshold, *T,* as illustrated.
- Segmentation is then accomplished by scanning the image pixel by pixel and labeling each pixel as object or background, depending on whether the gray level of that pixel is greater or less than the value of *T.* The following algorithm can be used to obtain *T* automatically:

1.Select an initial estimate for *T.*

2. Segment the image using *T.* This will produce two groups of pixels: G, consisting of all pixels with gray level values >*T* and G2 consisting of pixels with values ~ *T.*

3. Compute the average gray level values 1-' ,and 1-'2 for the pixels in regions

$G_1$and G_2.

4. Compute a new threshold value:

$$T = \frac{1}{2}(\mu_1 + \mu_2).$$

5. Repeat steps 2 through 4 until the difference in *T* in successive iterations is smaller than a predefined parameter *To.*

Basic Adaptive Thresholding

All sub images containing boundaries had variances in excess of 100. Each sub image with variance greater than 100 was segmented with a threshold computed for that sub image using the algorithm discussed in the previous section. The initial value for *T* in each case was selected as the point midway between the minimum and maximum gray levels in the sub image. All subimages with variance

less than 100 were treated as one composite image, which was segmented using a single threshold estimated using the same algorithm.

Optimal Global and Adaptive Thresholding

- The method is applied to a problem that requires sol1ltion of several important issues found frequently in the practical application of thresholding. Suppose that an image contains only two principal gray- level regions. Let *z* denote gray-level values.
- We can view these values as random quantities, and their histogram may be considered an estimate of their probability density function(PDF), *p(z).* This overall density function is the sum or mixture of two densities, one for the light and the other for the dark regions in the image.
- Furthermore, the mixture parameters are proportional to the relative areas of the dark and light regions. If the form of the densities is known or assumed, it is possible to determine an optimal threshold (in terms of minimum error) for segmenting the image into the two distinct regions. Assume that the larger of the two PDFs corresponds to the background levels while the smaller one describes the gray levels of objects in the image. The mixture probability density function describing the overall gray-level variation in the image is

$$p(z) = P_1 p_1(z) + P_2 p_2(z).$$

Here, *P,* and *P,* are the probabilities of occurrence of the two classes of pixels; that is, *P,* is the probability (a number) that a random pixel with value *z* is an object pixel. Similarly, *P,* is the probability that the pixel is a background pixel. We are assuming that any given pixel belongs either to an object or to the background, so that

$$P_1 + P_2 = 1.$$

Thus, the probability of *erroneously* classifying a background point as an object point is

$$E_1(T) = \int_{-\infty}^{T} p_2(z)\, dz.$$

This is the area under the curve of *p,(z)* to the left of the threshold. Similarly, the probability of erroneously classifying an object point as background is

$$E_2(T) = \int_{T}^{\infty} p_1(z)\, dz,$$

Which is the area under the curve of *p,(z)* to the right of *T.* 111en the overall probability of error is

$$E(T) = P_2 E_1(T) + P_1 E_2(T).$$

Note how the quantities $E_1$and $E_2$are weighted (given importance) by the probability of occurrence of object or background pixels. Note also that the sub-scripts are opposites

To find the threshold value for which this error is minimal requires differentiating *E(T)* with respect to *T* (using Leibniz's rule) and equating the result to 0. The result is

$$P_1 p_1(T) = P_2 p_2(T).$$

One of the principal densities used in this manner is the Gaussian density, which is completely characterized by two parameters: the mean and the variance. In this case

$$p(z) = \frac{P_1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(z-\mu_1)^2}{2\sigma_1^2}} + \frac{P_2}{\sqrt{2\pi}\sigma_2} e^{-\frac{(z-\mu_2)^2}{2\sigma_2^2}}$$

Where$\mu_1$and *cr T* are the mean and variance of the Gaussian density of one class of pixels (say, objects) and μ_2. And O'j are the mean and variance of the other class. Using this equation in the general solution

$$AT^2 + BT + C = 0$$

$$A = \sigma_1^2 - \sigma_2^2$$
$$B = 2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2)$$
$$C = \sigma_1^2\mu_2^2 - \sigma_2^2\mu_1^2 + 2\sigma_1^2 2\sigma_2^2 \ln(\sigma_2 P_1/\sigma_1 P_2).$$

