SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

UNIT – I Digital Image Processing for Real Time Applications – SEC1628

I. Digital Image Fundamentals

1

Image acquisition and storage; Basic Relationships between Pixels; Monochromatic Vision Models; Colour Vision Models; Colour Fundamentals; Colour Models; Image resizing; Image noise- additive and multiplicative noises; Image quality indicators (Quality Metrics)- PSNR, SSIM, VIF, accuracy, Correlation. Case study: Develop a software program to measure the quality of the given image.

Elements of Visual Perception

Although the field of digital image processing is built on a foundation of mathematical and probabilistic formulations, human intuition and analysis play a central role in the choice of one technique versus another, and this choice often is made based on subjective, visual judgments. Hence, developing a basic understanding of human visual perception is necessary. In particular, our interest is in the mechanics and parameters related to how images are formed and perceived by humans.

Structure of Human eye

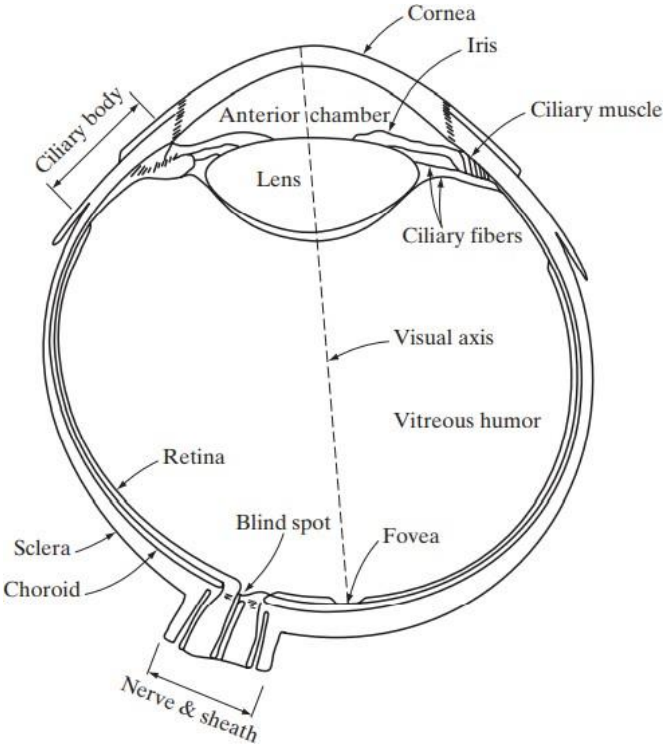


Fig. 1.1 Simplified diagram of a cross section of the human eye.

Figure 1.1 shows a simplified horizontal cross section of the human eye. The eye is nearly a sphere, with an average diameter of approximately 20 mm. Three membranes enclose the eye: the cornea and sclera outer cover; the choroid; and the retina. The cornea is a tough, transparent tissue that covers the anterior surface of the eye. Continuous with the cornea, the sclera is an opaque membrane that encloses the remainder of the optic globe. The choroid lies directly below the sclera. This membrane contains a network of blood vessels that serve as the major source of nutrition to the eye. Even superficial injury to the choroid, often not deemed serious, can lead to severe eye damage as a result of inflammation that restricts blood

flow. The choroid coat is heavily pigmented and hence helps to reduce the amount of extraneous light entering the eye and the backscatter within the optic globe. At its anterior extreme, the choroid is divided into the ciliary body and the iris. The latter contracts or expands to control the amount of light that enters the eye. The central opening of the iris (the pupil) varies in diameter from approximately 2 to 8 mm. The front of the iris contains the visible pigment of the eye, whereas the back contains a black pigment. The lens is made up of concentric layers of fibrous cells and is suspended by fibers that attach to the ciliary body. It contains 60 to 70% water, about 6% fat, and more protein than any other tissue in the eye. The lens is colored by a slightly yellow pigmentation that increases with age. In extreme cases, excessive clouding of the lens, caused by the affliction commonly referred to as cataracts, can lead to poor color discrimination and loss of clear vision. The lens absorbs approximately 8% of the visible light spectrum, with relatively higher absorption at shorter wavelengths. Both infrared and ultraviolet light are absorbed appreciably by proteins with in the lens structure and, in excessive amounts, can damage the eye.

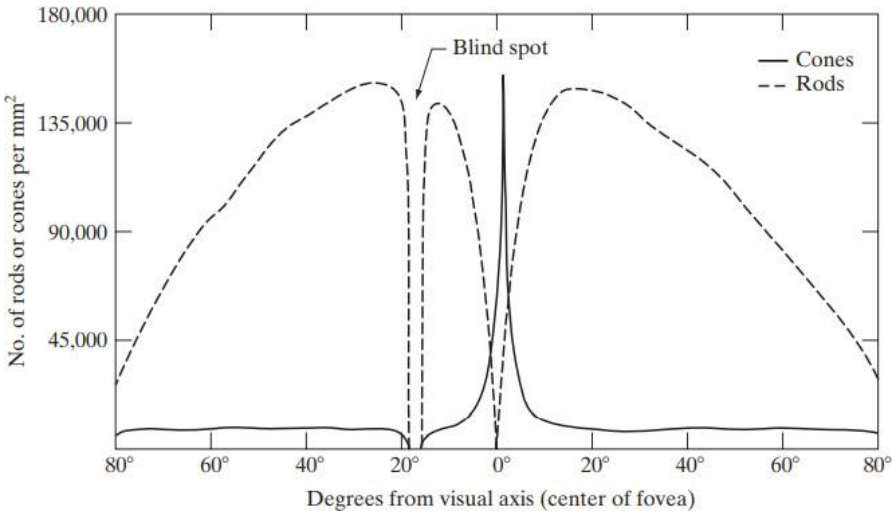


Fig. 1.2 Distribution of rods and cones in the retina.

The innermost membrane of the eye is the retina, which lines the inside of the wall's entire posterior portion. When the eye is properly focused, light from an object outside the eye is imaged on the retina. Pattern vision is afforded by the distribution of discrete light receptors over the surface of the retina. There are two classes of receptors: cones and rods. The cones in each eye number between 6 and 7 million. They are located primarily in the central portion of the retina, called the fovea, and are highly sensitive to color. Humans can resolve fine details with these cones largely because each one is connected to its own nerve end. Muscles controlling the eye rotate the eyeball until the image of an object of interest falls on the fovea. Cone vision is called photopic or bright-light vision. The number of rods is much larger: Some 75 to 150 million are distributed over the retinal surface. The larger area of distribution and the fact that several rods are connected to a single nerve end reduce the amount of detail discernible by these receptors. Rods serve to give a general, overall picture of the field of view. They are not involved in color vision and are sensitive to low levels of illumination. For example, objects that appear brightly colored in daylight when seen by moonlight appear as colorless forms because only the rods are stimulated. This phenomenon is known as scotopic or dim-light vision. Figure 1.2 shows the density of rods and cones for a cross

section of the right eye passing through the region of emergence of the optic nerve f rom the eye. The absence of receptors in this area results in the so-called blind spot. Except f or this region, the distribution of receptors is radially symmetric about the fovea. Receptor density is measured in degrees from the fovea (that is, in degrees off axis, as measured by the angle formed by the visual axis and a line passing through the center of the lens and intersecting the retina). Note in Fig.1.2 that cones are most dense in the center of the retina (in the center area of the fovea). Note also that rods increase in density from the center out to approximately 20° off axis and then decrease in density out to the extreme periphery of the retina. The fovea itself is a circular indentation in the retina of about 1.5 mm in diameter.

Image Formation in the Eye

In an ordinary photographic camera, the lens has a fixed focal length, and focusing at various distances is achieved by varying the distance between the lens and the imaging plane, where the film (or imaging chip in the case of a digital camera) is located. In the human eye, the converse is true; the distance between the lens and the imaging region (the retina) is fixed, and the focal length needed to achieve proper focus is obtained by varying the shape of the lens. The fibers in the ciliary body accomplish this, flattening or thickening the lens for distant or near objects, respectively. The distance between the center of the lens and the retina along the visual axis is approximately 17 mm. The range of focal lengths is approximately 14 mm to 17 mm, the latter taking place when the eye is relaxed and focused at distances greater than about 3 m. The geometry in Fig. 1.3 illustrates how to obtain the dimensions of an image formed on the retina. For example, suppose that a person is looking at a tree 15 m high at a distance of 100 m. Letting h denote the height of that object in the retinal image, the geometry of Fig. 1.3 yields or As indicated, the retinal image is focused primarily on the region of the fovea. Perception then takes place by the relative excitation of light receptors, which transform radiant energy into electrical impulses that ultimately are decoded by the brain.

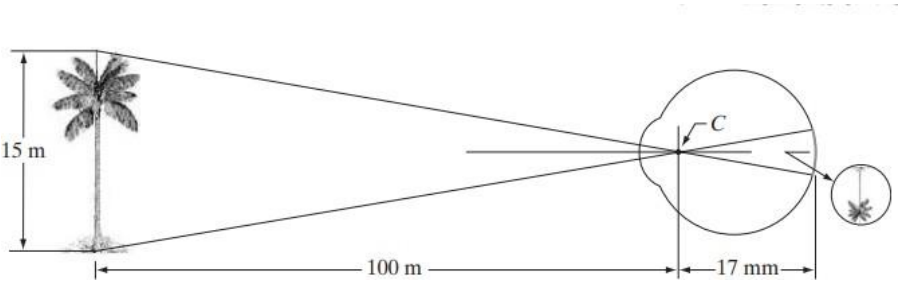


Fig. 1.3 Graphical representation of the eye looking at a palm tree. Point C is the optical center of the lens

Brightness Adaptation and Discrimination

Because digital images are displayed as a discrete set of intensities, the eye's ability to discriminate between different intensity levels is an important consideration in presenting image processing results. The range of light intensity levels to which the human visual

system can adapt is enormous—on the order of — from the scotopic threshold to the glare limit. Experimental evidence indicates that subjective brightness (intensity as perceived by the human visual system) is a logarithmic function of the light intensity incident on the eye.

Image Sensing and Acquisition

Most of the images in which we are interested 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. We enclose illumination and scene in quotes to emphasize the fact that they are considerably more general than the familiar situation in which a visible light source illuminates a common everyday 3-D (three-dimensional) scene. For example, the illumination may originate from a source of electromagnetic energy such as radar, infrared, or X-ray system. 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. An example in the first category is light reflected from a planar surface. An example in the second category is when X-rays pass through a patient's body for the purpose of generating a diagnostic X-ray film. In some applications, the reflected or transmitted energy is focused onto a photoconverter (e.g., a phosphor screen), which converts the energy into visible light. Electron microscopy and some applications of gamma imaging use this approach. Figure 1.4 shows the three principal sensor arrangements used to transform illumination energy into digital images. The idea is simple: Incoming energy is transformed into a voltage by the combination of input electrical power and sensor material that is responsive to the particular type of energy being detected. The output voltage waveform is the response of the sensor(s), and a digital quantity is obtained from each sensor by digitizing its response. In this section, we look at the principal modalities for image sensing and generation.

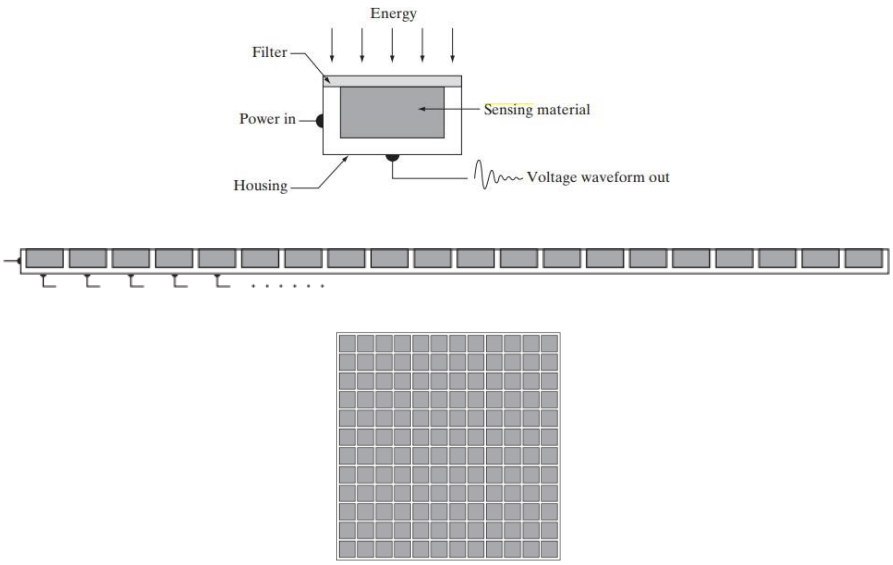


Fig. 1.4 (a) Single imaging sensor. (b) Line sensor. (c) Array sensor.

Image Acquisition Using a Single Sensor

Figure 1.4 (a) shows the components of a single sensor. Perhaps the most familiar sensor of this type is the photodiode, which is constructed of silicon materials and whose output voltage waveform is proportional to light. The use of a filter in front of a sensor improves selectivity. For example, a green (pass) filter in front of a light sensor favors light in the green band of the color spectrum. As a consequence, the sensor output will be stronger f or green light than for other components in the visible spectrum. 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.

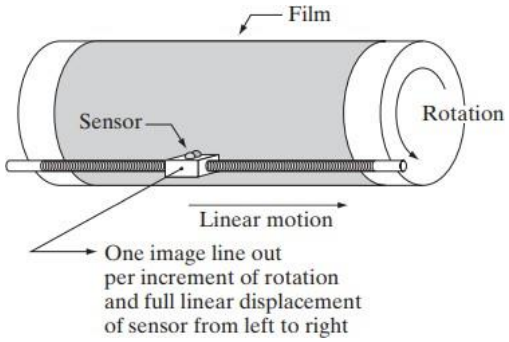


Fig. 1.5 Combining a single sensor with motion to generate a 2-D image.

Figure 1.5 shows an arrangement used in high-precision scanning, where a f ilm 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. Because 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 microdensitometers. 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 ref lect ed laser signal onto the sensor. This arrangement can be used also to acquire images using strip and array sensors, which are discussed in the following two sections.

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, as Fig. 1.4 (b) shows. The strip provides imaging elements in one direction Motion perpendicular to the strip provides imaging in the other direction, as shown in Fig. 1.6 (a). 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 electromagnet ic 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, as Fig. 1.6 (b) shows. A rotating X-ray source provides illumination and the sensors opposite the source collect the X-ray energy that passes 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) imaging.

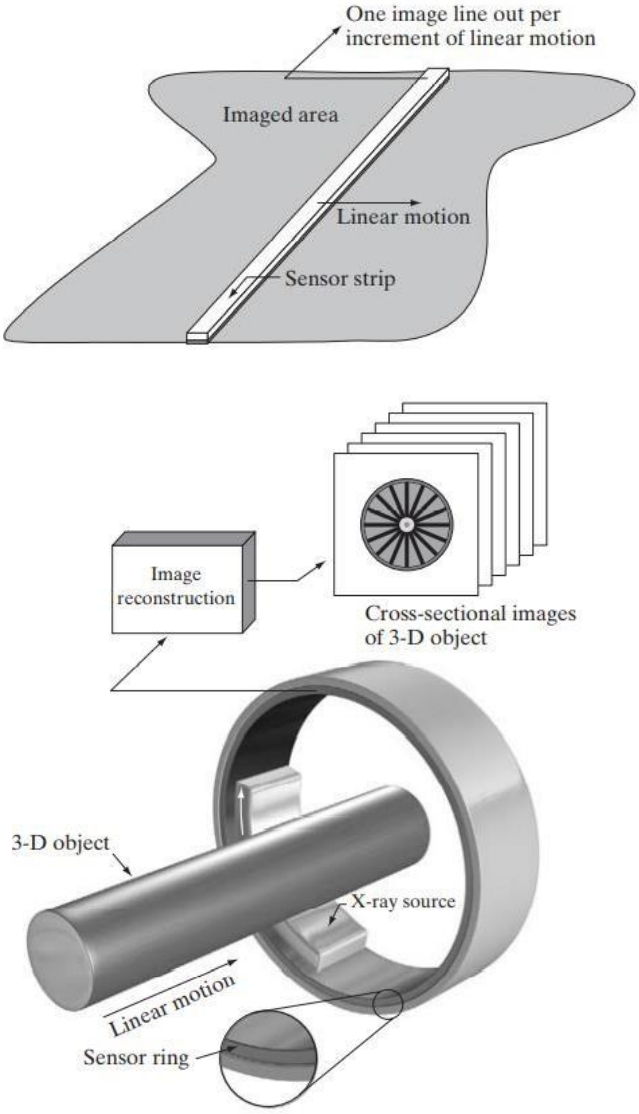


Fig. 1.6 (a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

Image Acquisition Using Sensor Arrays

Figure 1.4 (c) shows individual sensors arranged in the form of a 2 -D array. 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 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. Because the sensor array in Fig. 1.4 (c) is two-dimensional, its key advantage is that a complete image can be obtained by focusing the energy pattern onto the surface of the array. Motion obviously is not necessary, as is the case with the sensor arrangements discussed in the preceding two sections. The principal manner in which array sensors are used is shown in Fig. 1.7. This figure shows the energy from an illumination source being reflected from a scene element (as mentioned at the beginning of this section, the energy also could be transmitted through the scene elements). The first function performed by the imaging system in Fig. 1.7 (c) 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 an optical lens that projects the viewed scene onto the lens focal plane, as Fig. 1.7 (d) shows. 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 sweep these outputs and convert them to an analog signal, which is then digitized by another section of the imaging system. The output is a digital image, as shown diagrammatically in Fig. 1.7(e).

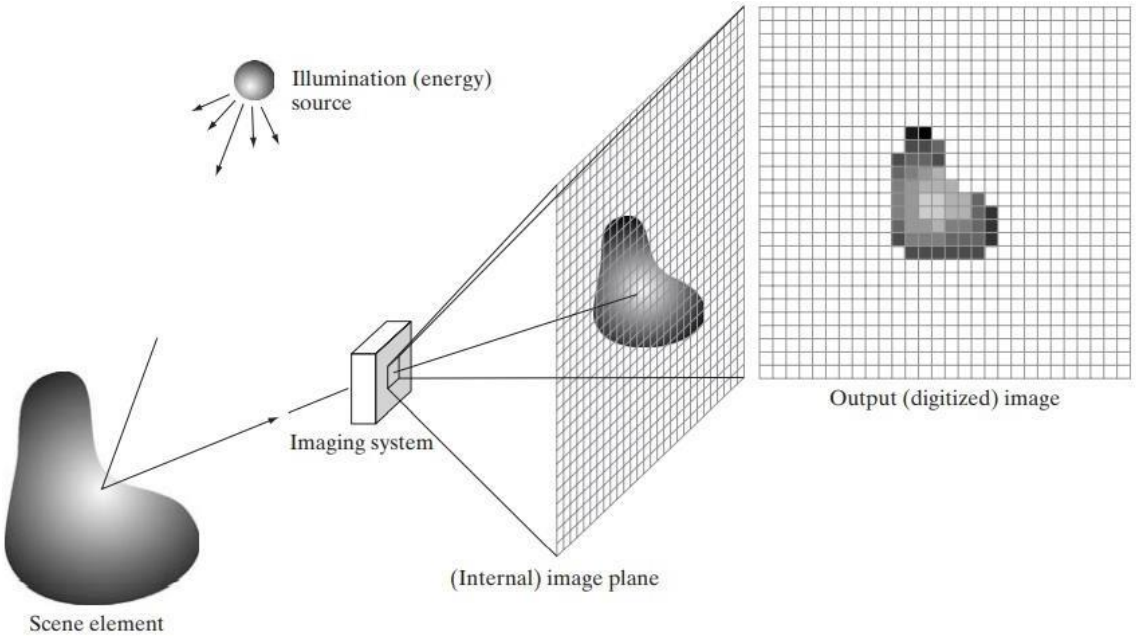


Fig. 1.7 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image

Image Sampling and Quantization

The basic idea behind sampling and quantization is illustrated in Fig. 1.8. Figure 1.8 (a) shows a continuous image f that we want to convert to digital form. An image may be continuous with respect to the x- and y-coordinates, and also in amplitude. To convert it to digital form, we have to sample the function in both coordinates and in amplitude. Digitizing

the coordinate values is called sampling. Digitizing the amplitude values is called quantization. The one-dimensional function in Fig. 1.8 (b) is a plot of amplitude (intensity level) values of the continuous image along the line segment AB in Fig. 1.8(a). The random variations are due to image noise. To sample this function, we take equally spaced samples along line AB, as shown in Fig. 1.8(c). The spatial location of each sample is indicated by a vertical tick mark in the bottom part of the figure. The samples are shown as small white squares superimposed on the function. The set of these discrete locations gives the sampled function. However, the values of the samples still span (vertically) a continuous range of intensity values. In order to form a digital function, the intensity values also must be converted (quantized) into discrete quantities. The right side of Fig. 1.8(c) shows the intensity scale divided into eight discrete intervals, ranging from black to white. The vertical tick marks indicate the specific value assigned to each of the eight intensity intervals. The continuous intensity levels are quantized by assigning one of the eight values to each sample. The assignment is made depending on the vertical proximity of a sample to a vertical tick mark. The digital samples resulting from both sampling and quantization are shown in Fig. 1.8 (d). Starting at the top of the image and carrying out this procedure line by line produces a two-dimensional digital image. It is implied in Fig. 1.8 that, in addition to the number of discrete levels used, the accuracy achieved in quantization is highly dependent on the noise content of the sampled signal. Sampling in the manner just described assumes that we have a continuous image in both coordinate directions as well as in amplitude. In practice, the method of sampling is determined by the sensor arrangement used to generate the image.

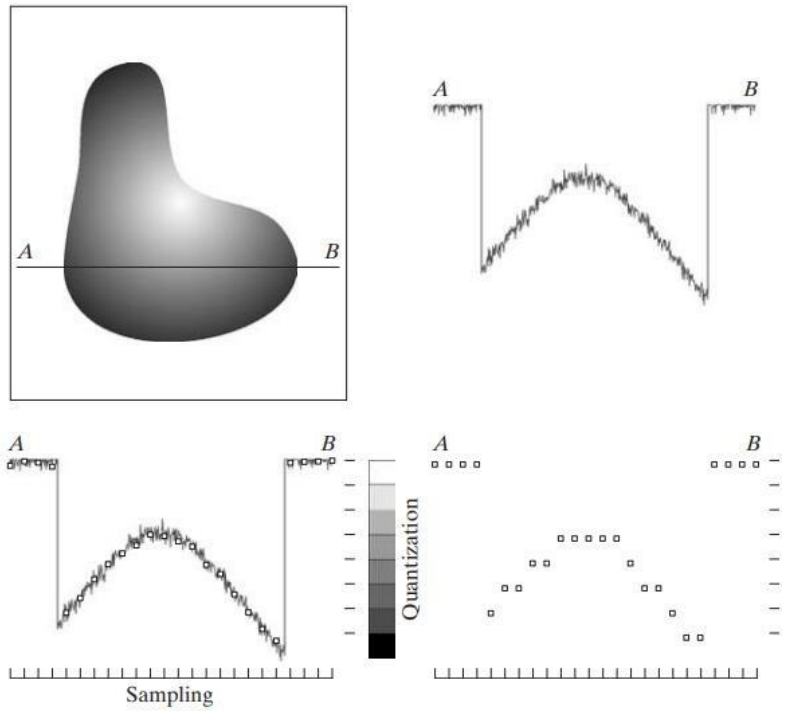


Fig. 1.8 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

When an image is generated by a single sensing element combined with mechanical motion, as in Fig. 1.5, the output of the sensor is quantized in the manner described above. However , spatial sampling is accomplished by selecting the number of individual mechanical increments at which we activate the sensor to collect data. Mechanical motion can be made very exact so, in principle, there is almost no limit as to how fine we can sample an image using this approach. In practice, limits on sampling accuracy are determined by other factors, such as the quality of the optical components of the system. When a sensing strip is used for image acquisition, the number of sensors in the strip establishes the sampling limitations in one image direction. Mechanical motion in the other direction can be controlled more accurately, but it makes little sense to try to achieve sampling density in one direction that exceeds the sampling limits established by the number of sensors in the other. Quantization of the sensor outputs completes the process of generating a digital image. When a sensing array is used for image acquisition, there is no motion and the number of sensors in the array establishes the limits of sampling in both directions. Quantization of the sensor outputs is as before. Figure 1.9 illustrates this concept. Figure 1.9(a) shows a continuous image projected onto the plane of an array sensor. Figure 1.9(b) shows the image after sampling and quantization. Clearly, the quality of a digital image is determined to a large degree by the number of samples and discrete intensity levels used in sampling and quantization.

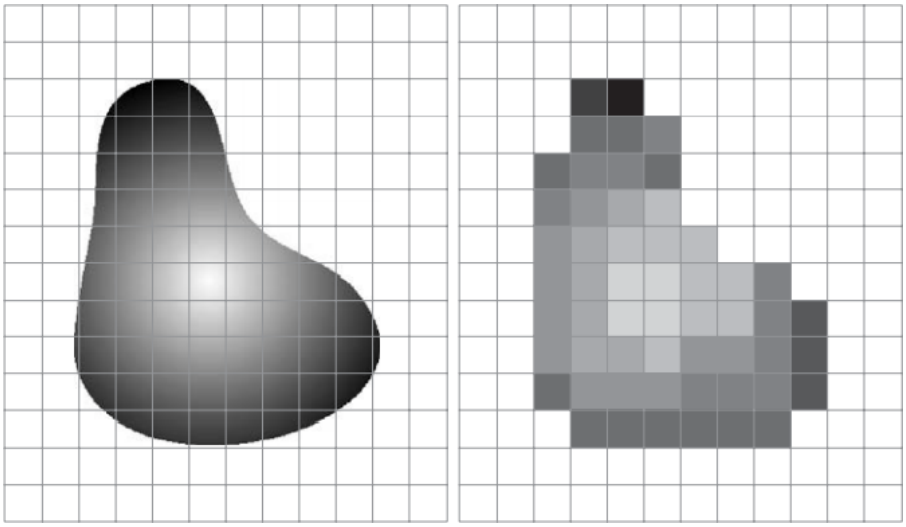


Fig. 1.9 (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

Basic Relationships between Pixels

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 neighbor locations 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).

Adjacency, Connectivity, Regions, and Boundaries

Let V be the set of intensity values used to define adjacency. In a binary image, V = {1} if we are referring to adjacency of pixels with value 1. In a gray-scale 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 intensity 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

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

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

(i) q is in or

(ii) q is in ND(p) and the set $N_4(p) \cap N_4(q)$ 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 is a sequence of distinct pixels with coordinates

(x0, y0), (x1, y1), … , (xn, yn)

where (x0, y0) = (x, y), (xn, yn) = (s, t) and pixels (xi, yi) and (xi-1, yi-1) are adjacent for 1 <= i <= n. In this case, n is the length of the path. . If (x0, y0) = (xn, yn) the path is a closed path

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. Two regions, and are said to be adjacent if their union forms a connected set. Regions that are not adjacent are said to be disjoint. We consider 4- and 8-adjacency when referring to regions.

Distance Measures

For pixels p, q, and z, with coordinates (x, y), (s, t), and (v, w), respectively, D is a distanc e function or metric if

i. D(p, q) >= 0 (D(p, q) = 0 iff p = q),

ii. D(p, q) = D(q, p) and

iii. D(p, z) < = D(p, q) + D(q, z).

The Euclidean distance between p and q is defined as

$$D_e(p, q) = \left[(x - s)^2 + (y - t)^2\right]^{\frac{1}{2}}$$

For this distance measure, the pixels having a distance less than or equal to some value r from (x, y) are the points contained in a disk of radius r centered at (x, y).

The distance (called the city-block distance) between p and q is defined as

$$D_4(p, q) = |x - s| + |y - t|$$

In this case, the pixels having a D4 distance from (x, y) less than or equal to some value r form a diamond centered at (x, y). For example, the pixels with D4 distance <=2 from (x, y) (the center point) form the following contours of constant distance:

```
      2
    2 1 2
  2 1 0 1 2
    2 1 2
      2
```

The pixels with D4=1 are the 4-neighbors of (x, y).

The distance (called the chessboard distance) between p and q is defined as

$$D_8(p, q) = \max(|x - s|, |y - t|)$$

In this case, the pixels with D8 distance from (x, y) less than or equal to some value r f orm a square centered at (x, y). For example, the pixels with D8 distance <= 2 from (x, y) (the center point) form the following contours of constant distance:

```
2 2 2 2 2
2 1 1 1 2
2 1 0 1 2
2 1 1 1 2
2 2 2 2 2
```

The pixels with D8=1 distance are the 8-neighbors of (D x, y).

Monochromatic Vision Models

When Light enters the eye,

- optical characteristics are represented by **LPF (Low Pass Filter)** with frequency response $\mathbf{H_l(\xi_1, \xi_2)}$
- spatial response are represented by the **relative luminous efficiency** function V(λ), yields the luminance distribution f(x, y) via

$$f(x, y) = \int_0^\infty I(x, y, \lambda) V(\lambda)\, d\lambda$$

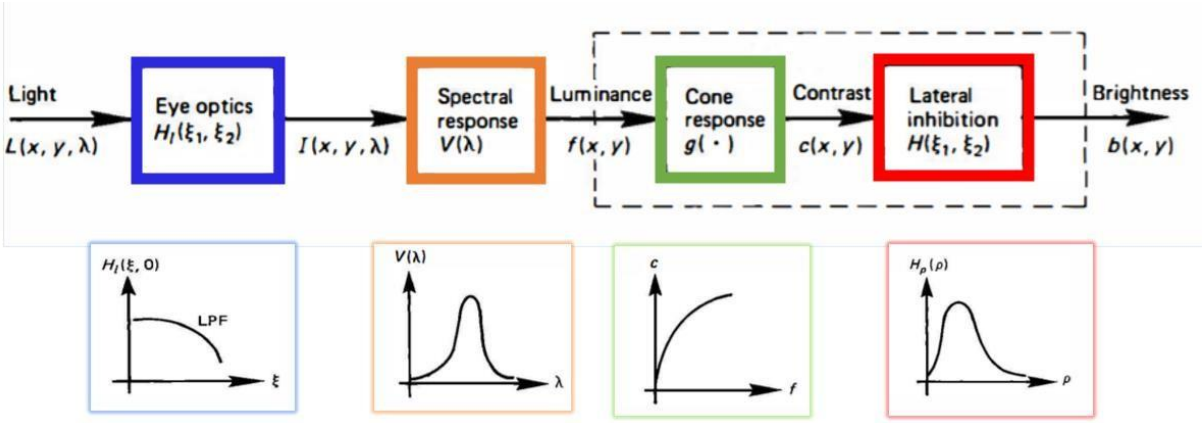


Fig. 1.10 Monochrome Vision Model

- The nonlinear response of the rods and cones, represented by the point nonlinearity **g (·),** yields the contrast **c(x, y)**
- The lateral inhibition phenomenon is represented by a spatially invariant, isotropic, linear system whose frequency response is $\mathbf{H(\xi_1, \xi_2)}$
- Its output is the neural signal, which represents the apparent brightness **b(x, y)**
- For an optically well-corrected eye, the low-pass filter has a much slower drop-off with increasing frequency than that of the lateral inhibition mechanism
- Thus the optical effects of the eye could be ignored, and the simpler model showing the transformation between the luminance and the brightness suffices

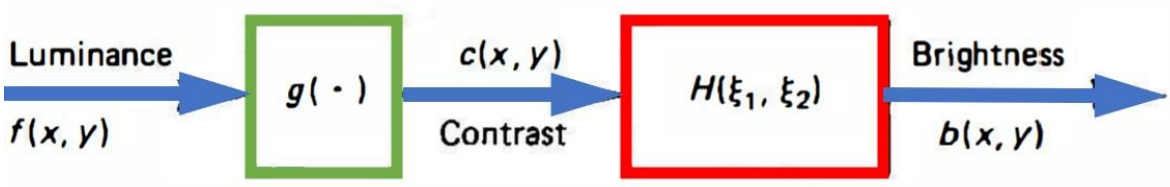


Fig. 1.11 Simplified Monochrome Vision Model

Colour Vision Models

- The color image is represented by the $\mathbf{R_N}$, $\mathbf{G_N}$, $\mathbf{B_N}$ coordinates at each pixel.
- The matrix A transforms the input into the three **cone responses α_k (x, y, C),** k = 1, 2, 3 where (x, y) are the spatial pixel coordinates and C refers to its color
- In Fig., we have represented the **normalized cone responses**
- In analogy with the definition of tristimulus values, Tk are called the retinal cone tristimulus coordinates
- The cone responses undergo nonlinear point transformations to give three f ields Tk (x, y), k = 1, 2, 3
- The 3 x 3 matrix B transforms the {f(x, y)} into {Ck(x, y)} such that C1 (x, y) is the monochrome (achromatic) contrast field c(x, y), as in simplified model, and C2 (x, y) and C3(x, y) represent the corresponding chromatic fields

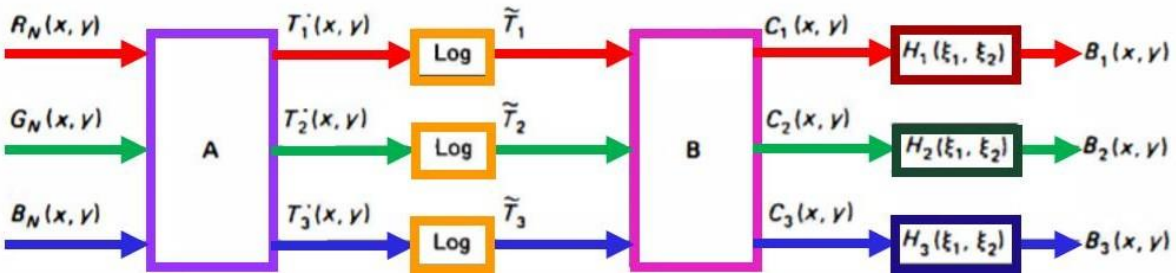


Fig. 1.12 Colour Vision Model

- The spatial filters $H_k(\xi_1 , \xi_2)$, k = 1, 2, 3, represent the frequency response of the visual system to luminance and chrominance contrast signals
- Thus $H1(\xi_1 , \xi_2)$ is the same as H (ξ 1 , ξ 2) in simplified model and is a bandpass filter that represents the lateral inhibition phenomenon

$$T_k \triangleq \frac{\alpha_k(x, y, C)}{\alpha_k(x, y, W)}, \qquad k = 1, 2, 3$$

- The visual frequency response to chrominance signals are not well established but are believed to have their passbands in the lower frequency region, as shown in figure
- The 3 x 3 matrices A and B are given as follows:

$$\mathbf{A} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.127 & 0.724 & 0.175 \\ 0.000 & 0.066 & 1.117 \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} 21.5 & 0.0 & 0.00 \\ -41.0 & 41.0 & 0.00 \\ -6.27 & 0.0 & 6.27 \end{pmatrix}$$

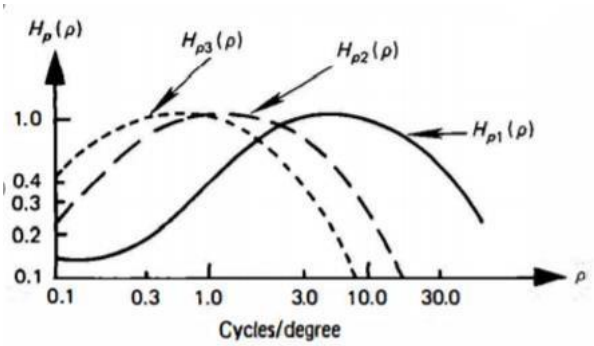


Figure 1.13 Frequency responses of the three color channels C_1, C_2, C_3 of the color vision model. Each filter is assumed to be isotropic so that $H_{pk}(\rho) \triangleq H_k(\xi_1, \xi_2)$, $\rho = \sqrt{\xi_1^2 + \xi_2^2}$, $k = 1, 2, 3$.

- From the model , a criterion for color image fidelity can be defined
- For example, for two color images {RN , GN , BN} and {R,,V, G,,V, BN}, their subjective mean square error could be defined by

$$e_{ls} = \frac{1}{A} \sum_{k=1}^{3} \iint_{\Re} (B_k(x, y) - B_i(x, y))^2 dx\, dy$$

Where r is the region which over the image is defined (or available), A is its area. And $\{B_k(x,y)\}$ and $\{B_{kdot}(x,y)\}$ are the outputs of the model for the two colour images.

Colour Fundamentals

Color spectrum

- In 1666, Sir Isaac Newton
- When a beam of sunlight is passed through a glass prism
- The emerging beam of light consists of a continuous spectrum of colors ranging from violet to red (not white)
- Divided into six broad regions
    - Violet, blue, green, yellow, orange, and red
- Each color blends smoothly into the next

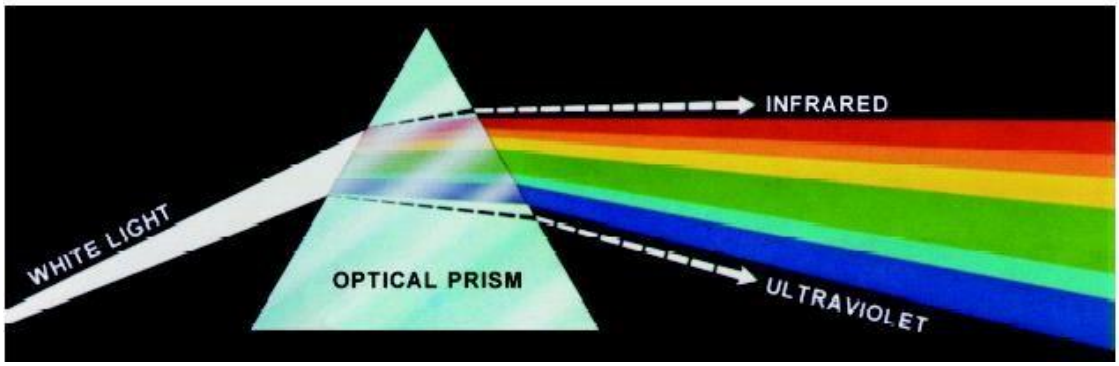


FIGURE 1.14 Color spectrum seen by passing white light through a prism. (Courtesy of the General Electric Co., Lamp Business Division.)

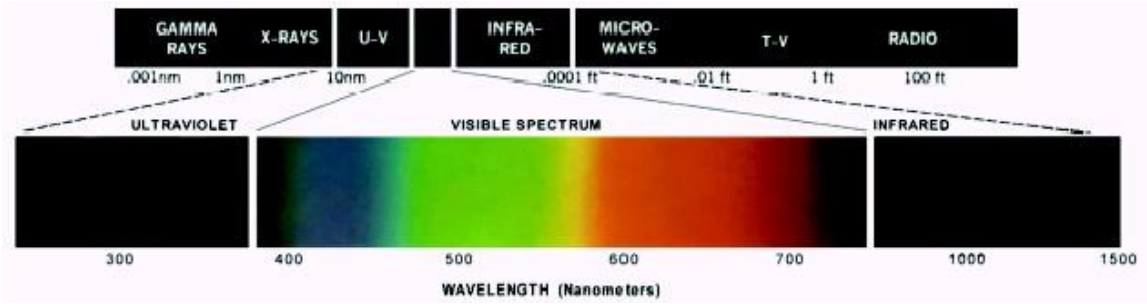


FIGURE 1.15 Wavelengths comprising the visible range of the electromagnetic spectrum. (Courtesy of the General Electric Co., Lamp Business Division.)

Primary colors of pigments or colorants

- cyan, magenta, yellow
- A primary color of pigments is defined as one that subtracts or absorbs a primary color of light and reflects or transmits the other two

Secondary colors of pigments or colorants

- red, green, blue
- Combination of the three pigment primaries, or a secondary with its opposite primary, produces black

Characteristics of colors

- Brightness:
  - The chromatic notion of intensity
- Hue:
  - An attribute associated with the dominant wavelength in a mixture of light waves
  - Representing dominant color as perceived by an observer
- Saturation
  - Referring to relative purity or the amount of white mixed with a hue
  - Saturation is inversely proportional to the amount of white light

Hue and saturation taken together are called chromaticity

- A color may be characterized by its brightness and chromaticity
- The amounts of red, green, and blue needed to form any particular color are called the tristimulus values (Denoted X(red), Y(green), and Z(blue))

$$x = \frac{X}{X+Y+Z}, \quad y = \frac{Y}{X+Y+Z}, \quad z = \frac{Z}{X+Y+Z}$$

$$x + y + z = 1 \ , \ X = \frac{x}{y}Y, Z = \frac{z}{y}Y$$

Chromaticity diagram

- Showing color composition as a function of x(R) and y(G)
- For any value of x and y, z(B) = 1 – (x + y)

- The positions of the various spectrum colors are indicated around the boundary of the tongue-shaped
- The point of equal energy = equal fractions of the three primary colors = CIE standard for white light
- Boundary : completely saturated
- Useful for color mixing
- A triangle(RGB) does not enclose the entire color region

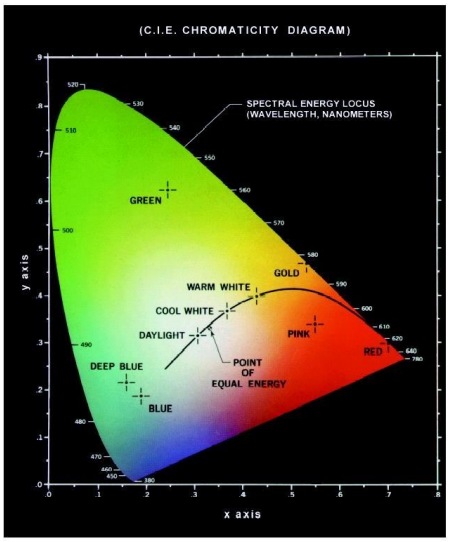


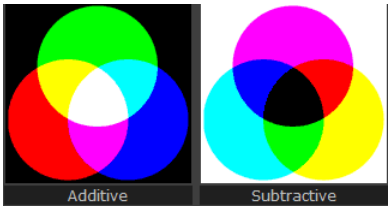
Fig. 1.17 Chromaticity diagram

Colour Models

The purpose of a color model is to facilitate the specification of colors in some standard Color models are oriented either toward hardware or applications

- Hardware-oriented
  - Color monitor or Video camera : RGB
  - Color printer : CMY
  - Color TV broadcast : YIQ (I : inphase, q : quadrature)
- Color image manipulation : HSI, HSV
- Image processing : RGB, YIQ, HIS

- Additive processes create color by adding light to a dark background (Monitors)
- Subtractive processes use pigments or dyes to selectively block white light (Printers)



Additive Color Mixing			Subtractive Color Mixing		
Red + Green	→	Yellow	Cyan + Magenta	→	Blue
Green + Blue	→	Cyan	Magenta + Yellow	→	Red
Blue + Red	→	Magenta	Yellow + Cyan	→	Green
Red + Green + Blue	→	White	Cyan + Magenta + Yellow	→	Black

Fig. 1.18 Colour Models

RGB color Model

Images represented in the RGB color model consist of three independent image planes, one for each primary color. The number of bits used to represent each pixel in RGB space is called the pixel depth. The term full-color image is used often to denote 24-bit RGB color image

- RGB model is based on a Cartesian coordinate system
- The color subspace of interest is the cube
- RGB values are at three corners
- Colors are defined by vectors extending from the origin
- For convenience, all color values have been normalized
- All values of R, G, and B are in the range [0, 1]

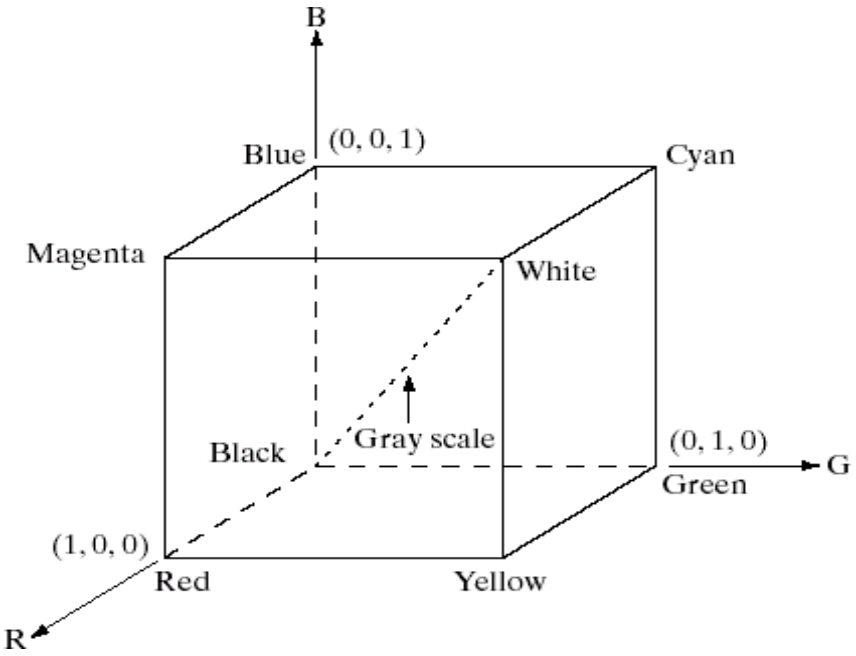


Fig. 1.19 RGB colour cube

CMY Colour Model

General purpose of CMY color model is to generate hardcopy output

- The primary colors of pigments
  - Cyan, Magenta, and Yellow
  - C = W - R, M = W - G, and Y = W - B
- Most devices that deposit colored pigments on paper require CMY data input
  - Converting RGB to CMY
  - The inverse operation from CMY to RGB is generally of no practical interest

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

HIS Colour Model

H : Hue

S : Saturation

I : Intensity

- The intensity is decoupled from the color information
- The hue and saturation are intimately related to the way in which human beings perceive color
- An ideal tool for developing image procession algorithms based on some of the color sensing properties of the human visual system

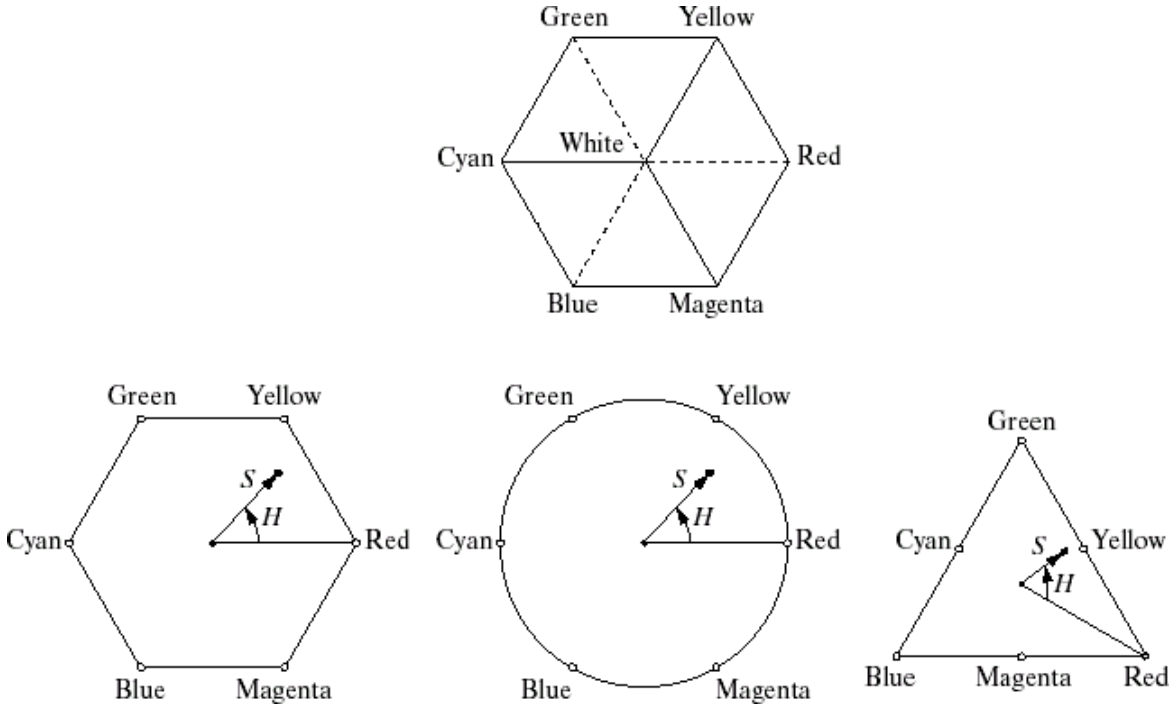


Fig. 1.20 Hue and Saturation in the HIS colour Model

Converting colors from RGB to HSI

- Hue component

$$H = \begin{cases} \theta & \text{if } B \le G \\ 360 - \theta & \text{if } B > G \end{cases} \qquad \theta = \cos^{-1}\left[\frac{\frac{1}{2}\{(R-G)+(R-B)\}}{\{(R- \quad ^2 \quad)^{1/2}} \right]$$

- Saturation component

$$S = 1 - \frac{3}{(R+G+B)}[\min(R, G, B)]$$

- Intensity component

$$I = \frac{1}{3}(R+G+B)$$

RGB values have been normalized to the range [0,1]

Angle θ is measured with respect to the red axis

Hue can be normalized to the range [0, 1] by dividing by 360°

Converting colors from HSI to RGB

- Three sectors of interest, corresponding to the 120^0 intervals in the separation of primaries

RG sector

$$(0^\circ \le H < 120^\circ)$$

$$B = I(1-S)$$
$$R = \left\lceil I\left[1 + \frac{S\cos H}{\cos(60^o - H)}\right]\right\rceil$$
$$G = 3I - (R+B)$$

GB sector

$$(120^\circ \le H < 240^\circ)$$

$$H = H - 120^o$$

$$R = I(1-S)$$
$$G = \left\lceil I\left[1 + \frac{S\cos H}{\cos(60^o - H)}\right]\right\rceil$$
$$B = 3I - (R+G)$$

BR sector

$$(240^\circ \le H < 360^\circ)$$

$$H = H - 240^o$$

$$G = I(1 - S)$$

$$B = I\left[1 + \frac{S\cos H}{\cos(60^o - H)}\right]$$

$$R = 3I - (G + B)$$

Quality metrics

Types of noise in an image have been categories on the basis of its statistical parameters

$$\text{Mean}(\mu) = \Sigma x_i / n$$

$$\text{Standard deviation}(\sigma) = \sqrt{\Sigma (x_i - x)^2 / (n - 1)}$$

$$\text{Skewness} = E(X - \mu)^3 / \sigma^3$$

$$\text{Kurtosis} = E(X - \mu)^4 / \sigma^{4-3}$$

where x_i is the ith sampled value of the input variable X.

The evaluation matrices used for the assessment of segmentation and denoising algorithms were Accuracy, Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), Visual Information Fidelity (VIF), Structural Similarity Index (SSIM) and Correlation matrix.

i. *Accuracy* have been determined from the confusion matrix using the formula

$$\text{Accuracy} = \text{TP} + \text{TN}/(\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

True positive (TP) = the number of pixels correctly identified as object pixels
True negative (TN) = the number of pixels correctly identified as background pixels
False positive (FP) = the number of pixels incorrectly identified as object pixels
False negative (FN) = the number of pixels incorrectly identified as background pixels.

ii. *PSNR* has been computed from Mean Square Error (MSE) as given below;

$$\text{PSNR} = 10log10(255/\text{MSE})$$

where

$$\text{MSE} = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} |\text{F}(\text{i}, \text{j}) - \text{f}(\text{i}, \text{j})|$$

Typical PSNR values range between 20 and 40. The actual value is not meaningful, but the comparison between two values for differently reconstructed images gives one, a measure of quality.

iii. *Visual Information Fidelity (VIF)* has been derived from the ratio between filtered output image information to the reference image information.
iv. *Structural Similarity Index Matrix (SSIM)* gives the comparison between the two images (original image -X, segmented image -Y) of same size with three measurements like luminance (L), contrast (C) and structure (S).

$$SSIM(X,Y) = [L(X,Y)a, C(X,Y)b, S(X,Y)c], \quad \text{Where}, a, b, c \text{ are weights.}$$

v. *Correlation* between two images X and Y having n-number of pixels has been determined by the equation

$$\text{Correlation}(X,Y) = \left[\sum x_i y_j - \left(\sum x_i y_j / n\right)\right] / \sqrt{\left[\left(\sum x_i - \left(\sum x_i\right)2/n\right) \times \left(\sum y_i - \left(\sum y_i\right)2/n\right)\right]}$$

Among these five metrics except MSE all other parameter values were required to be high for a featured image.