Since a quadratic equation has two possible solutions, two threshold values may be required to obtain the optimal solution. Lf the variances are equal, *(T2 = O"T = uL* a single threshold is sufficient:

$$T = \frac{\mu_1 + \mu_2}{2} + \frac{\sigma^2}{\mu_1 - \mu_2} \ln\left(\frac{P_2}{P_1}\right).$$

For example, the mean square error between the (continues) mixture density *p(z)* and the (discrete) image histogram *h(Zi)* is

$$e_{\text{ms}} = \frac{1}{n} \sum_{i=1}^{n} [p(z_i) - h(z_i)]^2$$

The chances of selecting a "good" threshold are enhanced considerably if the histogram peaks are tall, narrow, symmetric, and separated by deep valleys. One approach for improving the shape of histograms is to consider only those pixels that lie on or near the edges between objects and the background. Thresholds Based on Several Variables

A sensor can make available more than one variable to characterize each pixel in an image, and thus allow multispectral thresholding.

Region-Based Segmentation

- This problem by finding boundaries between regions based on discontinuities in gray levels, where as segmentation was accomplished via thresholds based on the distribution of pixel properties,

such as gray-level values or color. In this section we discuss segmentation techniques that are based on finding the regions directly. That partitions *R* into *n* sub regions, $R_1R_2 \ldots R_n$ such that

(a) $\bigcup_{i=1}^{n} R_i = R.$
(b) R_i is a connected region, $i = 1, 2, \ldots, n.$
(c) $R_i \cap R_j = \varnothing$ for all i and $j, i \neq j.$
(d) $P(R_i) =$ TRUE for $i = 1, 2, \ldots, n.$
(e) $P(R_i \cup R_j) =$ FALSE for $i \neq j.$

Here, *p(Ri)* is a logical predicate defined over the points in set *Ri* and 0 is the null set.

Condition (a) :indicates that the segmentation must be complete; that is, every pixel must be in a region.

Condition (b): requires that points in a region must be connected in some predefined sense **Condition (c) :** indicates that the regions must be disjoint.

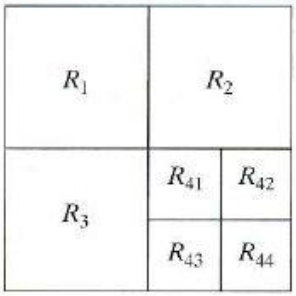
Condition(d):deals with the properties that must be satisfied by the pixels in a segmented region-for example *(*RJ = TRUE if a li pixels in *R,* have the same gray level. finally,

Condition(e): Indicates that regions *R,* and *Rj* are different in the sense of predicate *P.*

Region Growing

- *Region growing* is a procedure that group's pixels or sub regions into larger regions based on predefined criteria.111e basic approach is to start with a set of "seed" points and from these grow regions by appending to each seed those neighboring pixels that have properties similar to the seed (such as specific ranges of gray level or color).
- Selecting a set of one or more starting points often can be based on the nature of the problem, as. When a priori information is not available, the procedure is to compute at every pixel the same set of properties that ultimately will be used to assign pixels to regions during the growing process.
- If the result of these computations shows clusters of values, the pixels whose properties place them near the centroid of these clusters can be used as seeds. This location of similarity criteria depends not only on the problem under consideration, but also on the type of image data available.
- For example, the analysis of land-use satellite imagery depends heavily on the use of color. This problem would be significantly more difficult, or even impossible, visualize a random arrangement of pixels with only three distinct gray-level values. Grouping pixels with the same gray level to form a "region" without paying attention to connectivity would yield a segmentation result that is meaningless in the context
- Another problem in region growing is the formulation of a stopping rule. Basically, growing a region should stop when no more pixels satisfy the criteria for inclusion in that region. Criteria such as gray level, texture, and color, are local in nature and do not take into account the "history" of region growth.
- An alternative is to subdivide an image initially into a set of arbitrary, disjointed regions and then merge and or split the regions in an attempt to satisfy the conditions stated

Let *R* represent the entire image region and select a predicate *P*. One approach for segmenting *R* is to subdivide it successively into smaller and smaller quadrant regions so that, for any region *Ri* ,*p(Ri)* = TRUE. We start with the entire region. If *P(R)* = FALSE, we divide the image into quadrants. If *P* is FALSE for any quadrant, we subdivide that quadrant into sub quadrants, and soon. This particular splitting technique has a convenient representation in the form of a so-called *quad tree*(that is, a tree in which nodes have exactly four descendants).