Fundamental Steps in Digital image processing

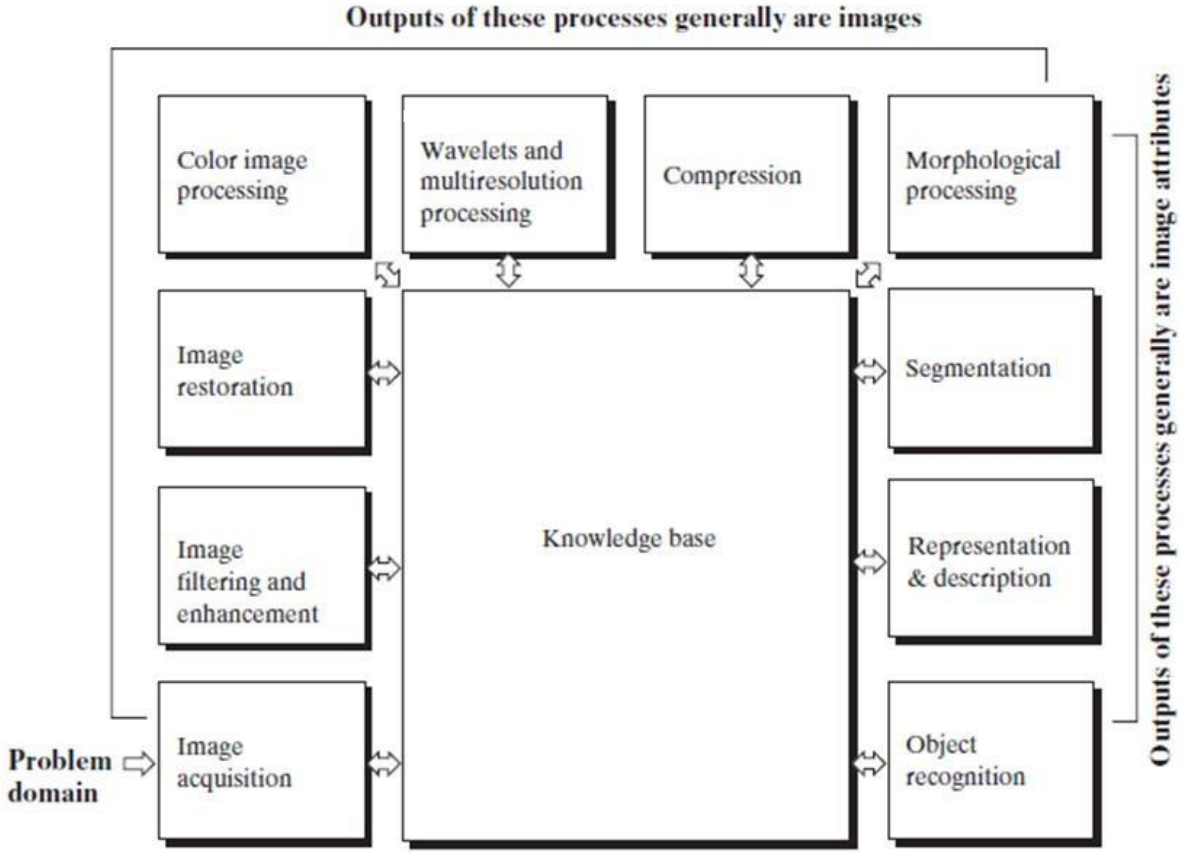


Image acquisition is the first process. Generally,the image acquisition stageinvolves preprocessing, such as scaling.

Image enhancement is the process of manipulating an image so that the result is more suitable than the original for a specific application. There is no general "theory" of image enhancement. When an image is processed for visual interpretation, the viewer is the ultimate judge of how well a particular method works.

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. This is true particularly in uses of the Internet.

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.

A segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually.

In general, the more accurate the segmentation, the more likely recognition is to succeed.

TEXT / REFERENCE BOOKS

1. Rafael C. Gonzalez, Richard E. Woods," Digital Image Processing", Pearson , Second Edition, 2004
2. Anil K. Jain, "Fundamentals of Digital Image Processin", Pearson 2002
3. Robert J.Schalkoff, "Pattern Recognition Statistical, Structural and Neural Approaches",John Wiley & Sons Inc., New York, 1992
4. R.O.Duda, P.E.Hart and D.G.Stork, "Pattern Classification", John Wiley, 2001
5. Himanshu Singh. "Practical Machine Learning and Image Processing", Apress, 2019
6. François Chollet "Deep Learning with Python", Manning Publications Co., NY, 2018
7. Phil Kim. "MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence", Apress, 2017
8. Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images
9. Valentina Zharkova, Lakhmi C. Jain, "Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images", Springer, 2007

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

UNIT – II – Digital Image Processing for Real Time Applications – SEC1628

II. Image Enhancement

Introduction; Point Processing - Image Negatives, Log transformations, Power Law Transformations, Piecewise-Linear Transformation Functions; Arithmetic/Logic Operations - Image Subtraction, Image Averaging; Histogram Processing - Histogram Equalization, Histogram Matching; Spatial filtering - Smoothing, Sharpening; Smoothing Frequency Domain Filters - Ideal Low Pass, Butterworth Low Pass, Gaussian Low Pass; Sharpening Frequency Domain Filters - Ideal High Pass, Butterworth High Pass, Gaussian High Pass. Image denoising- Wavelet Transform (DWT), Case study: Develop a program for the image denoising using DWT.

Point Processing

Image enhancement is to process the given image such that the result is more suitable to process than the original image. It sharpens the image features such as edges, boundaries or contrast the image for better clarity. It does not increase the inherent information content of the data, but increase the dynamic range of feature chosen. The main drawback of image enhancement is quantifying the criterion for enhancement and therefore large number of image enhancement techniques is empirical and require interactive procedure to obtain satisfactory results. Point Processing is the image enhancement at any point in an image depends only on the gray level at that point. Some of the basic intensity transformation functions are

- **Linear Functions:**
  - Identity Transformation
  - Negative Transformation
- **Logarithmic Functions:**
  - Log Transformation
  - Inverse-log Transformation
- **Power-Law Functions:**
  - n^{th} power transformation
  - n^{th} root transformation
- **Piecewise Transformation function**
  - Contrast Stretching
  - Gray-level Slicing
  - Bit-plane slicing

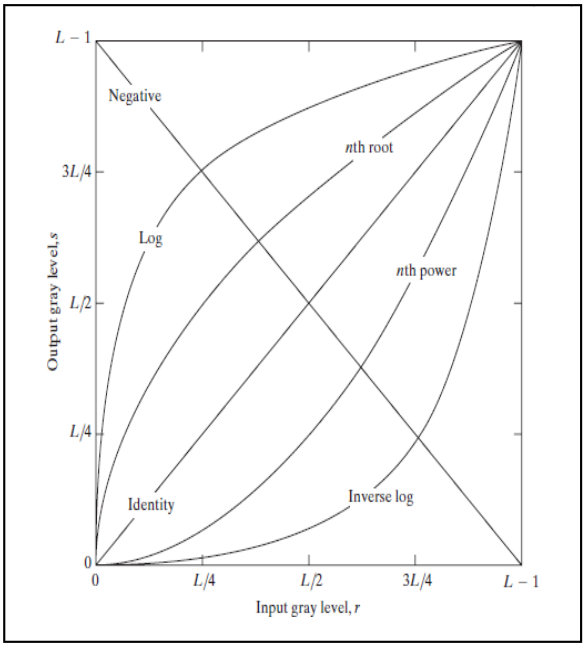


Fig. 2.1 Transformation Functions

Linear transformation

Linear transformation includes

- simple identity and
- negative transformation

- Identity transition is shown by a straight line
  - In this transition, each value of the input image is directly mapped to each other value of output image. That results in the same input image and output image. Hence is called identity transformation

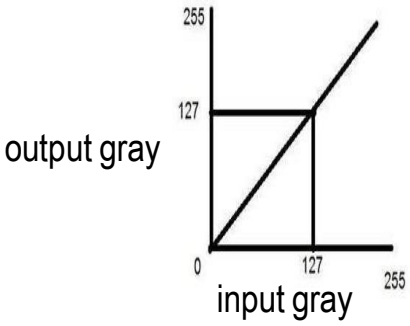
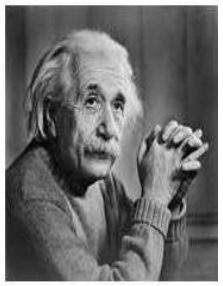


Fig. 2.2 Identity Transformation Function

Negative transformation

The negative of an image with gray level in the range **[0, L-1]** is obtained by using the negative transformation, the expression is given by **s = L-1-r**

- Reversing the intensity level of an image produces the equivalent of photographic negative
- Suitable for enhancing white or gray detail embedded in dark regions of an image, when the black areas are dominant in size



Input image

Output image

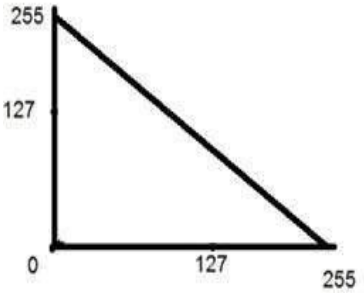
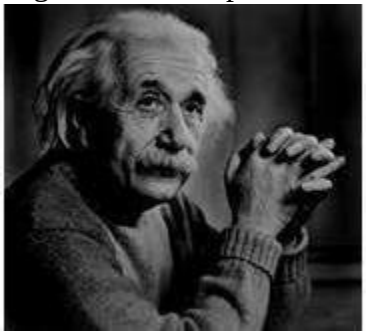


Fig. 2.2 Negative Transformation

Logarithmic transformations

The log transformations can be defined by $s = c \log(r + 1)$ where c is a constant and $r \geq 0$. During log transformation, the dark pixels in an image are expanded and the higher pixel values are compressed. The inverse log transform is opposite to log transform. Log transforms has the important characteristics: it compresses the dynamic range of images with large variation in pixel values.



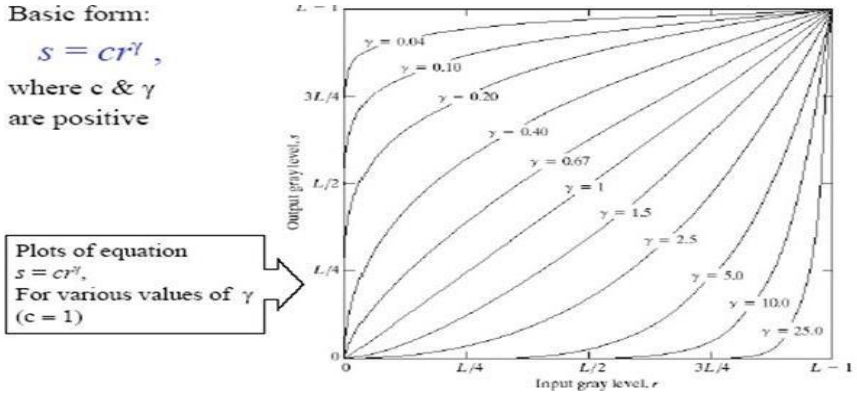
Input image

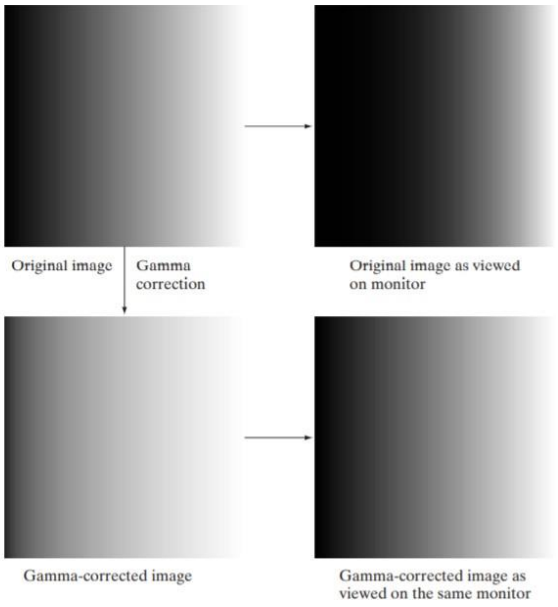
Output image

Fig. 2.3 Logarithmic Transformation

Power – Law transformations

This includes n^{th} power and n^{th} root transformations. It is given by the expression: $s=c\ r^{\gamma}$ (or) $s=c(r+\varepsilon)^{\gamma}$ where γ is called gamma, due to which this transformation is also known as gamma transformation. The exponent in the power law equation is referred to as gamma, the process used to correct this power-law response phenomenon is called gamma correction.





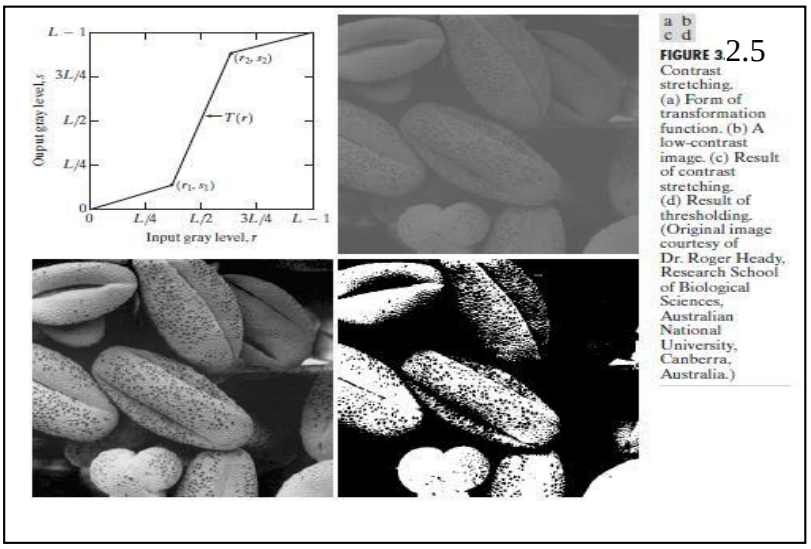
a b
c d

FIGURE 2.4
(a) Intensity ramp image. (b) Image as viewed on a simulated monitor with a gamma of 2.5. (c) Gamma-corrected image. (d) Corrected image as viewed on the same monitor. Compare (d) and (a).

Piecewise- Linear Transformation

One of the simplest piecewise linear functions is a contrast-stretching transformation, which is used to enhance the low contrast images. Low contrast images may result from poor illumination and wrong setting of lens aperture during image acquisition.

Contrast stretching



a b
c d
FIGURE 3.2.5 Contrast stretching. (a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

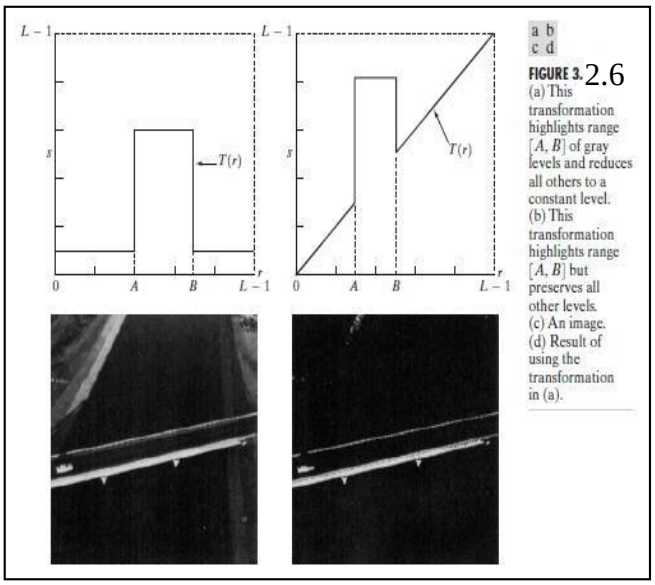
Figure shows a typical transformation used for contrast stretching. The locations of points (r1, s1) and (r2, s2) control the shape of the transformation function. If r1 = s1 and r2 = s2, the transformation is a linear function that produces no changes in gray levels. If r1 = r2, s1 = 0 and s2 = L-1, the transformation becomes a thresholding function that creates a binary image. Intermediate values of (r1, s1) and (r2, s2) produce various degrees of spread in the gray levels of the output image, thus affecting its contrast. In general, $r1 \leq r2$ and $s1 \leq s2$ is assumed, so the function is always increasing. Figure (b) shows an 8-bit image with low contrast. Fig. (c) shows the result of contrast stretching, obtained by setting (r1, s1) = (r_{min}, 0) and (r2, s2) = (r_{max},L-1) where r_{min} and r_{max} denote the minimum and maximum gray levels in the image, respectively. Thus, the transformation function stretched the levels linearly from their original range to the full range [0, L-1]. Finally, Fig. (d) shows the result of using the thresholding function defined previously, with r1=r2=m, the mean gray level in the image.

Gray-level Slicing

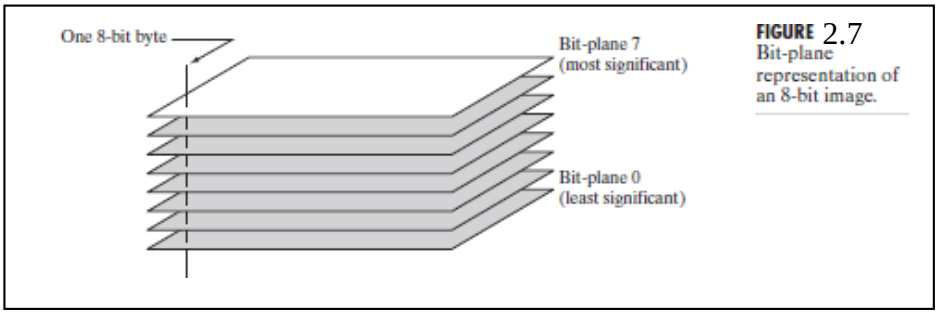
This technique is used to highlight a specific range of gray levels. It can be implemented in several ways, but the two basic themes are:

One approach is to display a high value for all gray levels in the range of interest and a low value for all other gray levels. This transformation, shown in Fig. (a), produces a binary image. The second approach, based on the transformation shown in Fig. (b), this brightens the desired range of gray levels but preserves gray levels unchanged Fig.(c)

shows a gray scale image, and fig.(d) shows the result of using the transformation in Fig.(a).



Bit-plane Slicing



Pixels are digital numbers, each one composed of bits. Instead of highlighting gray-level range, we could highlight the contribution made by each bit. This method is useful and used in image compression. Most significant bits contain the majority of visually significant data.

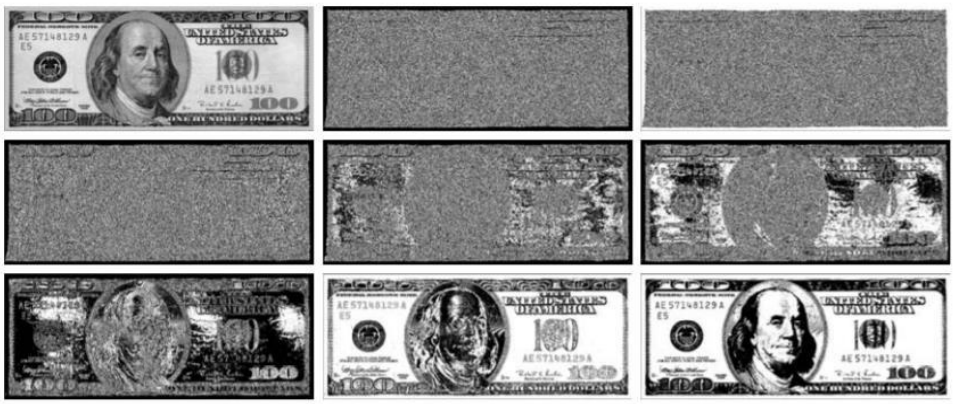


Fig. 2.8 Example for bit plane slicing

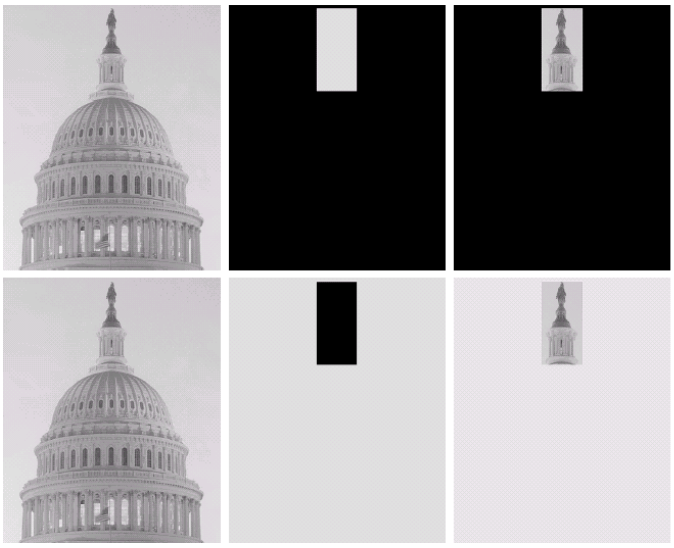
Arithmetic/Logic Operations

Image arithmetic applies one of the standard arithmetic operations or a logical operator to two or more images. The operators are applied in a pixel-by-pixel way, i.e. the value of a pixel in the output image depends only on the values of the corresponding pixels in the input images. Hence, the images must be of the same size. Although image arithmetic is the most simple form of image processing, there is a wide range of applications.

Logical operators are often used to combine two (mostly binary) images. In the case of integer images, the logical operator is normally applied in a bitwise way.

$$s(x, y) = f(x, y) + g(x, y)$$
$$d(x, y) = f(x, y) - g(x, y)$$
$$p(x, y) = f(x, y) \times g(x, y)$$
$$v(x, y) = f(x, y) \div g(x, y)$$

Arithmetic/logical operations are performed on pixel-by-pixel basis based on two or more images. When dealing with logical operations on gray-scale images, pixel values are processed as strings of binary numbers. In AND and OR image masks, light represents a binary 1 and dark represents a binary 0. Masking refers to as Region of Interest (ROI) processing.



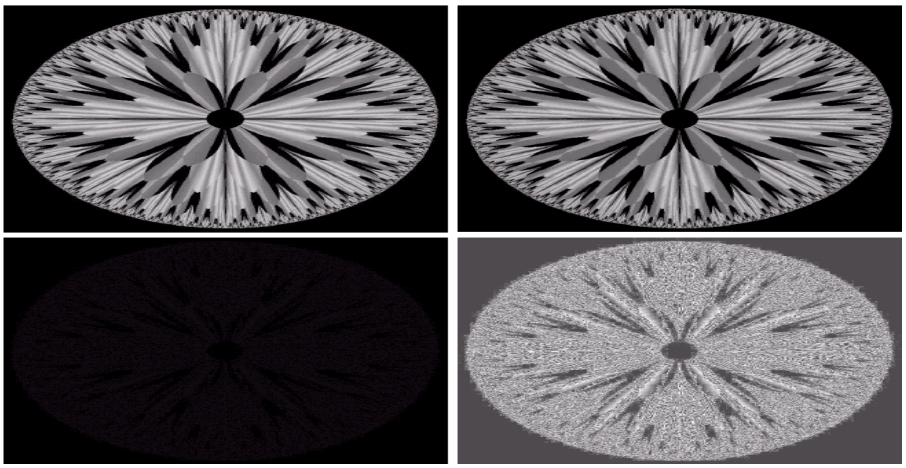
a b c
d e f

FIGURE 2.9 (a) Original image. (b) AND image mask. (c) Result of the AND operation on images (a) and (b). (d) Original image. (e) OR image mask. (f) Result of operation OR on images (d) and (e).

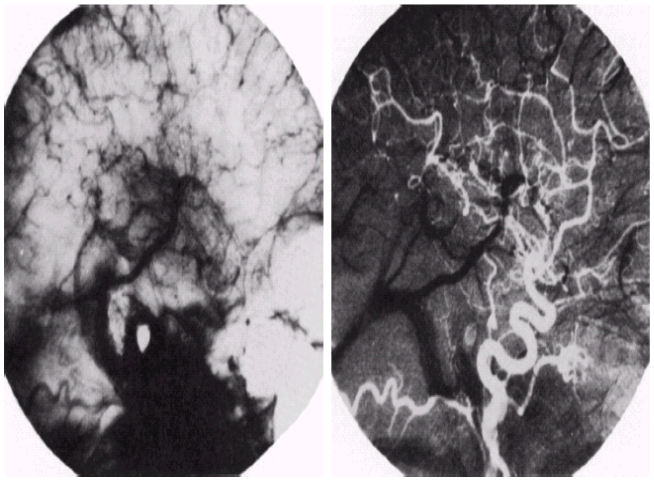
Image Subtraction

- Enhancement of differences between images
- Key usefulness of subtraction is the enhancement of differences between images.
- If the difference in the pixel value is small, then the image appears black when displayed in 8-bit display.
- To bring more detail contrast stretching can be performed

$$g(x, y) = f(x, y) - h(x, y)$$



a b
c d
FIGURE 2.10 (a) Original fractal image. (b) Result of setting the four lower-order bit planes to zero. (c) Difference between (a) and (b). (d) Histogram-equalized difference image. (Original image courtesy of Ms. Melissa D. Binde, Swarthmore College, Swarthmore, PA).



a b
FIGURE 2.11 Enhancement by image subtraction. (a) Mask image. (b) An image (taken after injection of a contrast medium into the bloodstream) with mask subtracted out.

Image Averaging

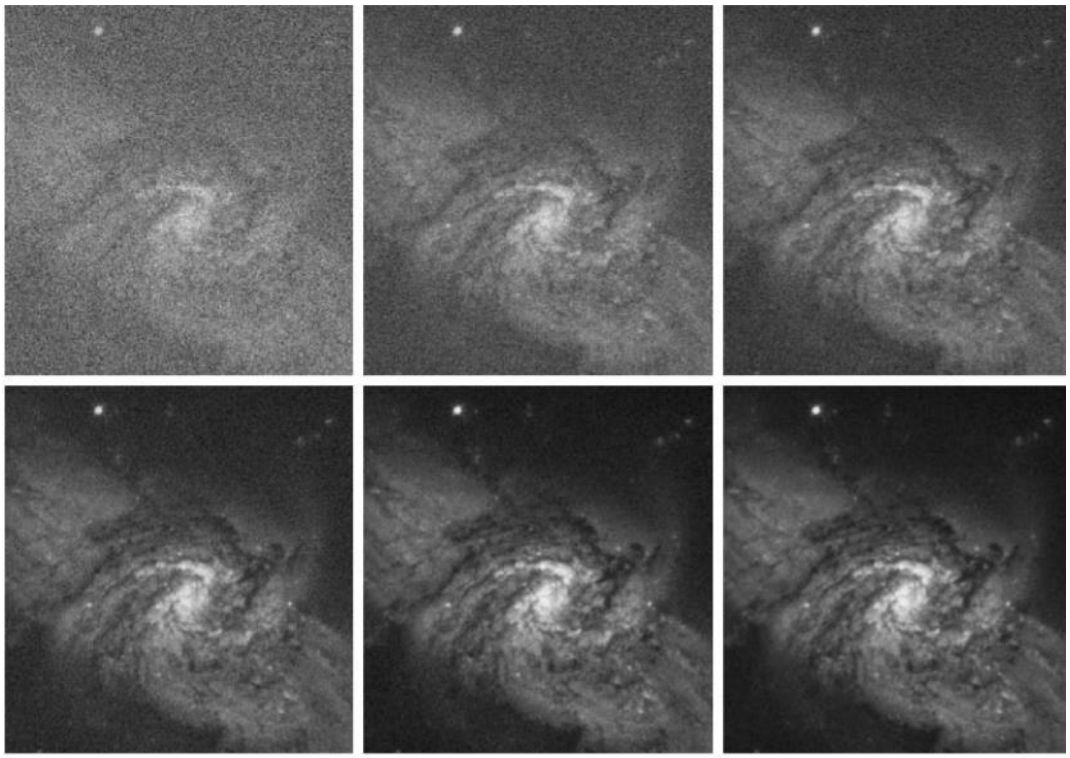
- Noisy image g(x,y) formed by the addition of noise

$$g(x, y) = f(x, y) + \eta(x, y)$$

- Averaging K different noisy images η(x,y) to an original image f(x,y)
- Objective is to reduce the noise content by adding a set of noisy images $\{g_i(x,y)\}$

$$\bar{g}(x,y) = \frac{1}{K}\sum_{i=1}^{K} g_i(x,y) \qquad E\{\bar{g}(x,y)\} = f(x,y)$$

$$\sigma^2_{\bar{g}(x,y)} = \frac{1}{K}\sigma^2_{\eta(x,y)} \qquad \sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{K}}\sigma_{\eta(x,y)}$$



a b c
d e f

FIGURE 2.12 (a) Image of Galaxy Pair NGC 3314 corrupted by additive Gaussian noise. (b)–(f) Results of averaging 5, 10, 20, 50, and 100 noisy images, respectively. (Original image courtesy of NASA.)

Histogram Processing

- Histogram is a graphical representation showing a visual impression of the distribution of data
- An Image Histogram is a type of histogram that acts as a graphical representation of the lightness/color distribution in a digital image
- It plots the number of pixels for each value

- The histogram of a digital image with gray levels in the range [0, L-1] is a discrete function $h(r_k) = n_k$ where r_k is the k^{th} gray level and n_k is the number of pixels in the image having gray level r_k

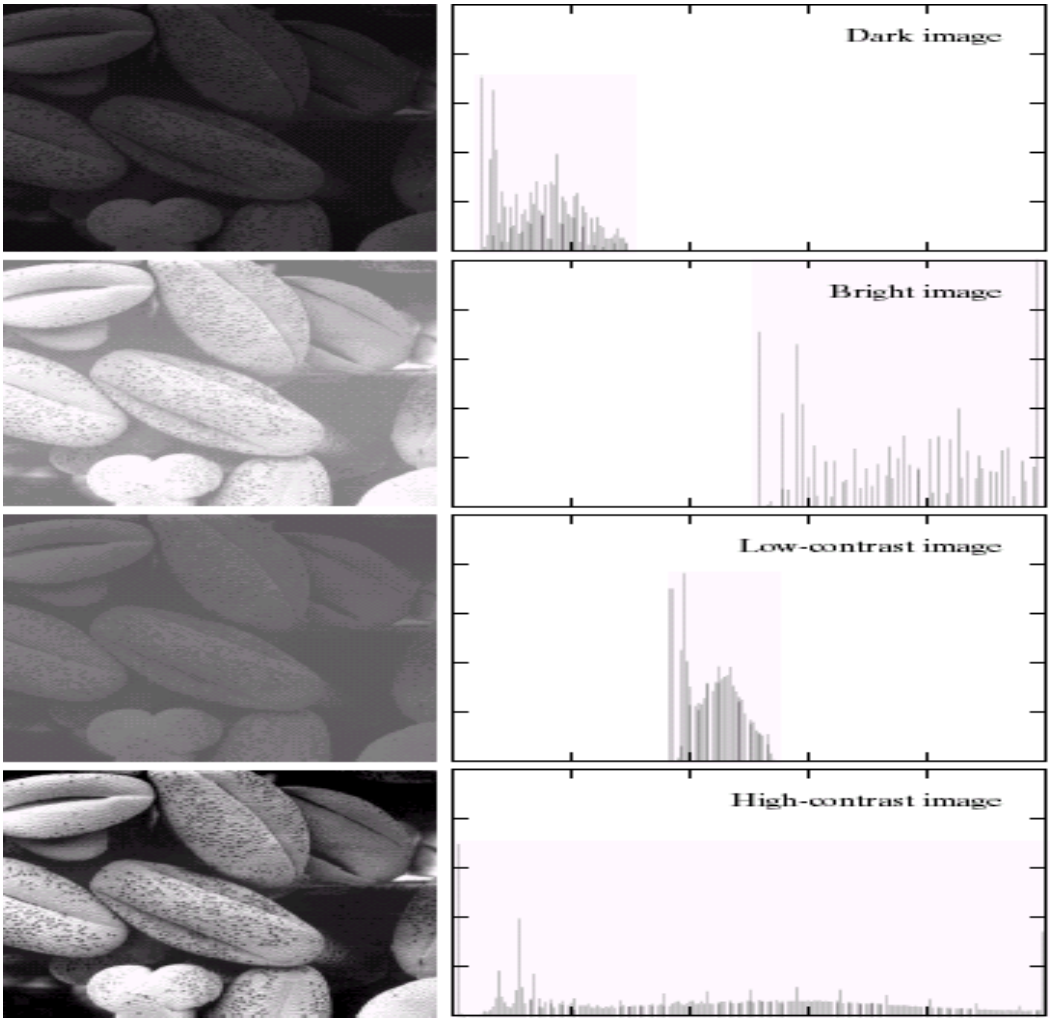


Fig. 2.13 Histogram of different types of images

It is common practice to normalize a histogram by dividing each of its values by the total number of pixels in the image, denoted by n.

A normalized histogram is given by

$$p(r_k) = n_k / n \qquad \text{for } k = 0, 1, \ldots, L-1$$

Thus, $p(r_k)$ gives an estimate of the probability of occurrence of gray level r_k

Note:

The sum of all components of a normalized histogram is equal to 1

Histogram Equalisation

The gray levels in an image is viewed as random variables in the interval[0,1]

Fundamental descriptors of a random variables is its probability density function

$p_s(s)$ and $p_s(r)$ are PDF of s and r

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$

Transformation has the particular importance in image processing

$$s = T(r) = \int_0^r p_r(\omega)d\omega$$

Discrete version of transformation- histogram equalization or histogram linearization

$$s_k = T(r_k) = \sum_{j=0}^{k}\frac{n_j}{n} = \sum_{j=0}^{k} p_r(r_j)$$

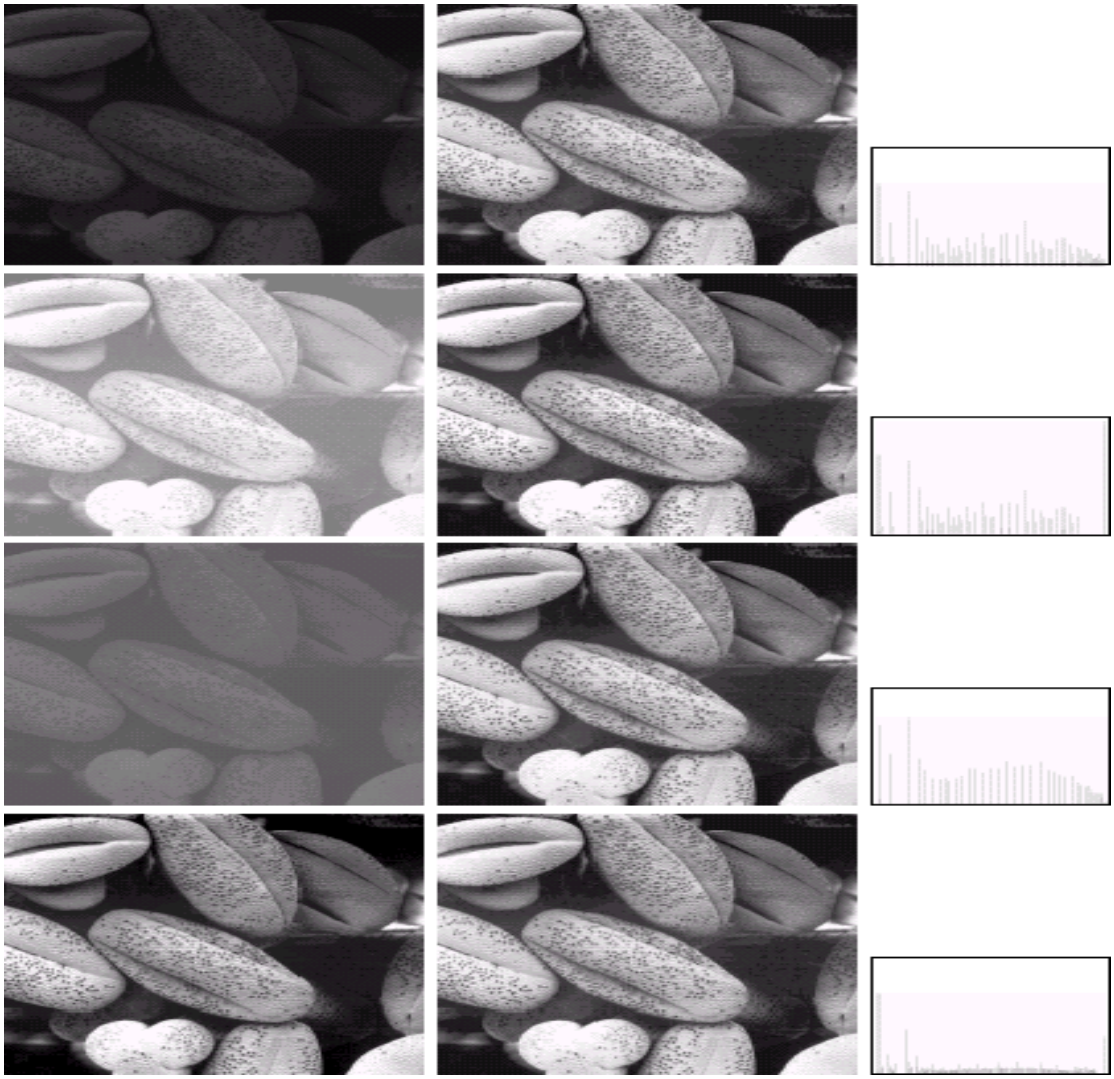


Fig. 2.14 Images after Histogram equalization

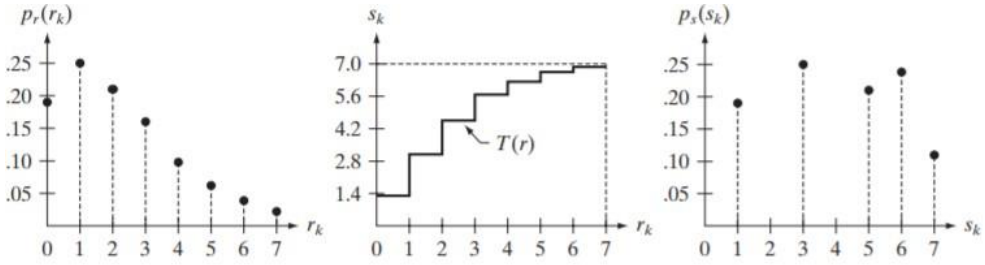
Assume the images have 64 x 64 = 4096 pixels in 8 gray levels. The following table shows the equalization process

Original Image Gray Level	No. of pixels (frequency)	Probability	Cumulative Probability	Multiply by Max. Gray Level	Rounding
0	790	0.19	0.19	1.33	1
1	1023	0.25	0.44	3.08	3

2	850	0.21	0.65	4.55	4
3	656	0.16	0.81	5.67	5
4	329	0.08	0.89	6.23	6
5	245	0.06	0.95	6.65	6
6	122	0.03	0.98	6.86	6
7	81	0.02	1.00	7	7

TABLE 2.1
Intensity distribution and histogram values for a 3-bit, 64 × 64 digital image.

r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02



a b c

FIGURE 2.15 Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

- r is in the range with representing black and representing white
- For r satisfying these conditions, we focus attention on transformations (intensity mappings) of the form

$$s = T(r) \quad 0 \le r \le L - 1 \tag{3.3-1}$$

that produce an output intensity level s for every pixel in the input image having intensity r. We assume that:

(a) $T(r)$ is a monotonically[†] increasing function in the interval $0 \le r \le L - 1$; and

(b) $0 \le T(r) \le L - 1$ for $0 \le r \le L - 1$.

In some formulations to be discussed later, we use the inverse

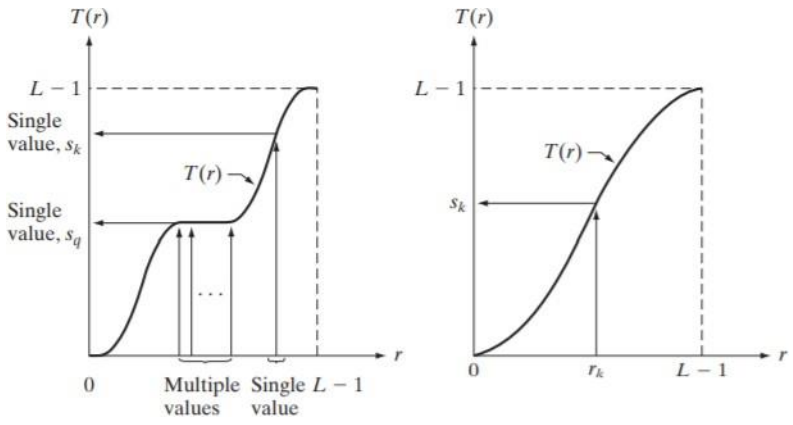
$$r = T^{-1}(s) \quad 0 \le s \le L - 1 \tag{3.3-2}$$

in which case we change condition (a) to

(a′) $T(r)$ is a strictly monotonically increasing function in the interval $0 \le r \le L - 1$.

a b

FIGURE 2.16 (a) Monotonically increasing function, showing how multiple values can map to a single value. (b) Strictly monotonically increasing function. This is a one-to-one mapping, both ways.



Histogram Matching (Specification)

Procedure for histogram matching:

- Obtain histogram of given image
- Use the equation to pre compute a mapped level s_k for each level r_k

$$s_k = T_r(k) = \sum_{j=0}^{k} p_r(r_j) = \sum_{j=0}^{k} n_j / n$$

- Obtain the transformation function G from the given $p_z(z)$ using the equation

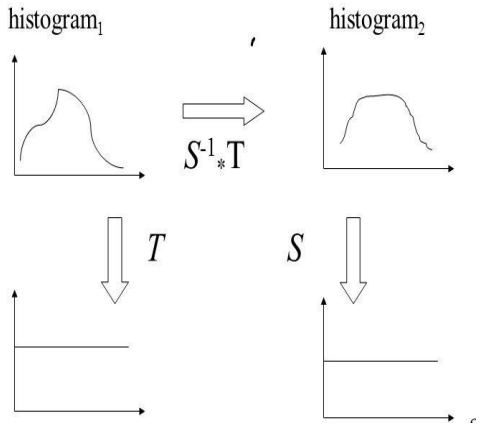
$$(G(\hat{z}) - sk) \ge 0; k = 0,1,2 \ \ldots L-1$$

- Precompute z_k for each value of s_k using the iterative scheme defined in connection with equation

$$(G(z\hat{}) - sk) \geq 0; k = 0,1,2 \ldots L-1$$

- For each pixel in the original image, if the value of that pixel is r_k, map this value to the corresponding level s_k; then map levels s_k into the final level z_k

Processed image that has a specified histogram is called histogram matching or histogram specification.



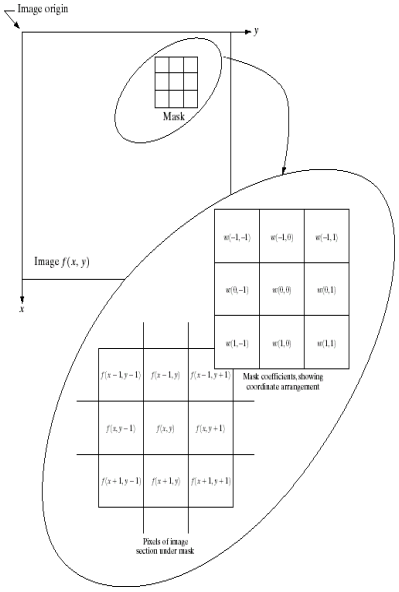
Spatial Filtering

- The output intensity value at (x,y) depends not only on the input intensity value at (x,y) but also on the specified number of neighboring intensity values around (x,y)
- Spatial masks (also called window, filter, kernel, template) are used and convolved over the entire image for local enhancement (spatial filtering)
- The size of the masks determines the number of neighboring pixels which inf luence the output value at (x,y)
- The values (coefficients) of the mask determine the nature and properties of enhancing technique.
- The mechanics of spatial filtering
- For an image of size *M* x *N* and a mask of size *m* x *n*
- The resulting output gray level for any coordinates x and y is given by

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t)\, f(x+s, y+t)$$

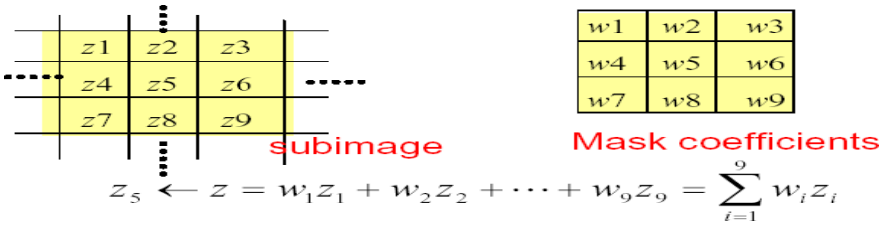
where $a = (m-1)/2, \quad b = (n-1)/2$

$x = 0, 1, 2, \ldots\ldots, M-1, \quad y = 0, 1, 2, \ldots\ldots, N-1,$



Given the 3×3 mask with coefficients: w_1, w_2,…, w_9

The mask cover the pixels with gray levels: z_1, z_2,…, z_9



z gives the output intensity value for the processed image (to be stored in a new array) at the location of z_5 in the input image

Mask operation near the image border

Problem arises when part of the mask is located outside the image plane; to handle the problem:

- Discard the problem pixels (e.g. $512x512_{input}$ $510x510_{output}$ if mask size is 3x3)
- Zero padding: expand the input image by padding zeros ($512x512_{input}$ $514x514_{output}$)
- Zero padding is not good - create artificial lines or edges on the border

- We normally use the gray levels of border pixels to fill up the expanded region (for 3x3 mask). For larger masks a border region equal to half of the mask size is mirrored on the expanded region.

Spatial Filtering for Smoothing

- For blurring/noise reduction;
- Smoothing/Blurring is usually used in preprocessing steps,

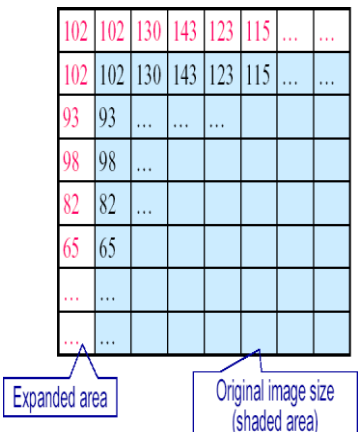
e.g., to remove small details from an image prior to object extraction, or to bridge small gaps in lines or curves

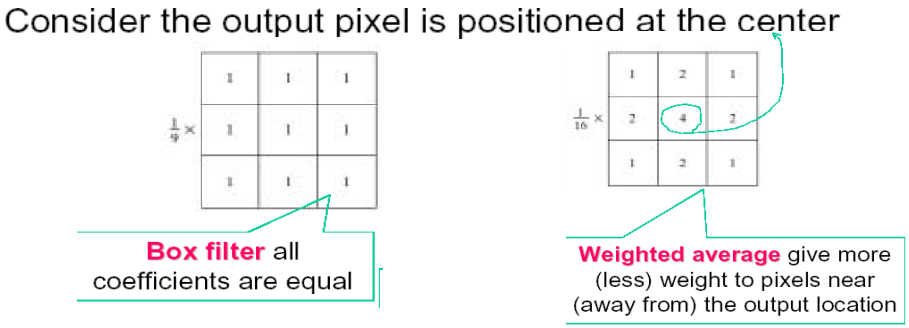
- Equivalent to Low-pass spatial filtering in frequency domain because smaller (high frequency) details are removed based on neighborhood averaging (averaging filters)

Implementation: The simplest form of the spatial filter for averaging is a square mask (assume $m \times m$ mask) with the same coefficients $1/m^2$ to preserve the gray levels (averaging).

Applications: Reduce noise; smooth false contours

Side effect: Edge blurring





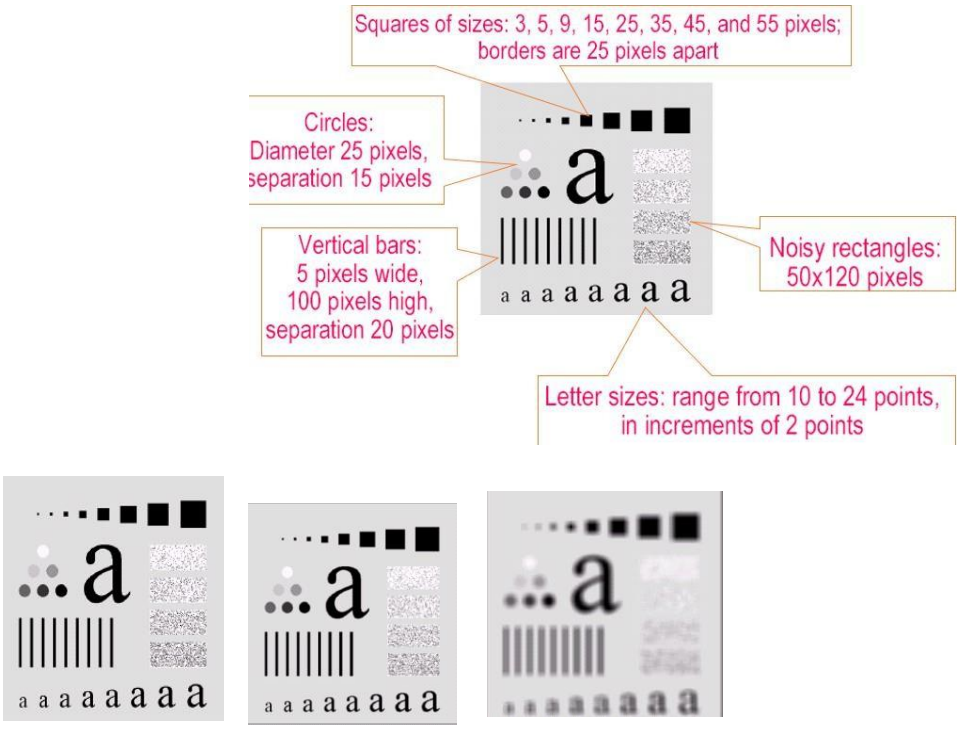


Fig. 2.17 Spatial filtering

Spatial Filtering for Sharpening

Background: to highlight fine detail in an image or to enhance blurred detail

Applications: electronic printing, medical imaging, industrial inspection, autonomous target detection (smart weapons)

Foundation:

- Blurring/smoothing is performed by spatial averaging (equivalent to integration)
- Sharpening is performed by noting only the gray level changes in the image that is the differentiation

Operation of Image Differentiation

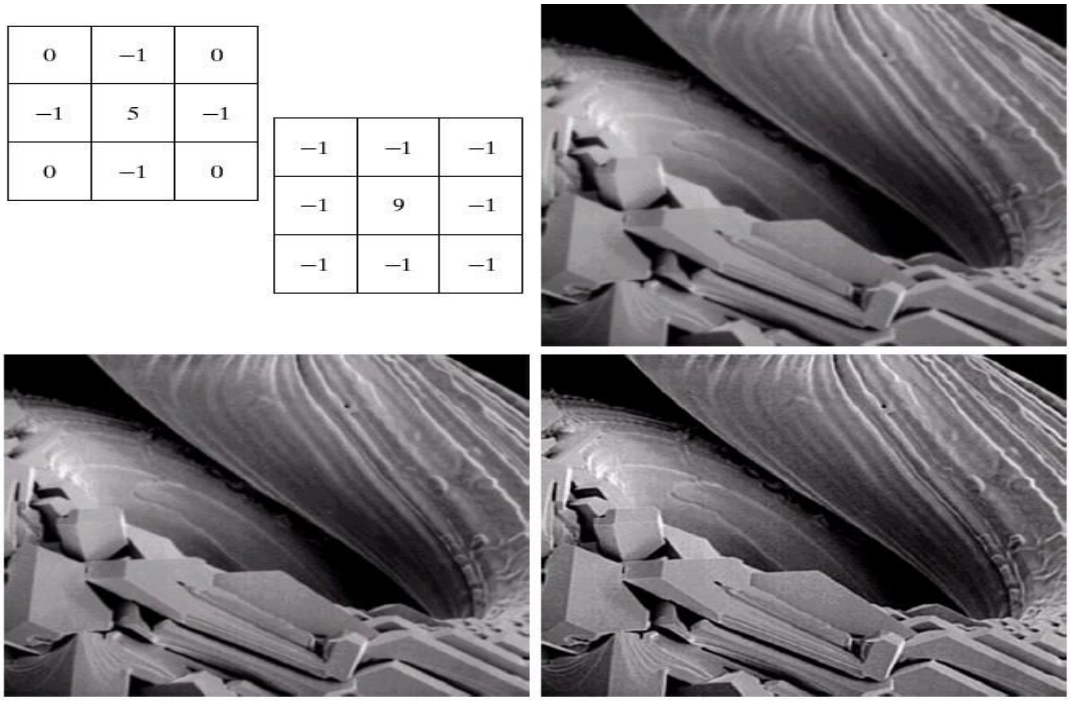
- Enhance edges and discontinuities (magnitude of output gray level >>0)
- De-emphasize areas with slowly varying gray-level values (output gray level: 0)

Mathematical Basis of Filtering for Image Sharpening

- First-order and second-order derivatives
- Approximation in discrete-space domain
- Implementation by mask filtering

0	−1	0
−1	5	−1
0	−1	0

−1	−1	−1
−1	9	−1
−1	−1	−1



a b c
d e

FIGURE 2.18 (a) Composite Laplacian mask. (b) A second composite mask. (c) Scanning electron microscope image. (d) and (e) Results of filtering with the masks in (a) and (b), respectively. Note how much sharper (e) is than (d). (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

Frequency Domain Filters

- Any function that periodically repeats itself can be expressed as the sum of sines and/or cosines of different frequencies, each multiplied by a different coefficient (Fourier series).
- Even functions that are not periodic (but whose area under the curve is finite) can be expressed as the integral of sines and/or cosines multiplied by a weighting function (Fourier transform).
- The **frequency domain** refers to the plane of the two dimensional discrete Fourier transform of an image.
- The purpose of the Fourier transform is to represent a signal as a linear combination of sinusoidal signals of various frequencies.