1. Split into [our disjoint quadrants any region $R_;$ for which $p(R;)$ = FALSE.
2. Merge any adjacent regions R ,and $R,$ for which $P(R_j U R,)$ = TRUE.
3. Stop when no further merging or splitting is possible.

Segmentation by Morphological Watersheds:

- Segmentation based on three principal concepts :(a) detection of discontinuities, (b) thresholding, and (c) region processing. Each of these approaches was found to have advantages (for example, speed in the case of global thresholding) and disadvantages (for example, the need for post processing, such as edge linking, in methods based on detecting discontinuities in gray levels).
- The concept of watersheds is based on visualizing an image in three dimensions: two spatial coordinates versus gray levels.
- In such a " topographic" interpretation, we consider three types of points:

  (a) Points belonging to a regional minimum;

  (b) Points at which a drop of water, if placed at the location of any of those points, would fall with certainty to a single minimum;

  (c) Points at which water would be equally likely to rail to more than one such minimum. For a particular regional minimum, the set of points satisfying condition (b) is called the *catchment basin* or *watershed* of that minimum. The point's satisfying condition(c) Form crest lines on the topographic surface and are termed *divide lines* or *watershed lines.*
- The basic idea is simple: Suppose that a hole is punched in each regional minimum and that the entire topography is flooded from below by letting water rise through the holes at a uniform rate.
- The flooding will eventually reach a stage when only the tops of the dams are visible above the water line. These dam boundaries correspond to the divide lines of the watersheds. Therefore, they are the (continuous) boundaries extracted by a watershed segmentation algorithm.

- Three-dimensional representation is of interest. In order to prevent the rising water from spilling out through the edges of the structure, we imagine the perimeter of the entire topography (image) being enclosed by dams of height greater than the highest possible mountain, whose value is determined by the highest possible gray-level value in the input image.
- Suppose that a hole is punched in each regional minimum and that the entire topography is flooded by letting water rise through the holes at a uniform rate. Figure 1O.44(c) shows the first stage of flooding, where the "water," shown in light gray, has covered only areas that correspond to the very dark background in the image.
- That the water now has risen into the first and second catchment basins, respectively. As the water continues to rise, it will eventually overflow from one catchment basin into another. The first indication Here, water from the left basin actually overflowed into the basin on the right and a short "dam " (consisting of single pixels) was built to prevent water from merging at that level of flooding (the details of dam building are disclosed in the following section). The effect is more pronounced as water continues to rise, as shown
- The latter dam was built to prevent merging of Water from that basin with water from areas corresponding to the background. This process is continued until the maximum level of flooding (corresponding to the highest gray- level value in the image) is reached. The final dams correspond to the watershed lines. Which are the desired segmentation results.
- Before proceeding, let us consider how to construct the dams or watershed lines required by watershed segmentation algorithms. Dam construction is based on binary images, which are members of 2-D integer space z^2The simplest way to construct dams separating sets of binary points is to use morphological dilation.

PONDICHERRY UNIVERSITY QUESTIONS

GRAPHICS AND IMAGE PROCESSING

1. Write any four Morphology Algorithm. **(APRIL / MAY 2012)**
2. Discuss about Thresholding. **(APR 2012) (Pg.no.22)**
3. What are two fundamental operations in Morphology? Explain. **(NOVEMBER 2012)**
4. Explain the general image compression model. **(NOVEMBER 2012) (APR 2014)**
5. Briefly explain Elements of Information theory. **(APRIL 2013)**
6. Discuss the different techniques in Detection of Discontinuities. **(APRIL 2013)**
7. Explain "Huffman Encoding Techniques" in detail. **(NOVEMBER 2013)**
8. Describe the various gray-scale morphological operations in detail with an example for each. **(NOVEMBER 2013)**
9. Describe the techniques based on gray level discontinuity. **(APR 2014)**