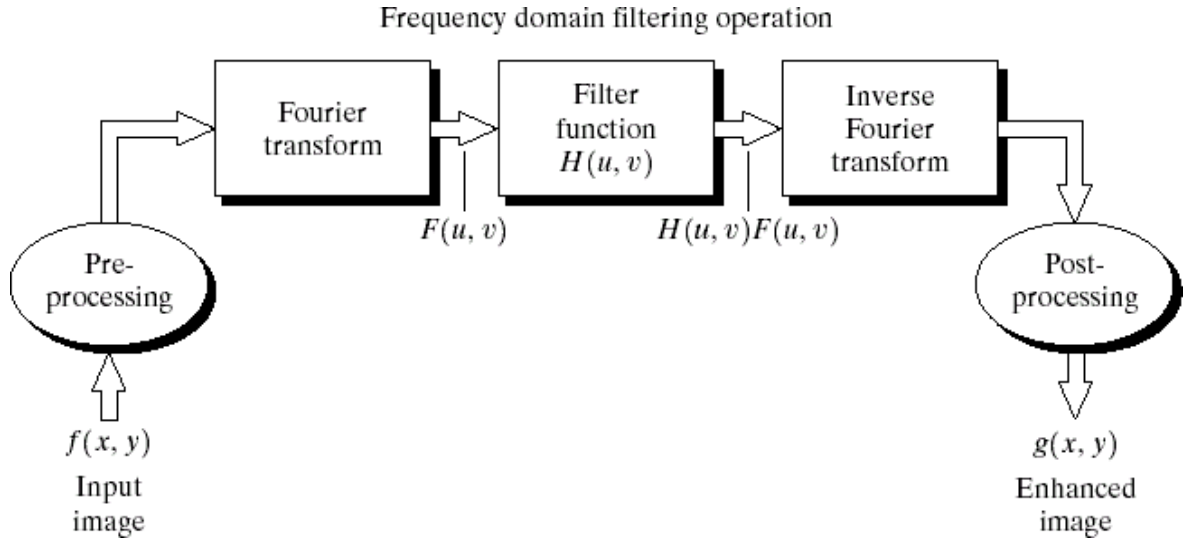


Fig. 2.19 Frequency domain operations

Frequency Domain Filters - Smoothing

Ideal Low Pass Filter

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) \ge D_0 \end{cases}$$

where D(u,v) is the distance to the center freq.

$$D(u,v) = [(u - M/2)^2 + (v - N/2)^2]^{1/2}$$

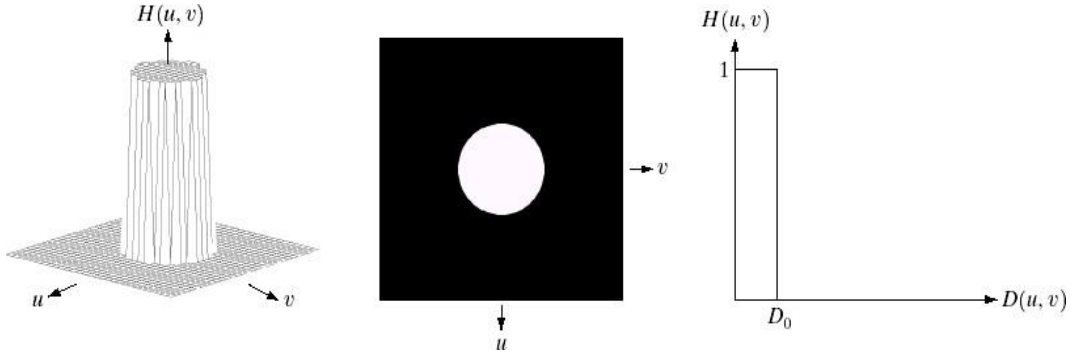


Fig. 2.20 Ideal Low Pass filter

Butterworth Low Pass Filter

$$H(u,v) = \frac{1}{1+[D(u,v)/D_0]^{2n}}$$

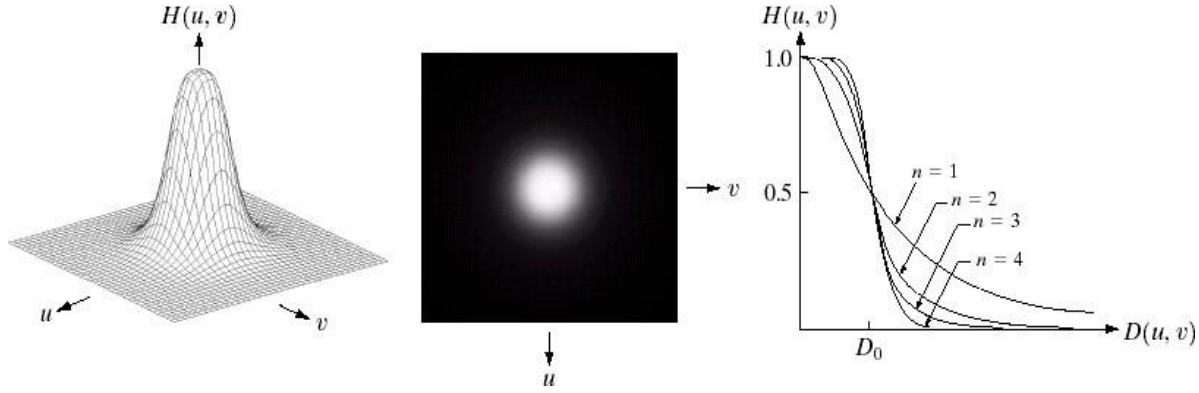


Fig. 2.21 Butterworth Low Pass filter

Gaussian Low Pass Filter

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$

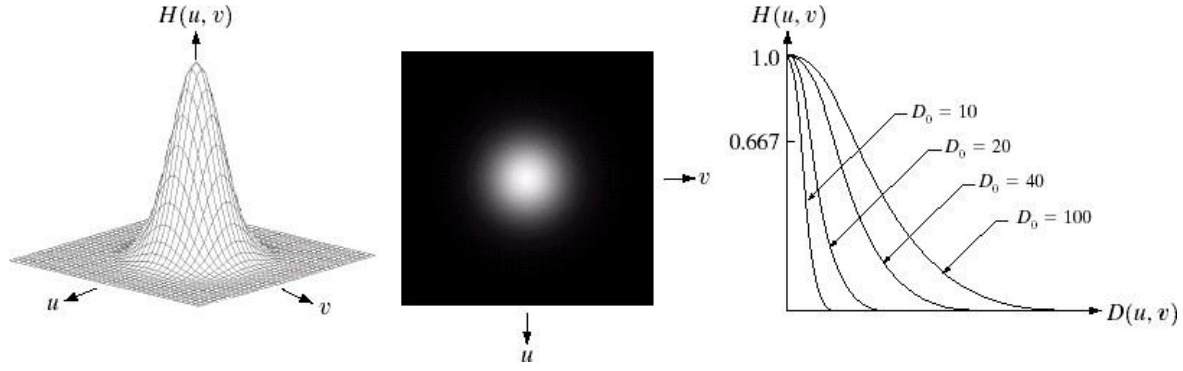


Fig. 2.22 Gaussian Low Pass filter

Frequency Domain Filters - Sharpening

- Image details corresponds to high-frequency
    - Sharpening: high-pass filters
    - $H_{hp}(u,v)=1-H_{lp}(u,v)$

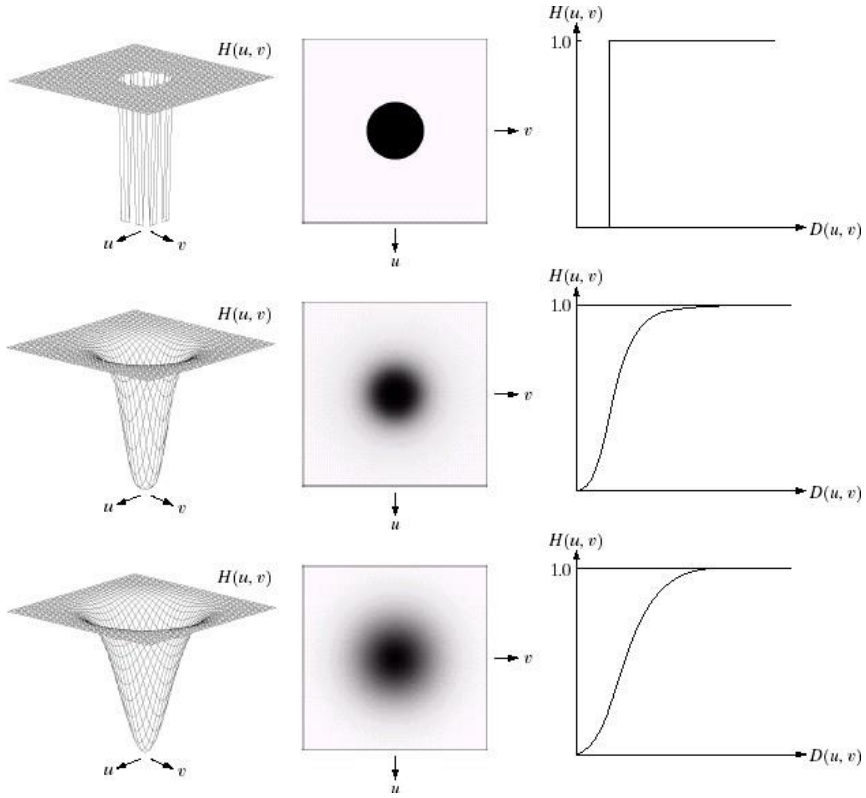


Fig. 2.23 High Pass filter a) Ideal b) Butterworth c) Gaussian

TEXT / REFERENCE BOOKS

1.Rafael C. Gonzalez, Richard E. Woods," Digital Image Processing", Pearson , Second Edition, 2004
2. Anil K. Jain, "Fundamentals of Digital Image Processin", Pearson 2002
3. Robert J.Schalkoff, "Pattern Recognition Statistical, Structural and Neural Approaches",John Wiley & Sons Inc., New York, 1992
4. R.O.Duda, P.E.Hart and D.G.Stork, "Pattern Classification", John Wiley, 2001
5. Himanshu Singh. "Practical Machine Learning and Image Processing", Apress, 2019
6. François Chollet "Deep Learning with Python", Manning Publications Co., NY, 2018
7. Phil Kim. "MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence", Apress, 2017
8. Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images
9. Valentina Zharkova, Lakhmi C. Jain, "Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images", Springer, 2007

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

UNIT – III – Digital Image Processing for Real Time Applications – SEC1628

III. Morphological Image Processing & Segmentation

Morphological Image Processing - Logic Operations involving Binary Images; Dilation and Erosion; Opening and Closing; Basic Morphological Algorithms - Boundary Extraction, Region Filling, Thickening, Thinning; Image Segmentation - Detection of Discontinuities; Edge Linking; Boundary Detection; Thresholding - Global and Adaptive; Region based Segmentation. Case study: Develop a program for segmentation of an objects from the background and transfer them from one image to another.

Logic Operations involving Binary Images

Mathematical Morphology is based on the algebra of non-linear operators operating on object shape and in many respects supersedes the linear algebraic system of convolution. It performs in many tasks – pre-processing, segmentation using object shape, and object quantification – better and more quickly than the standard approach. Mathematical morphology tool is different from the usual standard algebra and calculus. Morphology tools are implemented in most advanced image analysis.

Mathematical morphology is very often used in applications where shape of objects and speed is an issue—example: analysis of microscopic images, industrial inspection, optical character recognition, and document analysis. The non-morphological approach to image processing is close to calculus, being based on the point spread function concept and linear transformations such as convolution. Mathematical morphology uses tools of non-linear algebra and operates with point sets, their connectivity and shape. Morphology operations simplify images, and quantify and preserve the main shape characteristics of objects. Morphological operations are used for the following purpose:

- Image pre-processing (noise filtering, shape simplification)
- Enhancing object structure (skeleton zing, thinning, thickening, convex hull, object marking)
- Segmenting objects from the background
- Quantitative description of objects (area, perimeter, projections, Euler-Poincare characteristics)

Mathematical morphology exploits point set properties, results of integral geometry, and topology. The real image can be modelled using point sets of any dimension; the Euclidean 2D space and its system of subsets is a natural domain for planar shape description.

Computer vision uses the digital counterpart of Euclidean space – sets of integer pairs (∈) f or binary image morphology or sets of integer triples (∈) for gray-scale morphology or binary 3D morphology. Discrete grid can be defined if the neighbourhood relation between points is well defined. This representation is suitable for both rectangular and hexagonal grids. A morphological transformation is given by the relation of the image with another small point set B called structuring element. B is expressed with respect to a local origin. Structuring element is a small image-used as a moving window-- whose support delineates pixel neighbourhoods in the image plane. It can be of any shape, size, or connectivity (more than 1 piece, have holes). To apply the morphologic transformation () to the image means that the structuring element B is moved systematically across the entire image. Assume that B is positioned at some point in the image; the pixel in the image corresponding to the representative point O of the structuring element is called the current pixel. The result of the relation between the image X and the structuring element B in the current position is stored in the output image in the current image pixel position.

- **Reflection**

  $\hat{B} = \{w | w = -b, \text{for } b \in B\}$

- **Translation**

  $(B)_Z = \{c | c = b + z, \text{for } b \in B\}$

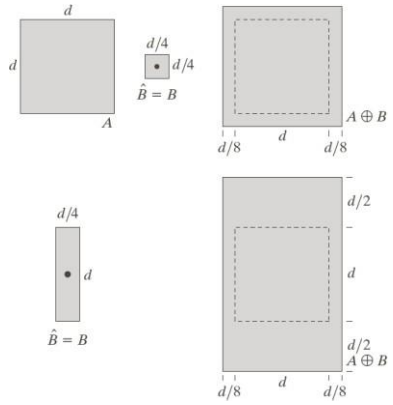
Dilation and Erosion

Dilation

With A and B as sets in Z^2, the dilation of A by B, denoted $A \oplus B$, is defined as

$A \oplus B = \{z | (B\hat{})z \cap A \neq \emptyset\}$

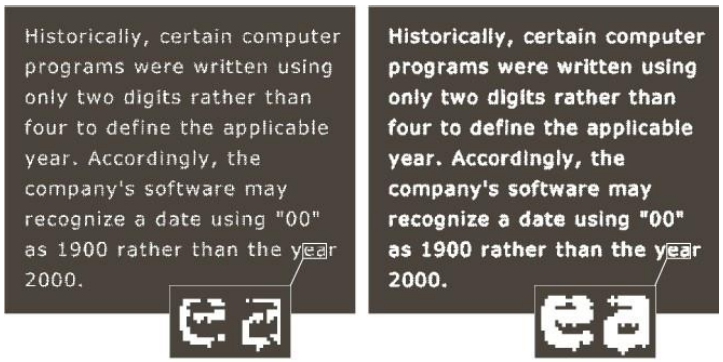
$$A \oplus B = \left\{ z \middle| \left[(\hat{B})_z \cap A \right] \subseteq A \right\}$$



a b c
d e

FIGURE 9.5 (a) Set A. (b) Square structuring element (the dot denotes the origin). (c) Dilation of A by B, shown shaded. (d) Elongated structuring element. (e) Dilation of A using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference

4.1



0	1	0
1	1	1
0	1	0

a c
b

FIGURE 4.2
(a) Sample text of poor resolution with broken characters (see magnified view).
(b) Structuring element.
(c) Dilation of (a) by (b). Broken segments were joined.

Erosion

With A and B as sets in Z^2, the erosion of A by B, denoted $A \ominus B$, defined

$$A \ominus B = \{z | (B)_z \subseteq A\}$$

The set of all points z such that B, translated by z, is contained by A.

$$A \ominus B = \left\{ z \mid (B)_z \cap A^c = \varnothing \right\}$$

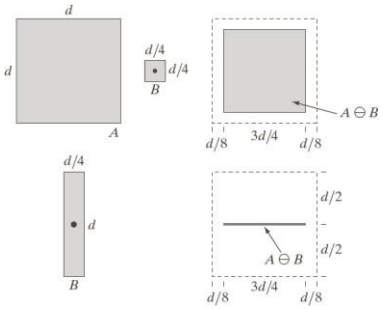
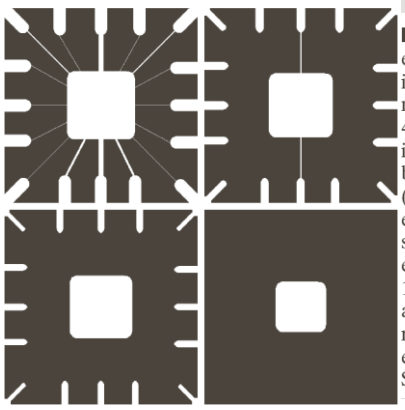


FIGURE 4.3 (a) Set A. (b) Square structuring element, B. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A by B using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference.



a b
c d
FIGURE 4.4 Using erosion to remove image components. (a) A 486 × 486 binary image of a wire-bond mask. (b)–(d) Image eroded using square structuring elements of sizes 11 × 11, 15 × 15, and 45 × 45, respectively. The elements of the SEs were all 1s.

Opening and Closing

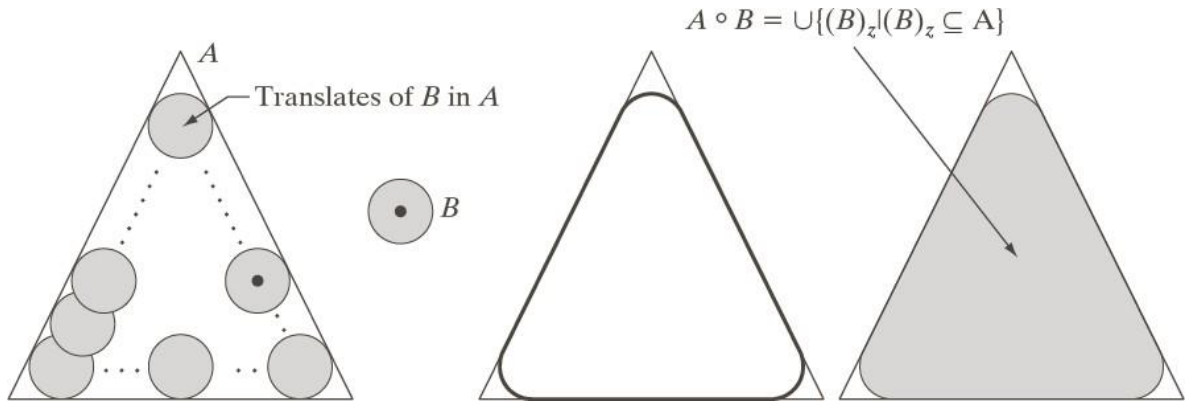
Opening generally smooths the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions. Closing tends to smooth sections of contours but it generates f uses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour.

The opening of set A by structuring element B,denoted $A \circ B$, is defined as

$$A \circ B = (A - B) \oplus B$$

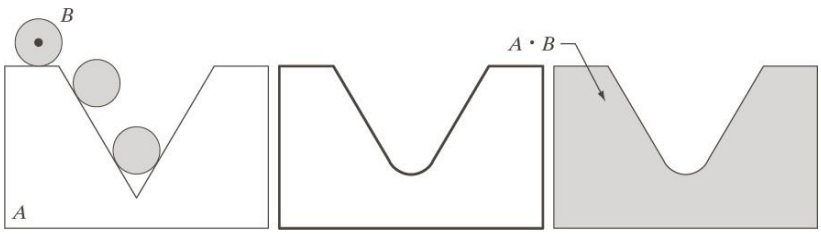
The closing of set A by structuring element B, denoted $A \bullet B$, is defined as

$$A \bullet B = (A \oplus B) - B$$



a b c d

FIGURE 4.5 (a) Structuring element B "rolling" along the inner boundary of A (the dot indicates the origin of B). (b) Structuring element. (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded). We did not shade A in (a) for clarity.



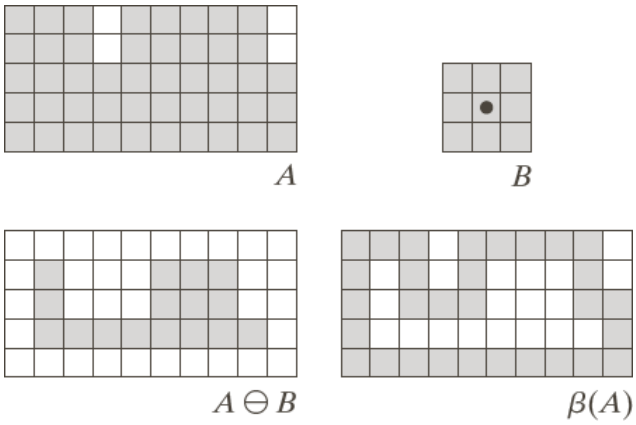
a b c

FIGURE 4.6 (a) Structuring element B "rolling" on the outer boundary of set A. (b) The heavy line is the outer boundary of the closing. (c) Complete closing (shaded). We did not shade A in (a) for clarity.

Boundary Extraction

The boundary of a set A, can be obtained by first eroding A by B and then performing the set difference between A and its erosion.

$$\beta(A) = A - (A - B)$$



a b
c d

FIGURE 4.7 (a) Set *A*. (b) Structuring element *B*. (c) *A* eroded by *B*. (d) Boundary, given by the set difference between *A* and its erosion.

a b

FIGURE 4.8 (a) A simple binary image, with 1s represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

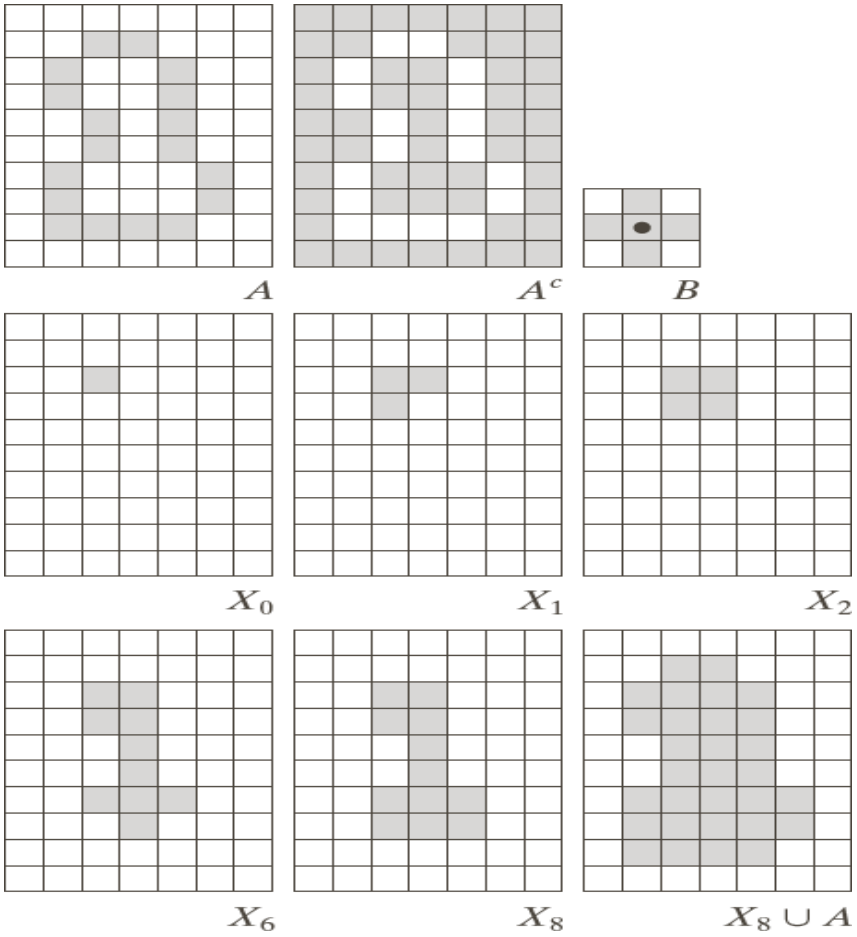
Region Filling

A hole may be defined as a background region surrounded by a connected border of foreground pixels. Let A denote a set whose elements are 8-connected boundaries, each boundary enclosing a background region (i.e., a hole). Given a point in each hole, the objective is to fill all the holes with 1s.

1. Forming an array X_0 of 0s (the same size as the array containing A), except the locations in X_0 corresponding to the given point in each hole, which we set to 1.

2. $X_k = (X_{k-1} + B)$ A^c k=1,2,3,…

Stop the iteration if $X_k = X_{k-1}$



a b c
d e f
g h i

FIGURE 4.9 Hole filling. (a) Set A (shown shaded). (b) Complement of A. (c) Structuring element B. (d) Initial point inside the boundary. (e)–(h) Various steps of Eq. (9.5-2). (i) Final result [union of (a) and (h)].

Thickening

The thickening is defined by the expression

$A \odot B = A \cup (A * B)$

The thickening of A by a sequence of structuring element $\{B\}$ $A \odot \{B\} = ((...((A \odot B\text{^}1) \odot B\text{^}2)...) \odot B\text{^}n)$

In practice, the usual procedure is to thin the background of the set and then complement the result.

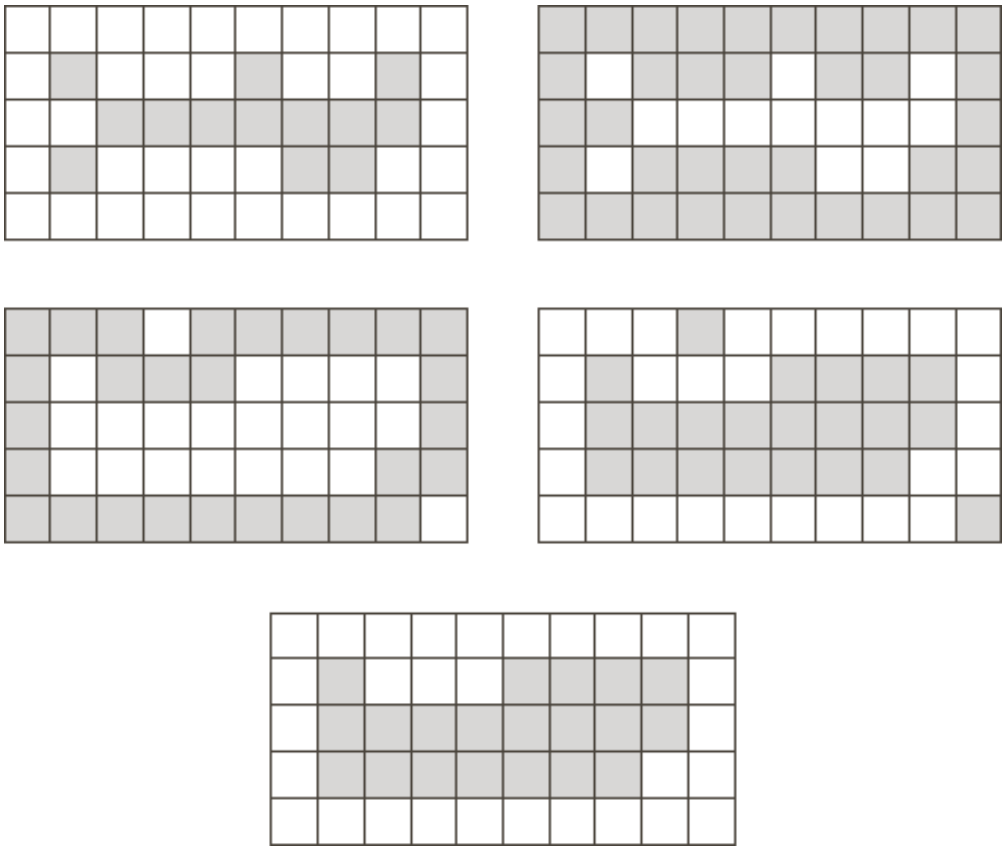


FIGURE 9.12 (a) Set A. (b) Complement of A. (c) Result of thinning the complement of A. (d) Thickened set obtained by complementing (c). (e) Final result, with no disconnected points.

Thinning

The thinning of a set A by a structuring element B, defined

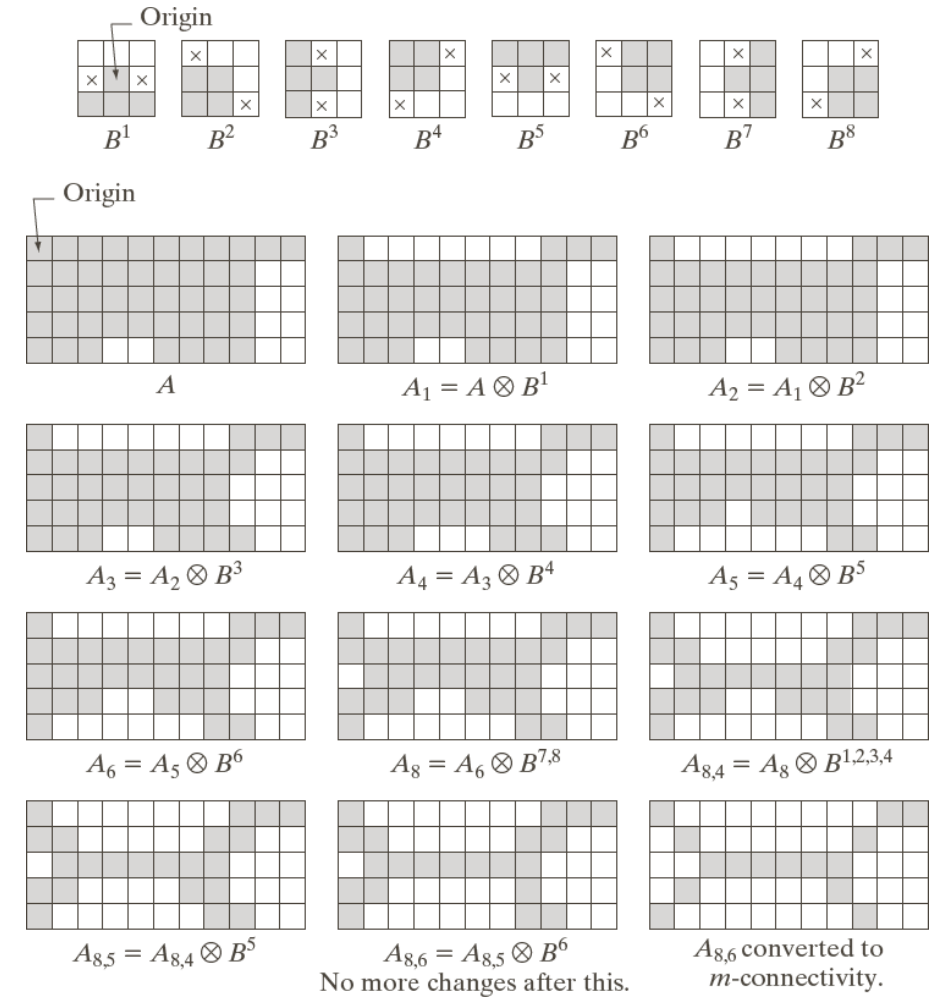
$$A \otimes B = A - (A * B)$$
$$= A \cap (A * B)^c$$

$$\{B\} = \{B^1, B^2, B^3, ..., B^n\}$$

where B^i is a rotated version of B^{i-1}

The thinning of A by a sequence of structuring element $\{B\}$

$$A \otimes \{B\} = ((...((A \otimes B^1) \otimes B^2)...) \otimes B^n)$$



a
b c d
e f g
h i j
k l m

FIGURE 4.11 (a) Sequence of rotated structuring elements used for thinning. (b) Set A. (c) Result of thinning with the first element. (d)–(i) Results of thinning with the next seven elements (there was no change between the seventh and eighth elements). (j) Result of using the first four elements again. (l) Result after convergence. (m) Conversion to m-connectivity.

Image Segmentation

Image segmentation divides an image into regions that are connected and have some similarity within the region and some difference between adjacent regions. The goal is usually to find individual objects in an image. For the most part there are fundamentally two kinds of approaches to segmentation: discontinuity and similarity.

- Similarity may be due to pixel intensity, color or texture.
- Differences are sudden changes (discontinuities) in any of these, but especially sudden changes in intensity along a boundary line, which is called an edge.

There are three kinds of discontinuities of intensity: points, lines and edges. The most

common way to look for discontinuities is to scan a small mask over the image. The mask determines which kind of discontinuity to look for. Only slightly more common than point detection is to find one pixel wide line in an image. For digital images the only three point straight lines are only horizontal, vertical, or diagonal (+ or -45^0).

Edge Linking & Boundary Detection

Two properties of edge points are useful for edge linking:

- the strength (or magnitude) of the detected edge points
- their directions (determined from gradient directions)

- This is usually done in local neighbourhoods.
- Adjacent edge points with similar magnitude and direction are linked.
- For example, an edge pixel with coordinates (x_0,y_0) in a predefined neighbourhood of (x,y) is similar to the pixel at (x,y) if

$$|\nabla f(x, y) - \nabla(x_0, y_0)| \le E, \quad E : \text{a nonnegative threshold}$$

$$|\alpha(x, y) - \alpha(x_0, y_0)| < A, \quad A : \text{a nonegative angle threshold}$$

Hough transform: a way of finding edge points in an image that lie along a straight line. Example: *xy*-plane v.s. *ab*-plane (parameter space)

$$y_I = ax_I + b$$

- The Hough transform consists of finding all pairs of values of θ and ρ which satisfy the

  equations that pass through (x,y).
- These are accumulated in what is basically a 2-dimensional histogram.
- When plotted these pairs of θ and ρ will look like a sine wave. The process is repeated for all appropriate (x,y) locations.

Thresholding

Global – T depends only on gray level values

Local – T depends on both gray level values and local property

Dynamic or Adaptive – T depends on spatial coordinates

Different approaches possible in Graylevel threshold

- Interactive threshold

- Adaptive threshold
- Minimisation method

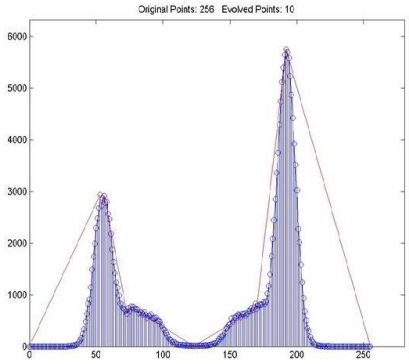


Fig. 4.11 Gray level thresholding

Region based Segmentation

- Edges and thresholds sometimes do not give good results for segmentation.
- Region-based segmentation is based on the connectivity of similar pixels in a region.
  - Each region must be uniform.
  - Connectivity of the pixels within the region is very important.
- There are two main approaches to region-based segmentation: region growing and region splitting.

Basic Formulation

- Let R represent the entire image region.

For example: $P(R_k)$=TRUE if all pixels in R_k have the same gray level. Region splitting is the opposite of region growing.

  - First there is a large region (possible the entire image).
  - Then a predicate (measurement) is used to determine if the region is uniform.
  - If not, then the method requires that the region be split into two regions.
  - Then each of these two regions is independently tested by the predicate (measurement).
  - This procedure continues until all resulting regions are uniform.

The main problem with region splitting is determining where to split a region. One method to divide a region is to use a quad tree structure. Quadtree: a tree in which nodes have exactly four descendants. The split and merge procedure:

  - Split into four disjoint quadrants any region R_i for which $P(R_i)$ =FALSE.
  - Merge any adjacent regions R_j and R_k for which $P(R_j \cup R_k)$ = TRUE. (the

quadtree structure may not be preserved)

– Stop when no further merging or splitting is possible.

TEXT / REFERENCE BOOKS

1. Rafael C. Gonzalez, Richard E. Woods," Digital Image Processing", Pearson , Second Edition, 2004

2. Anil K. Jain, "Fundamentals of Digital Image Processin", Pearson 2002

3. Robert J.Schalkoff, "Pattern Recognition Statistical, Structural and Neural Approaches",John Wiley & Sons Inc., New York, 1992

4. R.O.Duda, P.E.Hart and D.G.Stork, "Pattern Classification", John Wiley, 2001

5. Himanshu Singh. "Practical Machine Learning and Image Processing", Apress, 2019

6. François Chollet "Deep Learning with Python", Manning Publications Co., NY, 2018

7. Phil Kim. "MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence", Apress, 2017

8. Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images

9. Valentina Zharkova, Lakhmi C. Jain, "Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images", Springer, 2007

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

UNIT – IV – Digital Image Processing for Real Time Applications – SEC1628

IV. PATTERN CLASSIFIER

Overview of Pattern recognition – Discriminant functions – Supervised learning – Parametric estimation – Maximum Likelihood Estimation – Bayesian parameter Estimation– Problems with Bayes approach– Pattern classification by distance functions –Minimum distance pattern classifier, Template matching - probabilistic approach- K-nearnest neighbour (KNN), Path Forest- Fuzzy logic – Fuzzy Pattern Classifiers, Case study: Fuzzy clustering algorithm.

Overview of Pattern recognition

Pattern is everything around in this digital world. A pattern can either be seen physically or it can be observed mathematically by applying algorithms.

Example: The colours on the clothes, speech pattern etc. In computer science, a pattern is represented using vector feature values.

What is Pattern Recognition?

Pattern recognition is the process of recognizing patterns by using machine learning algorithm. Pattern recognition can be defined as the classification of data based on knowledge already gained or on statistical information extracted from patterns and/or their representation. One of the important aspects of the pattern recognition is its application potential.

Examples: Speech recognition, speaker identification, multimedia document recognition (MDR), automatic medical diagnosis.

In a typical pattern recognition application, the raw data is processed and converted into a form that is amenable for a machine to use. Pattern recognition involves classification and cluster of patterns.

- In classification, an appropriate class label is assigned to a pattern based on an abstraction that is generated using a set of training patterns or domain knowledge. Classification is used in supervised learning.
- Clustering generated a partition of the data which helps decision making, the specific decision making activity of interest to us. Clustering is used in an unsupervised learning.

Features may be represented as continuous, discrete or discrete binary variables. A feature is a function of one or more measurements, computed so that it quantifies some significant characteristics of the object.

Example: consider our face then eyes, ears, nose etc are features of the face. A set of features that are taken together, forms the **features vector**.

Example: In the above example of face, if all the features (eyes, ears, nose etc) taken together then the sequence is feature vector([eyes, ears, nose]). Feature vector is the sequence of a features represented as a d-dimensional column vector. In case of speech, MFCC (Melfrequency Cepstral Coefficient) is the spectral features of the speech. Sequence of first 13 features forms a feature vector.

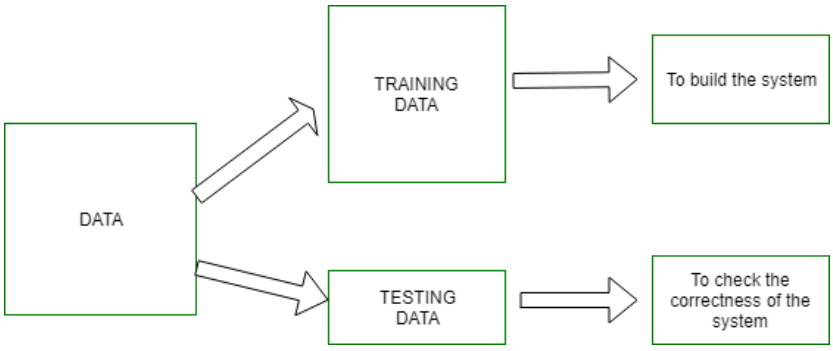
Pattern recognition possesses the following features:

- Pattern recognition system should recognise familiar pattern quickly and accurate
- Recognize and classify unfamiliar objects
- Accurately recognize shapes and objects from different angles
- Identify patterns and objects even when partly hidden
- Recognize patterns quickly with ease, and with automaticity.

Training and Learning in Pattern Recognition

Learning is a phenomena through which a system gets trained and becomes adaptable to give result in an accurate manner. Learning is the most important phase as how well the system performs on the data provided to the system depends on which algorithms used on the data. Entire dataset is divided into two categories, one which is used in training the model i.e. Training set and the other that is used in testing the model after training, i.e. Testing set.

- **Training set:**
  Training set is used to build a model. It consists of the set of images that are used to train the system. Training rules and algorithms used give relevant information on how to associate input data with output decision. The system is trained by applying these algorithms on the dataset, all the relevant information is extracted from the data and results are obtained. Generally, 80% of the data of the dataset is taken for training data.
- **Testing set:**
  Testing data is used to test the system. It is the set of data which is used to verify whether the system is producing the correct output after being trained or not. Generally, 20% of the data of the dataset is used for testing. Testing data is used to measure the accuracy of the system. Example: a system which identifies which category a particular flower belongs to, is able to identify seven category of flowers correctly out of ten and rest others wrong, then the accuracy is 70 %.



Real-time Examples and Explanations:

A pattern is a physical object or an abstract notion. While talking about the classes of animals, a description of an animal would be a pattern. While talking about various types of balls, then a description of a ball is a pattern. In the case balls considered as pattern, the classes could be football, cricket ball, table tennis ball etc. Given a new pattern, the class of the pattern is to be determined. The choice of attributes and representation of patterns is a very important step in pattern classification. A good representation is one which makes use of discriminating attributes and also reduces the computational burden in pattern classification.

An obvious representation of a pattern will be a **vector**. Each element of the vector can represent one attribute of the pattern. The first element of the vector will contain the value of the first attribute for the pattern being considered.

Example: While representing spherical objects, (25, 1) may be represented as an spherical object with 25 units of weight and 1 unit diameter. The class label can form a part of the vector. If spherical objects belong to class 1, the vector would be (25, 1, 1), where the first element represents the weight of the object, the second element, the diameter of the object and the third element represents the class of the object.

Advantages:

- Pattern recognition solves classification problems
- Pattern recognition solves the problem of fake bio metric detection.
- It is useful for cloth pattern recognition for visually impaired blind people.
- It helps in speaker diarization.
- We can recognise particular object from different angle.

Disadvantages:

- Syntactic Pattern recognition approach is complex to implement and it is very slow process.
- Sometime to get better accuracy, larger dataset is required.
- It cannot explain why a particular object is recognized. Example: my face vs my friend's face.

Applications:

- **Image processing, segmentation and analysis**
  Pattern recognition is used to give human recognition intelligence to machine which is required in image processing.
- **Computer vision**
  Pattern recognition is used to extract meaningful features from given image/video samples and is used in computer vision for various applications like biological and biomedical imaging.
- **Seismic analysis**
  Pattern recognition approach is used for the discovery, imaging, and interpretation of temporal patterns in seismic array recordings. Statistical pattern recognition is implemented and used in different types of seismic analysis models.
- **Radar signal classification/analysis**
  Pattern recognition and Signal processing methods are used in various applications of radar signal classifications like AP mine detection and identification.
- **Speech recognition**
  The greatest success in speech recognition has been obtained using pattern recognition paradigms. It is used in various algorithms of speech recognition which tries to avoid the problems of using a phoneme level of description and treats larger units such as words as pattern
- **Finger print identification**
  The fingerprint recognition technique is a dominant technology in the biometric market. A number of recognition methods have been used to perform fingerprint matching out of which pattern recognition approaches is widely used.

Phases in Pattern Recognition System

Approaches for Pattern Recognition Systems can be represented by different phases as Pattern Recognition Systems can be divided into components.

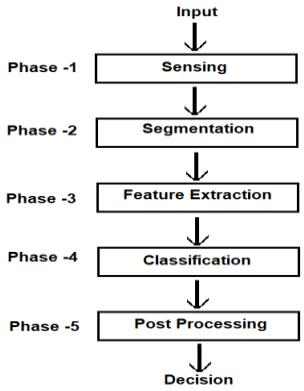
Phase 1: Converts images or sounds or other inputs into signal data.

Phase 2: Isolates the sensed objects from the background.

Phase 3: Measures objects properties that are useful for classification.

Phase 4: Assign the sensed object to category.

Phase 5: Take other consideration to decide for appropriate action.



Phases in Pattern Recognition

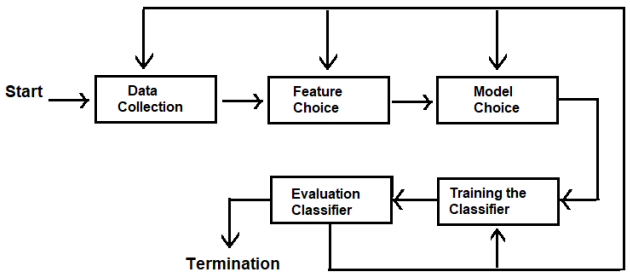
Problems solved by these Phases are as follows:

1. **Sensing**: It deals with problem arises in the input such as its bandwidth, resolution, sensitivity, distortion, signal-to-noise ratio, latency, etc.
2. **Segmentation and Grouping**: Deepest problems in pattern recognition that deals with the problem of recognizing or grouping together the various parts of an object.
3. **Feature Extraction**: It deals with the characterization of an object so that it can be recognized easily by measurements. Those objects whose values are very similar for the objects are consider to be in the same category, while whose values are very different for the objects are placed in different categories.
4. **Classification**: It deals with assigning the object to their particular categories by using the feature vector provided by the feature extractor and determining the values of all of the features for a particular input.
5. **Post Processing**: It deals with action decision making by using the output of the classifier. Action such as to minimum-error-rate classification that will minimize the total expected cost.

   Activities for designing the Pattern Recognition Systems

   There are various sequences of activities that are used for designing the Pattern Recognition Systems. These activities are as follows:

   - Data Collection
   - Feature Choice
   - Model Choice
   - Training
   - Evaluation

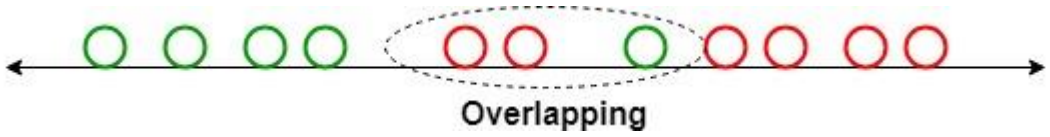


Activity Cycle

Linear Discriminant Analysis or **Normal Discriminant Analysis** or **Discriminant Function Analysis**

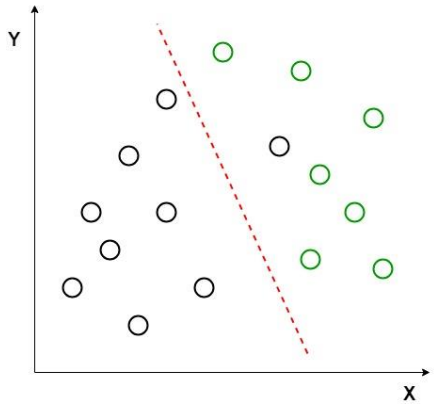
It is a dimensionality reduction technique which is commonly used for the supervised classification problems. It is used for modelling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space.

For example, we have two classes and we need to separate them efficiently. Classes can have multiple features. Using only a single feature to classify them may result in some overlapping as shown in the below figure. So, we will keep on increasing the number of features for proper classification.



Example:

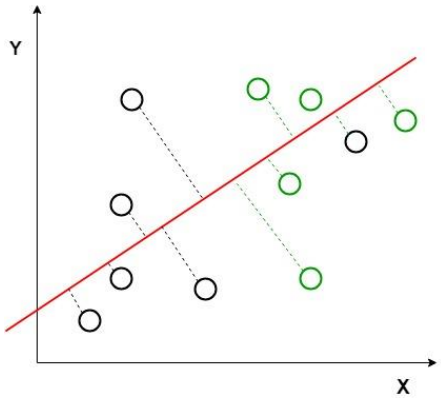
Suppose we have two sets of data points belonging to two different classes that we want to classify. As shown in the given 2D graph, when the data points are plotted on the 2D plane, there's no straight line that can separate the two classes of the data points completely. Hence, in this case, LDA (Linear Discriminant Analysis) is used which reduces the 2D graph into a 1D graph in order to maximize the separability between the two classes.



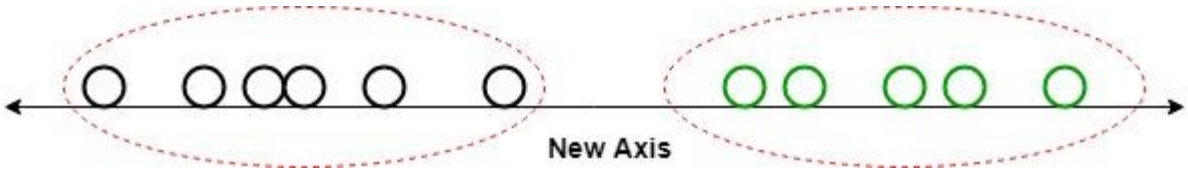
Here, Linear Discriminant Analysis uses both the axes (X and Y) to create a new axis and projects data onto a new axis in a way to maximize the separation of the two categories and hence, reducing the 2D graph into a 1D graph.

Two criteria are used by LDA to create a new axis:

1. Maximize the distance between means of the two classes.
2. Minimize the variation within each class.



In the above graph, it can be seen that a new axis (in red) is generated and plotted in the 2D graph such that it maximizes the distance between the means of the two classes and minimizes the variation within each class. In simple terms, this newly generated axis increases the separation between the dtla points of the two classes. After generating this new axis using the above-mentioned criteria, all the data points of the classes are plotted on this new axis and are shown in the figure given below.



But Linear Discriminant Analysis fails when the mean of the distributions are shared, as it becomes impossible for LDA to find a new axis that makes both the classes linearly separable. In such cases, we use non-linear discriminant analysis.

Extensions to LDA:

1. **Quadratic Discriminant Analysis (QDA):** Each class uses its own estimate of variance (or covariance when there are multiple input variables).
2. **Flexible Discriminant Analysis (FDA):** Where non-linear combinations of inputs is used such as splines.
3. **Regularized Discriminant Analysis (RDA):** Introduces regularization into the estimate of the variance (actually covariance), moderating the influence of different variables on LDA.

Applications:

1. **Face Recognition:** In the field of Computer Vision, face recognition is a very popular application in which each face is represented by a very large number of pixel values. Linear discriminant analysis (LDA) is used here to reduce the number of features to a more manageable number before the process of classification. Each of the new dimensions generated is a linear combination of pixel values, which form a template. The linear combinations obtained using Fisher's linear discriminant are called Fisher faces.
2. **Medical:** In this field, Linear discriminant analysis (LDA) is used to classify the patient disease state as mild, moderate or severe based upon the patient various

parameters and the medical treatment he is going through. This helps the doctors to intensify or reduce the pace of their treatment.

3. **Customer Identification:** Suppose we want to identify the type of customers which are most likely to buy a particular product in a shopping mall. By doing a simple question and answers survey, we can gather all the features of the customers. Here, Linear discriminant analysis will help us to identify and select the features which can describe the characteristics of the group of customers that are most likely to buy that particular product in the shopping mall.

Supervised Learning

What is supervised learning?

Supervised learning, also known as supervised machine learning, is a subcategory of machine learning and artificial intelligence. It is defined by its use of labelled datasets to train algorithms that to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately, which occurs as part of the cross validation process. Supervised learning helps organizations solve for a variety of real-world problems at scale, such as classifying spam in a separate folder from your inbox.

How supervised learning works

Supervised learning uses a training set to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.

Supervised learning can be separated into two types of problems when data mining—classification and regression:

Classification uses an algorithm to accurately assign test data into specific categories. It recognizes specific entities within the dataset and attempts to draw some conclusions on how those entities should be labelled or defined. Common classification algorithms are linear classifiers, support vector machines (SVM), decision trees, k-nearest neighbour, and random forest, which are described in more detail below.

Regression is used to understand the relationship between dependent and independent variables. It is commonly used to make projections, such as for sales revenue for a given business. Linear regression, logistical regression, and polynomial regression are popular regression algorithms.

Supervised learning algorithms

Various algorithms and computation techniques are used in supervised machine learning processes. Below are brief explanations of some of the most commonly used learning methods,

Neural networks

Primarily leveraged for deep learning algorithms, neural networks process training data by mimicking the interconnectivity of the human brain through layers of nodes. Each node is made up of inputs, weights, a bias (or threshold), and an output. If that output value exceeds a given threshold, it “fires” or activates the node, passing data to the next layer in the

network. Neural networks learn this mapping function through supervised learning, adjusting based on the loss function through the process of gradient descent. When the cost function is at or near zero, we can be confident in the model's accuracy to yield the correct answer.

Naive Bayes

Naive Bayes is classification approach that adopts the principle of class conditional independence from the Bayes Theorem. This means that the presence of one feature does not impact the presence of another in the probability of a given outcome, and each predictor has an equal effect on that result. There are three types of Naïve Bayes classifiers: Multinomial Naïve Bayes, Bernoulli Naïve Bayes, and Gaussian Naïve Bayes. This technique is primarily used in text classification, spam identification, and recommendation systems.

Linear regression

Linear regression is used to identify the relationship between a dependent variable and one or more independent variables and is typically leveraged to make predictions about future outcomes. When there is only one independent variable and one dependent variable, it is known as simple linear regression. As the number of independent variables increases, it is referred to as multiple linear regression. For each type of linear regression, it seeks to plot a line of best fit, which is calculated through the method of least squares. However, unlike other regression models, this line is straight when plotted on a graph.

Logistic regression

While linear regression is leveraged when dependent variables are continuous, logistical regression is selected when the dependent variable is categorical, meaning they have binary outputs, such as "true" and "false" or "yes" and "no." While both regression models seek to understand relationships between data inputs, logistic regression is mainly used to solve binary classification problems, such as spam identification.

Support vector machine (SVM)

A support vector machine is a popular supervised learning model developed by Vladimir Vapnik, used for both data classification and regression. That said, it is typically leveraged for classification problems, constructing a hyperplane where the distance between two classes of data points is at its maximum. This hyperplane is known as the decision boundary, separating the classes of data points (e.g., oranges vs. apples) on either side of the plane.

K-nearest neighbour

K-nearest neighbour, also known as the KNN algorithm, is a non-parametric algorithm that classifies data points based on their proximity and association to other available data. This algorithm assumes that similar data points can be found near each other. As a result, it seeks to calculate the distance between data points, usually through Euclidean distance, and then it assigns a category based on the most frequent category or average.

Its ease of use and low calculation time make it a preferred algorithm by data scientists, but as the test dataset grows, the processing time lengthens, making it less appealing for classification tasks. KNN is typically used for recommendation engines and image recognition.

Random forest

Random forest is another flexible supervised machine learning algorithm used for both classification and regression purposes. The "forest" references a collection of uncorrelated decision trees, which are then merged together to reduce variance and create more accurate data predictions.

Supervised learning examples

Supervised learning models can be used to build and advance a number of business applications, including the following:

- Image- and object-recognition: Supervised learning algorithms can be used to locate, isolate, and categorize objects out of videos or images, making them useful when applied to various computer vision techniques and imagery analysis.
- Predictive analytics: A widespread use case for supervised learning models is in creating predictive analytics systems to provide deep insights into various business data points. This allows enterprises to anticipate certain results based on a given output variable, helping business leaders justify decisions or pivot for the benefit of the organization.
- Customer sentiment analysis: Using supervised machine learning algorithms, organizations can extract and classify important pieces of information from large volumes of data—including context, emotion, and intent—with very little human intervention. This can be incredibly useful when gaining a better understanding of customer interactions and can be used to improve brand engagement efforts.
- Spam detection: Spam detection is another example of a supervised learning model. Using supervised classification algorithms, organizations can train databases to recognize patterns or anomalies in new data to organize spam and non-spam-related correspondences effectively.

Challenges of supervised learning

Although supervised learning can offer businesses advantages, such as deep data insights and improved automation, there are some challenges when building sustainable supervised learning models. The following are some of these challenges:

- Supervised learning models can require certain levels of expertise to structure accurately.
- Training supervised learning models can be very time intensive.
- Datasets can have a higher likelihood of human error, resulting in algorithms learning incorrectly.
- Unlike unsupervised learning models, supervised learning cannot cluster or classify data on its own.

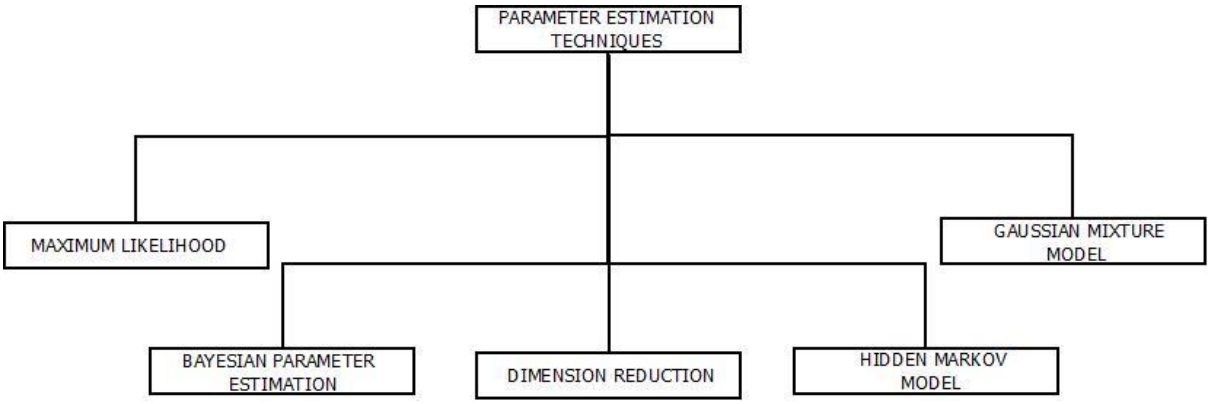
Parameter Estimation: Introduction

In order to estimate the parameters randomly from a given sample distribution data, the technique of parameter estimation is used.

To achieve this, number estimation techniques are available and listed below.

Parameter Estimation Techniques

To implement the estimation process, certain techniques are available including Dimension Reduction, Gaussian Mixture Model etc.



Parameter Estimation: Maximum likelihood Estimation

- Estimation model consists of a number of parameters. So, in order to calculate or estimate the parameters of the model, the concept of Maximum Likelihood is used.
- Whenever the probability density functions of a sample are unknown, they can be calculated by taking the parameters inside sample as quantities having unknown but fixed values.
- In simple words, consider we want to calculate the height of a number of boys in a school. But, it will be a time consuming process to measure the height of all the boys. So, the unknown mean and unknown variance of the heights being distributed normally, by maximum likelihood estimation we can calculate the mean and variance by only measuring the height of a small group of boys from the total sample.

Parameter Estimation: Bayesian Parameters Estimation

- "Parameters" in Bayesian Parameters Estimation are the random variable which comprises of known Priori Distribution.
- The major objective of Bayesian Parameters Estimation is to evaluate how varying parameter affect density estimation.
- The aim is to estimate the posterior density P(Θ/x).
- The above expression generates the final density P(x/X) by integrating the parameters.

$$P(x|X) = \int P(x|\Theta)\, P(\Theta|X)\, d\Theta$$

Bayesian Parameter Estimation

Parameter Estimation: Expectation Maximization (EM)

- Expectation maximization the process that is used for clustering the data sample.
- EM for a given data, has the ability to predict feature values for each class on the basis of classification of examples by learning the theory that specifies it.

- It works on the concept of, starting with the random theory and randomly classified data along with the execution of below mentioned steps.
- Step-1("E") : In this step, Classification of current data using the theory that is currently being used is done.
- Step-2("M") : In this step, With the help of current classification of data, theory for that is generated.
- Thus EM means, Expected classification for each sample is generated used step-1 and theory is generated using step-2.

Problems with Bayes approach

Bayes Theorem

Let X be the pattern whose class label is unknown. Let H_i be some hypothesis which indicates the class to which X belongs. For example, H_i is the hypothesis that a pattern belongs to class C_i. Let us assume that the prior probability of H_i, given by $P(H_i)$, is known. In order to classify X, we need to determine $P(H_i \mid X)$ which is the probability that the hypothesis H_i holds, given the observed pattern X.

$P(H_i \mid X)$ is the posterior probability of H_i conditioned on X. In contrast, $P(H_i)$ is the prior probability, or *apriori* probability, of H_i. It is the probability of the hypothesis

regardless of the value of X. The posterior probability, $P(H_i \mid X)$ is based on X and other information (such as background knowledge), whereas the prior probability, $P(H_i)$ is obtained before observing X.

Similarly, $P(X \mid H_i)$ is the probability of X conditioned on H_i. Note that $P(X)$ is defined as a weighted combination of $P(X \mid H_i)$ and is

$$P(X) = \sum_i P(X \mid H_i)P(H_i)$$

Bayes theorem is useful in that it provides a way of calculating the posterior probability, $P(H_i \mid X)$, from $P(H_i)$, $P(X)$ and $P(X \mid H_i)$. It states

$$P(H_i \mid X) = \frac{P(X \mid H_i)P(H_i)}{P(X)}$$

Example

In a coffee shop, 99% of the customers prefer coffee. The remaining 1% prefer tea. So, P(coffee drinker) = 0.99 and P(tea drinker) = 0.01. In the absence of any other information, we can classify any customer as a coffee drinker and the probability of error is only 0.01; this is because we are classifying a tea drinker also as a coffee drinker.

We can make a better decision if additional information is available. The following example illustrates this.

Example

If the prior probability of H that a road is wet is $P(\mathrm{H}) = 0.3$. Then the probability that a road is not wet is 0.7. If we use only this information, then it is good to decide that a road is not wet. The corresponding probability of error is 0.3.

Let us further say that the probability of rain, $P(\mathrm{X})$, is 0.3. Now if it rains, we need to calculate the posterior probability that the roads are wet, i.e., $P(\mathrm{H} \mid \mathrm{X})$. This can be calculated using Bayes theorem. If 90% of the time when the roads are wet, it is because it has rained

$$P(\text{road is wet} \mid \text{it has rained}) = \frac{P(\mathrm{X} \mid \mathrm{H}) \times P(\mathrm{H})}{P(\mathrm{X})} = \frac{0.9 \times 0.3}{0.3} = 0.9$$

Here, the probability of error is 0.1, which is the probability that a road is not wet given that it has rained.

Here, in addition to the prior probability $P(\mathrm{H})$, we require information about $P(\mathrm{X}|\mathrm{H})$ (P(it has rained|roads are wet)) and $P(\mathrm{X})$ (probability of rain).

Example

Consider the example given in Table Suppose only the first ten patterns are there in the data set. This data set is shown in Table. Consider a new pattern

Table Example training data set

Cook	Mood	Cuisine	Tasty
Sita	Bad	Indian	Yes
Sita	Good	Continental	Yes
Asha	Bad	Indian	No
Asha	Good	Indian	Yes
Usha	Bad	Indian	Yes
Usha	Bad	Continental	No
Asha	Bad	Continental	No
Asha	Good	Continental	Yes
Usha	Good	Indian	Yes
Usha	Good	Continental	No

Cook = Sita, Mood = Bad, Cuisine = Continental

We need to classify this example as Tasty = yes or Tasty = no. Since there are 6 examples out of 10 with Tasty = yes, the prior probability P(Tasty = yes) is $\frac{6}{10}$, which is 0.60. The prior probability P(Tasty = no) is $\frac{4}{10}$ which is 0.40. There are 2 examples with Cook = Sita and Tasty = yes and 0 examples with Cook = Sita and Tasty = no. The probability for Cook = Sita given that Tasty = yes will be

$$P(\text{Cook} = \text{Sita} \mid \text{Tasty} = \text{yes}) = \frac{2}{6} = 0.33$$

$$P(\text{Cook} = \text{Sita} \mid \text{Tasty} = \text{no}) = 0$$

The probability is zero, and since it is multiplied by other probabilities, those probabilities will also be zero. To avoid this, a small value is taken. Let it be 0.01.

There are 2 examples with Mood = Bad and Tasty = yes and 3 examples with Mood = Bad and Tasty = no. Therefore

$$P(\text{Mood} = \text{Bad} \mid \text{Tasty} = \text{yes}) = \frac{2}{6} = 0.33$$

$$P(\text{Mood} = \text{Bad} \mid \text{Tasty} = \text{no}) = \frac{3}{4} = 0.75$$

There are 2 examples with Cuisine = Continental and Tasty = yes and 3 examples with Cuisine = Continental and Tasty = no. Therefore

$$P(\text{Cuisine} = \text{Continental} \mid \text{Tasty} = \text{yes}) = \frac{2}{6} = 0.33$$

$$P(\text{Cuisine} = \text{Continental} \mid \text{Tasty} = \text{no}) = \frac{3}{4} = 0.75$$

Therefore,

$$P(\text{Tasty} = \text{yes} \mid \text{X}) = 0.6 \times 0.33 \times 0.33 \times 0.33 = 0.0216$$

$$P(\text{Tasty} = \text{no} \mid \text{X}) = 0.4 \times 0.01 \times 0.75 \times 0.75 = 0.00225$$

The new pattern is therefore classified as belonging to the class Tasty = yes as $P(\text{Tasty} = \text{yes} \mid \text{X}) > P(\text{Tasty} = \text{no} \mid \text{X})$.

Condensation of the training data

- In supervised learning, the training data is used to carry out the classification.
- A large training data may lead to better classification but requires a longer time for classification and larger space requirements.
- In the case of nearest neighbour classification techniques, every test pattern has to be compared with every pattern in the training data to find its closest neighbour.
- This leads to increase in the classification time.
- In the case of neural networks and decision trees, the large training data may effect the time to design the neural network or decision tree but the classification time is not affected. It is only the design time that goes up.
- In the case where the large training data leads to higher classification time, the training data can be reduced.
- Another option is to reduce the number of features.

Prototype Selection

- Prototypes are samples which represent patterns in the training data.
- The prototype set is either a subset of the training data or samples derived from the training data.
- By putting together some patterns close to each other and representing them by either one of the patterns or an abstraction of the patterns, it may be possible to reduce the training data without sacrificing on the classification accuracy.
- One method is to find the centroid of each class and use it to represent all the patterns in the class. This is called the **Minimal Distance Classifier**. If $X_1, X_2, ..., X_n$ represent the n patterns of a class, then the representative sample (centroid) will be

$$c = \frac{\sum_{i=1}^{n} X_i}{n}$$

- In order to classify a test pattern P, if c_i is the closest centroid to the test pattern P, then P is assigned the class label i.

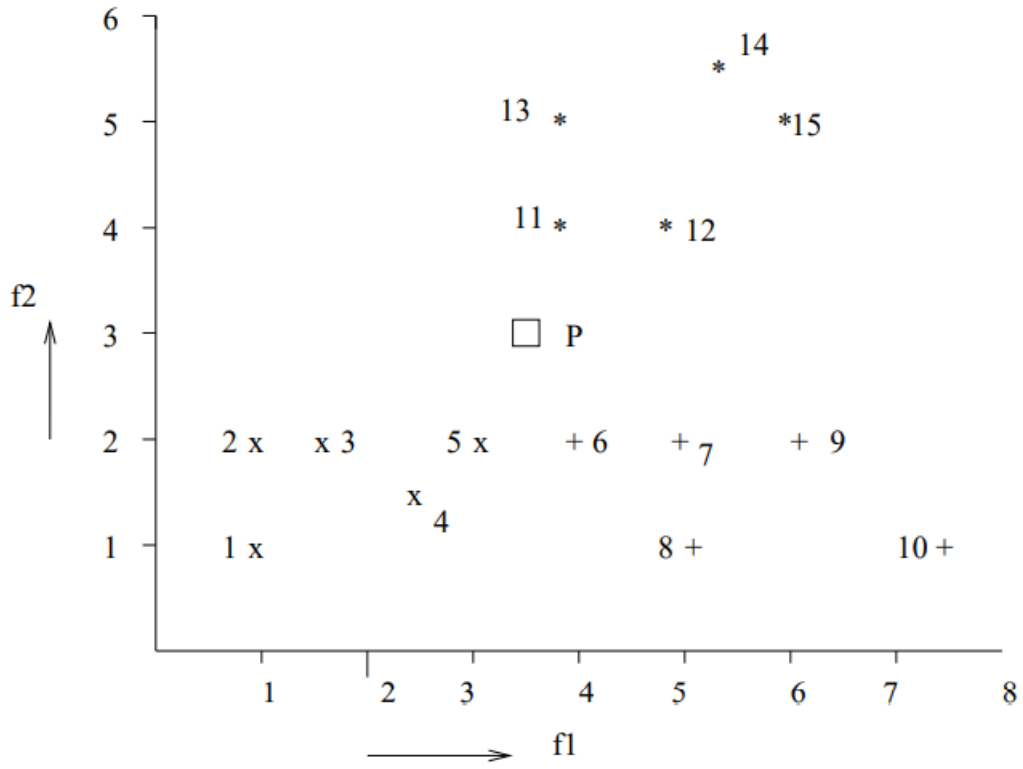


Figure 1: Three class problem

For example, consider Figure 1. Here there are three classes. The two-dimensional patterns are as follows :

Class 1 :

X1 : (1.0,1.0) ;

X2 : (1.0,2.0)

X3 : (1.5,2.0) ;

X4 : (2.5,1.5)

X5 : (3.0,2.0)

Class 2 :

X6 : (4.0,2.0) ;

X7 : (5.0,2.0)

X8 : (5.0,1.0) ;

X9 : (6.0,2.0)

X10 : (7.8,1.0)

Class 3 :

X11 : (4.0,4.0) ;

X12 : (5.0,4.0)

X13 : (4.0,5.0) ;

X14 : (6.0,5.5)

X15 : (6.0,5.0);

The centroid can be got by taking the mean of each co-ordinate. For the patterns in Class 1

$$\text{x-coordinate of centroid} = \frac{1.0+1.0+1.5+2.5+3.0}{5} = 1.8$$

$$\text{y-coordinate of centroid} = \frac{1.0+2.0+2.0+1.5+2.0}{5} = 1.7$$

So, the co-ordinate of centroid for class 1 is

$$c_1 = (1.8, 1.7)$$

For the patterns in Class 2

$$\text{x-coordinate of centroid} = \frac{4.0+5.0+5.0+6.0+7.8}{5} = 5.6$$

$$\text{y-coordinate of centroid} = \frac{2.0+2.0+1.0+2.0+1.0}{5} = 1.6$$

So, the co-ordinate of centroid for class 2 is

$$c_2 = (5.6, 1.6)$$

For the patterns in Class 3

$$\text{x-coordinate of centroid} = \frac{4.0+5.0+4.0+6.0+6.0}{5} = 5.0$$

$$\text{y-coordinate of centroid} = \frac{4.0+4.0+5.0+5.5+5.0}{5} = 4.7$$

So, the co-ordinate of centroid for class 3 is

When we have a test pattern P at (3.5,3.0)

The distance of the centroid of Class 1 from P is

$$d_2(P, c_1) = \sqrt{((3.5 - 1.8)^2 + (3.0 - 1.7)^2)} = 2.14$$

The distance of the centroid of Class 2 from P is

$$d_2(P, c_2) = \sqrt{((3.5 - 5.6)^2 + (3.0 - 1.6)^2)} = 2.52$$

The distance of the centroid of Class 3 from P is

$$d_2(P, c_3) = \sqrt{((3.5 - 5.0)^2 + (3.0 - 4.7)^2)} = 2.27$$

Since the distance of P from the centroid of Class 1 is the smallest of the three distances, P is classified as belonging to Class 1.

To find the closest pattern to P using NN, kNN or mkNN, the distances of P from 15 patterns have to be found. If the minimum distance classifier is used, the distance of P from the centroid of every class has to be found. Here, only three distances have to be computed and the minimum of these found. This will lead to a lot of saving in time especially if the data sets are very large. In that case C << N, where C is the number of classes and N is the total number of patterns.

• The Minimal Distance Classifier gives good results when the patterns in the classes are normally distributed with a diagonal covariance matrix and the variances in different directions are the same. This means that they are isotropic classes.

• If the classes are concentric then the minimum distance classifier will not work. The concentric classes will have almost identical centroids which makes it difficult to classify the test pattern.

For example

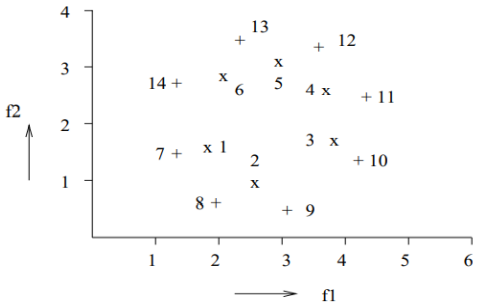


Figure 2: Two concentric classes

Consider the patterns in the two concentric classes shown in Figure 2. The patterns are as follows :

Class 1 :

$X_1 = (1.8, 1.7)$ $X_2 = (2.6, 1.9)$
$X_3 = (4.0, 1.9)$ $X_4 = (3.8, 2.5)$
$X_5 = (3.0, 3.2)$ $X_6 = (2.2, 2.8)$

Class 2 :

$X_7 = (1.3, 1.5)$ $X_8 = (2.0, 0.5)$
$X_9 = ((3.1, 0.3)$ $X_{10} = (4.2, 1.3)$
$X_{11} = (4.3, 2.4)$ $X_{12} = (3.7, 3.3)$
$X_{13} = (2.4, 3.5)$ $X_{14} = (1.3, 2.7)$

The x-coordinate of the centroid of Class 1 is

$$\frac{1.8+2.6+4.0+3.8+3.0+2.2}{6} = 2.9$$

The y-coordinate of the centroid of Class 1 is

$$\frac{1.7+1.9+1.9+2.5+3.2+2.8}{6} = 2.33$$

Therefore, the centroid of class 1 is

$$c_1 = (2.9, 2.33)$$

The x-coordinate of the centroid of Class 2 is

$$\frac{1.3+2.0+3.1+4.2+4.3+3.7+2.4+1.3}{8} = 2.79$$

The y-coordinate of the centroid of Class 2 is

$$\frac{1.5+0.5+0.3+1.3+2.4+3.3+3.5+2.7}{8} = 1.94$$

The centroid of class 2 is

c2 = (2.79, 1.94)

It can be seen that the centroid of class 1 and centroid of class 2 are

very close to each other.

• The main advantage of Minimal Distance Classifier is the time complexity which is O(n) to compute the centroids. To classify a test pattern, the time complexity is O(C) if there are C classes.

• In a similar way the medoid of each class can be used. The medoid of a set of patterns is the most centralized pattern.

• Consider the set of patterns shown in Figure 3. Consider the set of patterns 1-5 shown in the figure. These patterns are:

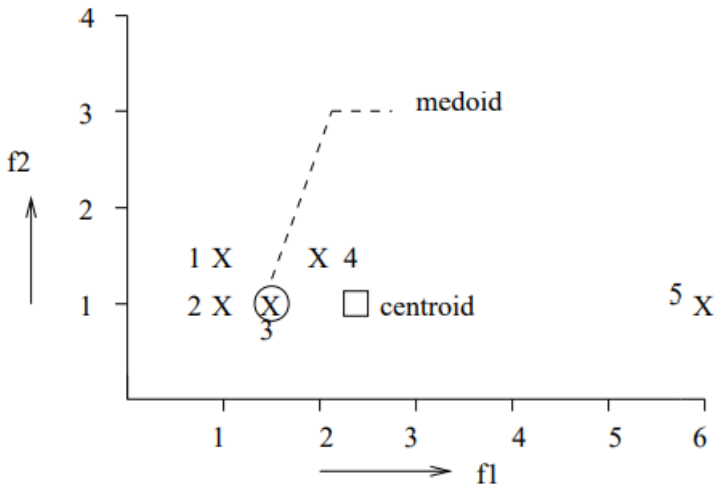


Figure 3: Medoid and Centroid of a set of patterns

Pattern 1 : (1.0,1.5) ; Pattern 2 : (1.0,1.0)
Pattern 3 : (1.5,1.0) ; Pattern 4 : (2.0,1.5)
Pattern 5 : (6.0,1.0)

The centroid of these patterns is

$$c = (\frac{1.0+1.0+1.5+2.0+6.0}{5}, \frac{1.5+1.0+1.0+1.5+1.0}{5})$$
$$= (2.3,1.0)$$

The medoid is the most centrally located pattern and therefore medoid will coincide with pattern 3 having the co-ordinates (1.5,1.0).
The centroid and medoid are shown in Figure 3. It can be seen that the medoid is more representative of the set of patterns than the centroid.
The outlier is causing the centroid to be located between the cluster of patterns 1-4 and the pattern 5. It can be seen that in this case, the medoid is more representative of the set of patterns than the centroid.
• It is also possible to have more than one representative per class.

Image Processing — Template Matching

From our previous articles, we have discussed several image techniques that can be used to create different segmentation and conversion in the image.

This time, how about we talk about an image processing technique that can be used the same was object tracking.

Another useful processing technique as mentioned above is the so-called Template Matching.

Template Matching

Template matching refers to the image processing where we find similar templates in a source image by giving a base template to compare on.

The process of template matching is done by comparing each of the pixel values of the source image one at a time to the template image. The output would be an array of similarity values when compared to the template image.

To be able to look at the similar templates found on the image, we can find the peaks in the resulting array of the template matching.

To better understand this, let us show some examples:

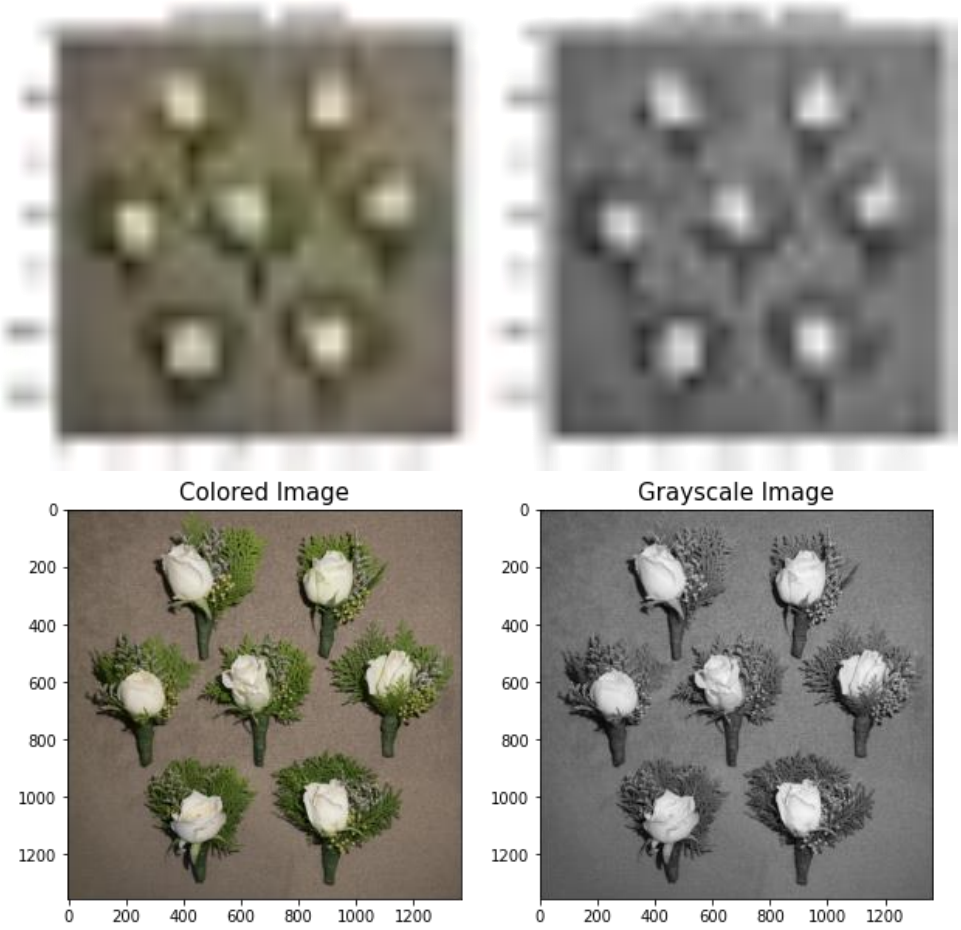


Figure 1: Sample Image

For this article, we will use our old sample of small flower bouquets. We can see that we have 7 small flower bouquets and for this example, we will try to see if our algorithm can detect the other 6 flower bouquets by using one flower bouquet as the template.

To conduct Template matching, we should first identify which patch or template we will use. An example code for patch allocation is seen:

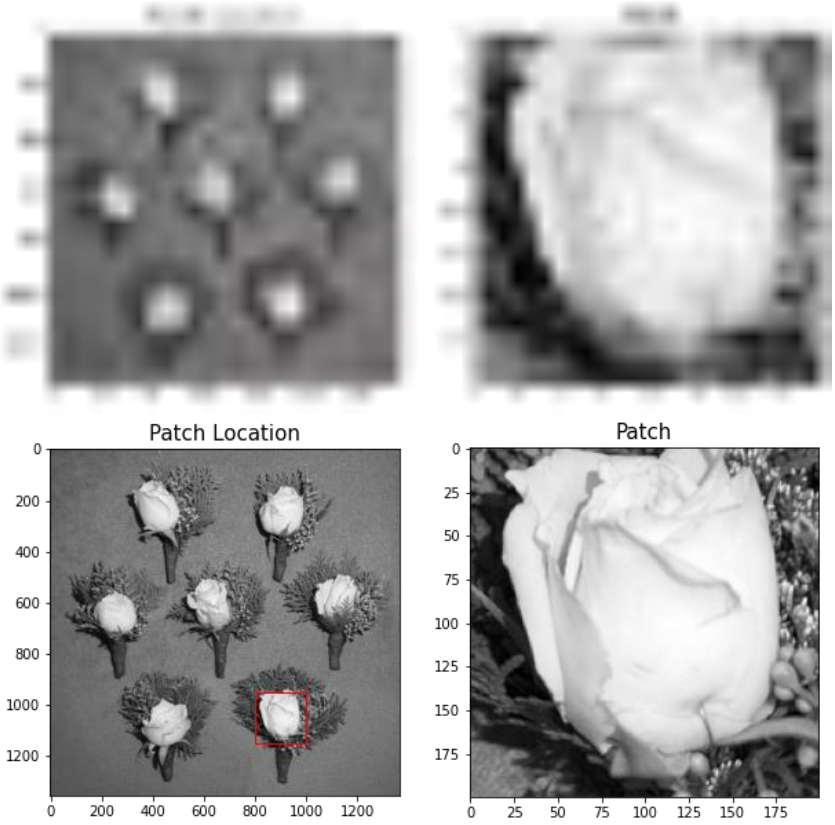


Figure 2: Template/Patch Image

For the trial, let us first use the white flower as the patch where we will compare the rest of the source image/sample image.

Now that we have already a patch that we have identified. We can already pass the source image and patch/template image to the template matching algorithm.

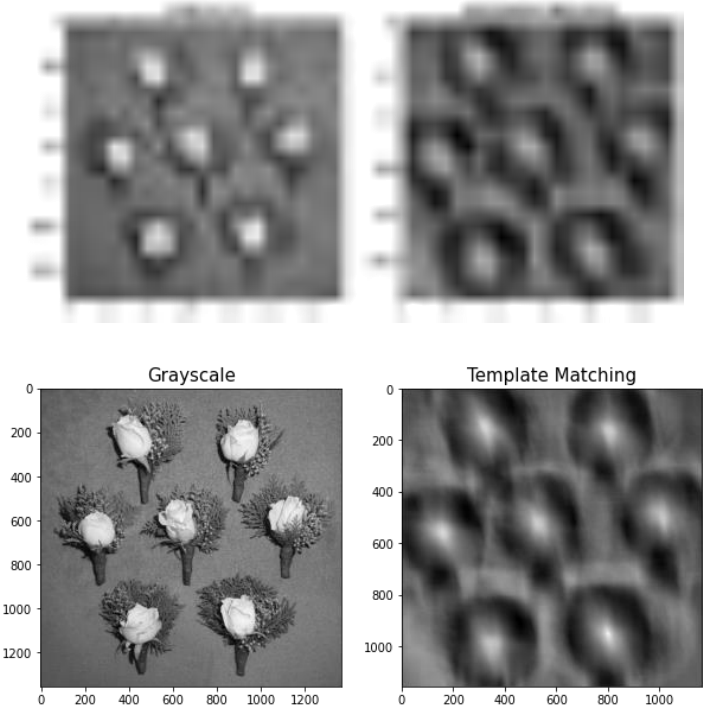


Figure 3: Performing Template Matching

Figure 3 shows the results of the template matching algorithm. Notice how there are bright spots in the graph? These are actually the similarity values that were derived from template matching.
Using the similarity values from the template matching results, we can plot the parts of the source image that has a high similarity from our template image.

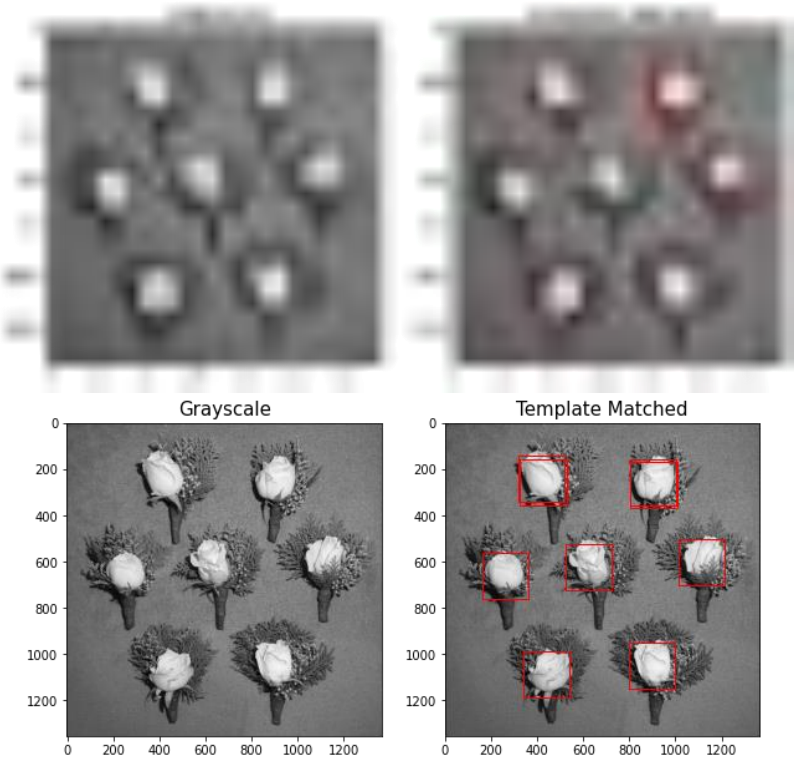
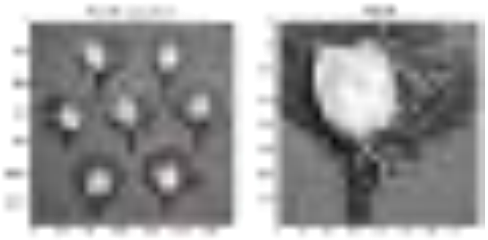


Figure 4: Plotting the Peaks

Figure 4 shows the templates that were detected that are quite similar to our template image. From visual truth, we can already say we are victorious in detecting and finding the white flowers using only one template image.The result is also dependent on the threshold that you have chosen to use. The different threshold would yield different results.

Now let us try a much larger patch!



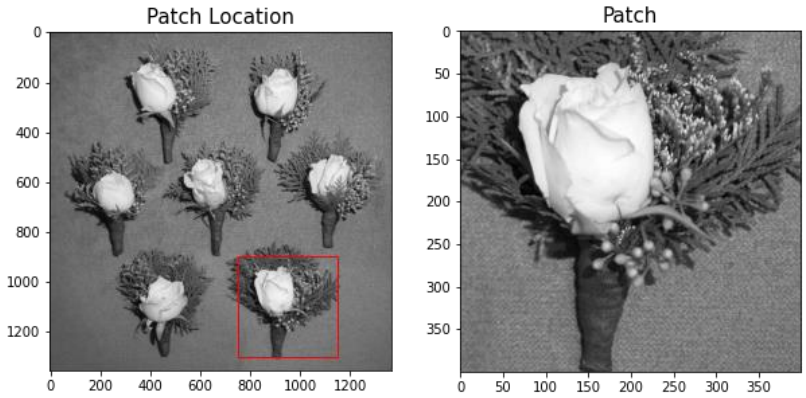


Figure 5: Patch #2

Figure 5 shows that we have chosen a match bigger patch or template to use. This time, we are using the whole flower bouquet as the template image. Let us see if we can also detect the other bouquets.

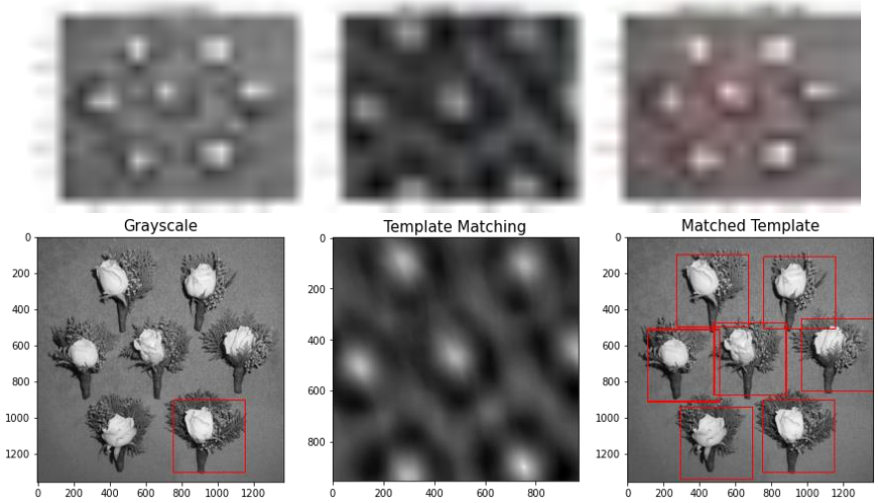


Figure 6: Template Matching Results

Figure 6 shows that indeed, we were able to detect and match the template of one flower bouquet to the different flower bouquets.

we were able to show how we can leverage the use of Image Processing to aid in Object Detection and Object Recognition. Through the examples shown, we were able to successfully detect all the small flower bouquets by only using one of the flower bouquets as a template.

Template matching can be used as a pipeline in conducting object detection for machine learning models and deep learning models.

Pattern Recognition using kNN Algorithm

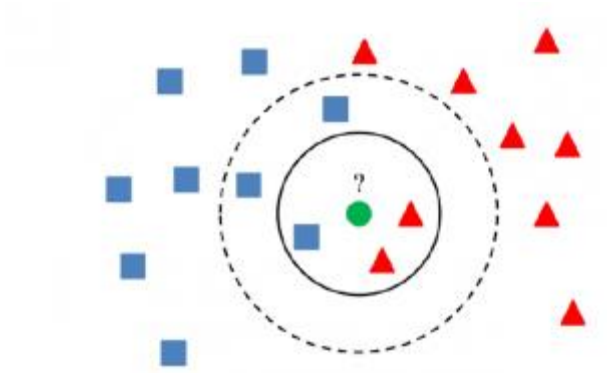
Introduction

k-Nearest Neighbours is a non-parametric classification algorithm. It can be used both for classification and regression. It is essentially a method of classification of the most similar points in the data used for training a model. kNN works best on small data sets that do not have many features. kNN assumes that similar things exist in close proximity. kNN plots the data points on a graph and gauges the similarity between them based on their distance. The performance of kNN is influenced by:

The distance metric used to locate the nearest neighbours.
The distance rule used to derive classifications.
The number of neighbours used to classify new samples (k values)

Concepts and Application of kNN

The kNN algorithm works on the basis of minimum distance from the query instance to the training samples to determine the K-nearest neighbours. In other words, if you are similar to your neighbours, then you are one of them. For example, if apple looks more similar to banana, orange, and melon (fruits) than donkey, dog and cat (animals), then most likely apple is a fruit.



Consider the image shown above, there are two classes in the data i.e, the blue square and red triangle. The green dot needs to be sorted out either in the mentioned two classes. K-NN sorts this issue by finding the nearest neighbors and classifying the data into the appropriate class. In this example, the green dot will mostly be classified as red triangle.

Applications/Use-Cases:

K-Nearest Neighbor techniques for Pattern Recognition are often used for theft prevention in the modern retail business. You're accustomed to seeing CCTV cameras around almost every store you visit, you might imagine that there is someone in the back room monitoring these cameras for suspicious activity, and perhaps that is how things were done in the past. But today, a modern surveillance system is intelligent enough to analyze and interpret video data on its own, without a need for human assistance.

K-Nearest Neighbor is also used in Retail to detect patterns in credit card usage. Many new transaction-scrutinizing software applications use kNN algorithms to analyze register data and spot unusual patterns that indicate suspicious activity. For example, if register data indicates that a lot of customer information is being entered manually rather than through automated scanning and swiping, this could indicate that the employee who's using that register is in fact stealing customer's personal information. Or if register data indicates that

a particular good is being returned or exchanged multiple times, this could indicate that employees are misusing the return policy or trying to make money from doing fake returns.

Other Application's of kNN:
Concept Searching
Recommender Systems

The K-Nearest Neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve classification problems. It's easy to implement and understand, but has a major drawback of becoming significantly slow as the size of the data grows.

Fuzzy logic
What is fuzzy logic?
Fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0) Boolean logic on which the modern computer is based.

The idea of fuzzy logic was first advanced by Lotfi Zadeh of the University of California at Berkeley in the 1960s. Zadeh was working on the problem of computer understanding of natural language. Natural language -- like most other activities in life and indeed the universe -- is not easily translated into the absolute terms of 0 and 1. Whether everything is ultimately describable in binary terms is a philosophical question worth pursuing, but in practice, much data we might want to feed a computer is in some state in between and so, frequently, are the results of computing. It may help to see fuzzy logic as the way reasoning really works and binary, or Boolean, logic is simply a special case of it.

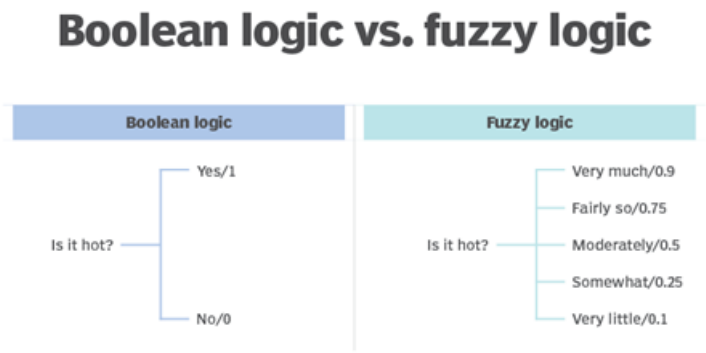


Chart comparing Boolean logic and fuzzy logic
Comparing the degrees of truth with Boolean logic vs. fuzzy logic

Fuzzy logic in AI
In artificial intelligence (AI) systems, fuzzy logic is used to imitate human reasoning and cognition. Rather than strictly binary cases of truth, fuzzy logic includes 0 and 1 as extreme cases of truth but with various intermediate degrees of truth.
As a result, fuzzy logic is well-suited for the following:

- engineering for decisions without clear certainties and uncertainties, or with imprecise data -- such as with natural language processing technologies; and
- regulating and controlling machine outputs, according to multiple inputs/input variables -- such as with temperature control systems.

IBM's Watson supercomputer is one of the most prominent examples of how variations of fuzzy logic and fuzzy semantics are used.

Examples of how fuzzy logic is used
How fuzzy logic is used in technology

Fuzzy logic applications

- Various types of AI systems and technologies use fuzzy logic. This includes vehicle intelligence, consumer electronics, medicine, software, chemicals and aerospace.
- In automobiles, fuzzy logic is used for gear selection and is based on factors such as engine load, road conditions and style of driving.
- In dishwashers, fuzzy logic is used to determine the washing strategy and power needed, which is based on factors such as the number of dishes and the level of food residue on the dishes.
- In copy machines, fuzzy logic is used to adjust drum voltage based on factors such as humidity, picture density and temperature.
- In aerospace, fuzzy logic is used to manage altitude control for satellites and spacecrafts based on environmental factors.
- In medicine, fuzzy logic is used for computer-aided diagnoses, based on factors such as symptoms and medical history.
- In chemical distillation, fuzzy logic is used to control pH and temperature variables.
- In natural language processing, fuzzy logic is used to determine semantic relations between concepts represented by words and other linguistic variables.
- In environmental control systems, such as air conditioners and heaters, fuzzy logic determines output based on factors such as current temperature and target temperature.
- In a business rules engine, fuzzy logic may be used to streamline decision-making according to predetermined criteria.

Why fuzzy classifiers?

A classifier is an algorithm that assigns a class label to an object, based on the object description. It is also said that the classifier predicts the class label. The object description comes in the form of a vector containing values of the features (attributes) deemed to be relevant for the classification task. Typically, the classifier learns to predict class labels using a training algorithm and a training data set. When a training data set is not available,

a classifier can be designed from prior knowledge and expertise. Once trained, the classifier is ready for operation on unseen objects.

Classification belongs to the general area of pattern recognition and machine learning.

- Soft labelling. The standard assumption in pattern recognition is that the classes are mutually exclusive. This may not be the case, as the example in Figure 1 shows. A standard classifier will assign a single crisp label (rain). A fuzzy classifier can assign degrees of membership (soft labels) in all four classes {rain, clouds, wind, sunshine}, accounting for the possibility of winds and cloudy weather throughout the day. A standard classifier can output posterior probabilities, and offer soft labelling too. However, a probability of, say, 0.2 for cloudy weather means that there is 20% chance that tomorrow will be cloudy. A probabilistic model would also assume that the four classes form a full group, i.e., snow, blizzards or thunderstorms must be subsumed by one of the existing four classes. Soft labelling is free from this assumption. A fuzzy classifier, D , producing soft labels can be perceived as a function approximator $D:F \rightarrow [0,1]c$, where F is the feature space where the object descriptions live, and c is the number of classes. While tuning such a function approximator outside the classification scenario would be very difficult, fuzzy classifiers may provide a solution that is both intuitive and useful.
- Interpretability. Automatic classification in most challenging applications such as medical diagnosis has been sidelined due to ethical, political or legal reasons, and mostly due to the black box philosophy underpinning classical pattern recognition. Fuzzy classifiers are often designed to be transparent, i.e., steps and logic statements leading to the class prediction are traceable and comprehensible.
- Limited data, available expertise. Examples include predicting and classification of rare diseases, oil depositions, terrorist activities, natural disasters. Fuzzy classifiers can be built using expert opinion, data or both.

Models of fuzzy classifiers

The broad definition of a fuzzy classifier implies a variety of possible models.

Fuzzy rule-based classifiers

Class label as the consequent

The simplest fuzzy rule-based classifier is a fuzzy if-then system, similar to that used in fuzzy control. Consider a 2D example with 3 classes. A fuzzy classifier can be constructed by specifying classification rules, e.g.,

IF $x1$ is *medium* AND $x2$ is *small* THEN class is 1
IF $x1$ is *medium* AND $x2$ is *large* THEN class is 2
IF $x1$ is *large* AND $x2$ is *small* THEN class is 2
IF $x1$ is *small* AND $x2$ is *large* THEN class is 3

The two features $x1$ and $x2$ are numerical but the rules use *linguistic values*. If there are M possible linguistic values for each feature, and n features in the problem, the number of possible different if-then rules of this conjunction type (AND) is Mn . If the fuzzy classifier comprises of all such rules, then it turns into a simple look-up table. Unlike look-up tables, however, fuzzy classifiers can provide outputs for combinations of linguistic values that are not included as one of the rules. Each linguistic value is represented by a membership function.

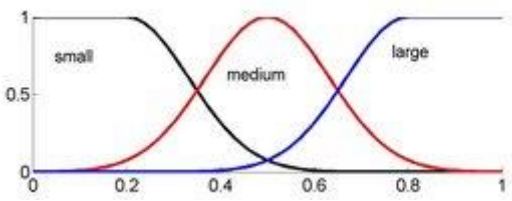


Figure 2: Membership functions for the linguistic terms of x1 .

Figure 2 shows an example of such functions for feature x1 . For any given pair of values x=(x1,x2) , the degree of satisfaction of the antecedent part of the rule determines the firing strength of the rule. For the example above, the firing strength of rule 1 is calculated as

$\tau 1(x)=\mu(1)\text{medium}(x1)\text{AND}\ \mu(2)\text{small}(x2)$.

The superscript (i) is the feature tag. This is needed because the notion of, say, small for x1 may be different from that for x2. The AND operation is typically implemented as minimum but any other t-norm may be used. The rule "votes" for the class of the consequent part. The weight of this vote is $\tau 1(x)$.

To find the output of the classifier, the votes of all rules are aggregated. Among the variety of methods that can be applied for this aggregation, consider the maximum aggregation method. The soft class label for x consists of membership values $gk(x)\in[0,1]$, k=1,...,c , where c is the number of classes. Let $i\rightarrow k$ denote that rule i votes for class k . Then

$$gk(x)=\max i\rightarrow k\tau i(x).(1)$$

The three classes in the example can be thought of as the RGB colours in an image as shown in Figure 3. Class 1 is encoded as red, class 2 as green and class 3 as blue. The dark regions correspond to points in the feature space with very low membership values in all three classes. Labelling of such points may need further analysis. If a crisp label is required, x is assigned to the class with the largest gk(x) . Figure 4 shows the crisp classification regions.

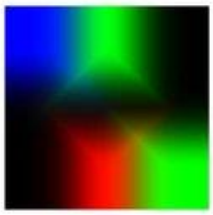


Figure 3: The 3-class output of the fuzzy if-then classifier encoded as RGB.

Figure 4: The classification regions formed from the crisp 3-class output of the fuzzy if-then classifier.

Linguistic labels as the consequent

The consequent part of the rule may also contain linguistic values. For example,

IF x1 is medium AND x2 is small THEN class 1 is large AND class 2 is small AND class 3 is small. This type of rule is easier to obtain from a human expert. The classifier in this case operates as a Mamdani-type fuzzy system (Mamdani, 1977). The output is again a soft label containing the values of c discriminant functions.

Function as the consequent

This type of fuzzy classifier is based on Takagi-Sugeno fuzzy systems (Takagi and Sugeno, 1985). A generic fuzzy if-then rule for classification is a regressor over the feature data space:

IF x1 is A1 and AND x2 is A2 AND ... AND xn is An THEN g1=$\sum$ni=0ai1xi AND ... gc=$\sum$ni=0aicxi ,

where Ai are linguistic values and aij are scalar coefficients (See more details in (Cordon et al., 1999)).

The most useful special case is when the support for each class is a single constant value, usually within the interval [0,1]

IF x1 is A1 and AND x2 is A2 AND ... AND xn is An THEN g1=β1 AND ... gc=βc ,

where βk are constants.

In this model every rule votes for all the classes. The maximum aggregation method will be the same as in equation (1) and the summation will be taken across all the rules. Another popular aggregation method is the weighted sumgk(x)=$\sum$iβk,iτi(x)$\sum$iτi(x) ,where βk,i is the consequent constant for class k in rule i .

Training fuzzy rule-based classifiers

The critical question is how fuzzy classifiers are trained. Where do the membership functions for the linguistic values come from? How are the rules constructed? How are the consequents determined? There is a myriad of methods for training and fine-tuning among which are fuzzy neural networks (Nauck et al., 1997), genetic algorithms (see Roubos et al., 2003; Ishibuchi et al., 1995), as well as numerous heuristic methods for rule extraction based on the data geometry. Online training of fuzzy classifiers has also been considered (Angelov and Zhou, 2008).

Some fuzzy rule-based classifiers suffer from combinatorial explosion of the number of rules (Kuncheva, 2000). The main reason is that the classifier is trained by partitioning of the data space along each feature (Babuska, 1998). A careful pay-off between accuracy and transparency must be made.

Fuzzy prototype-based classifiers

There are fuzzy classifier models inspired by the idea of "fuzzifying" conventional classifiers. A typical representative of this group is the K-nearest neighbour classifier (K-nn). In the classical K-nn, the object x is labelled as the majority of its K nearest neighbours in a reference data set. The approximations of the posterior probabilities for the classes are crude, given by the proportion of neighbours out of k voting for the respective class. Fuzzy K-nn uses the distances to the neighbours as well as their soft labels, if these are available.

The reference set for this classifier does not have to be selected from the existing data. A set of relevant objects (prototypes) with crisp or soft labels can be constructed. The class membership of x is obtained through combining the similarities between x and the prototypes. Fuzzy prototype-based classifiers can be related to popular classifier models including Parzen classifier, learning vector quantization (LVQ) and radial basis functions (RBF) neural networks (Kuncheva 2000).

Along with training from data, human expertise can be used. Experts can assign soft labels to prototypes, construct prototypes not present in the training data, specify meaningful types of combination of similarities, etc. Providing this type of expertise is much more intuitive for the domain expert compared to training fuzzy rule-based classifiers. Experts may not be able

to explicate the intuitive ways in which they reach a decision about class labels using the jargon of if-then rules and membership functions. On the other hand, in the training of fuzzy prototype-based classifiers the expert insight and intuition do not have to be taken to the fore, analysed and mimicked.

Fuzzy combination rules for classifier ensembles

In multiple classifier systems (classifier ensembles) the decisions of several classifiers are aggregated to produce a single (crisp or soft) class label. Given an object x , let $d_{i,j}(x) \in [0,1]$ be the degree of membership that classifier i suggests for class j , i=1,...,L (number of classifiers), j=1,...,c (number of classes). Many conventional classifiers can produce soft labels, usually as estimates the posterior probabilities for the classes, conditioned on x . The matrix $\{d_{i,j}(x)\}$ is called the decision profile for x .

Fuzzy aggregation functions (aggregation rules) abound in fuzzy decision making. The overall support for class j is calculated using the decision profile entries for that class, which is the j-th column of the matrix. Given an aggregation function A , the soft ensemble output for class j is $d_{,j}(x)=A(d_{1,j}(x),d_{2,j}(x),...,d_{L,j}(x))$.Minimum, maximum and mean (order statistics in fuzzy disguise) are the most popular choices for A . Any function $A:[0,1]^L \rightarrow [0,1]$ can be used instead, e.g., product or ordered weighted averaging (OWA) (Yager and Kacprzyk, 1997).

The aggregation can also be made by fuzzy integral, which combines the class support (evidence) with the competence of the classifiers in the ensemble (Kuncheva, 2003).

TEXT / REFERENCE BOOKS

1. **Rafael C. Gonzalez, Richard E. Woods," Digital Image Processing", Pearson , Second Edition, 2004**
2. **Anil K. Jain, "Fundamentals of Digital Image Processin", Pearson 2002**
3. **Robert J.Schalkoff, "Pattern Recognition Statistical, Structural and Neural Approaches",John Wiley & Sons Inc., New York, 1992**
4. **R.O.Duda, P.E.Hart and D.G.Stork, "Pattern Classification", John Wiley, 2001**
5. **Himanshu Singh. "Practical Machine Learning and Image Processing", Apress, 2019**
6. **François Chollet "Deep Learning with Python", Manning Publications Co., NY, 2018**
7. **Phil Kim. "MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence", Apress, 2017**
8. **Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images**
9. **Valentina Zharkova, Lakhmi C. Jain, "Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images", Springer, 2007**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

UNIT – V – Digital Image Processing for Real Time Applications – SEC1628

V. AI IN IMAGE PROCESSING

Machines can be taught to interpret images the same way our brains do and to analyze those images much more thoroughly than we can. When applied to image processing, artificial intelligence (AI) can power face recognition and authentication functionality for ensuring security in public places, detecting and recognizing objects and patterns in images and videos, and so on.

In this unit, we talk about digital image processing and the role of AI in it. We describe some AI-based image processing tools and techniques you may use for developing intelligent applications. We also take a look at the most popular neural network models used for different image processing tasks. This article will be useful for anyone aiming to build an AI solution for image processing.

How does it work?

Now, a few words about how image recognition works. Image recognition algorithms make image recognition possible. The first step here is gathering and organizing the data. Data organization means classifying each image and distinguishing its physical characteristics. Unlike humans, computers perceive a picture as a vector or raster image. So after the constructs depicting objects and features of the image are created, the computer analyzes them. Therefore, the correct collection and organization of data are essential for training the image recognition model, because if the quality of the data is discredited at this stage, it will not be able to recognize patterns at a later stage.

The second step of the image recognition process is building a predictive model. The classification algorithm has to be trained carefully; otherwise, it won't be able to deliver its function. Image recognition algorithms use deep learning datasets to distinguish patterns in images. These datasets consist of hundreds of thousands of tagged images. The algorithm looks through these datasets and learns how the image of a particular object looks like. When everything is done and tested, you can enjoy the image recognition feature.

Image recognition vs. Image Processing

You should remember that image recognition and image processing are not synonyms. Image processing is a kind of computer vision. Image processing means converting an image into a digital form and performing certain operations on it. As a result, it is possible to extract some information from such an image.

Image processing stages

- Color image processing — the colors are processed
- Image enhancement — the quality of the image is improved and the hidden details are extracted
- Image restoration — the image is cleaned from blurs and other unpleasant things
- Representation and description — the processes data is visualized
- Image acquisition — the image is captured and converted
- Image compression and decompression — the size and resolution of the image are changed, if necessary
- Morphological processing — the structure of the image's objects is described
- Image recognition — specific features of the image's objects are identified

Image recognition: Visual Search

Visual search uses real images (screenshots, web images, or photos) as an incentive to search the web. Current visual search technologies use artificial intelligence (AI) to understand the content and context of these images and return a list of related results. It is applied in more and more industries. One of them is e-commerce.

For instance, Boohoo, an online retailer, developed an app with the visual search feature. A user simply snaps an item they like, uploads the picture, and the technology does the rest. Thanks to image recognition, a user sees if Boohoo offers something similar and doesn't waste loads of time searching for a specific item. Other applications of image recognition (already existing and potential) include creating city guides, powering self-driving cars, making augmented reality apps possible, teaching manufacturing machines to see defects, and so on. There is even an app that helps users to understand if an object of the image is a hotdog or not.

Visual Search Statistics

- 90% of information transmitted to the human brain is visual.
- 62% of millennials want visual search over any other new technology.
- 45% of retailers in the UK now use visual search.
- The global visual search market is estimated to exceed USD 14,727 million by 2023, an increase of + 9% over the forecast period 2018-2023.

Visual Search Trends

- Brands (in particular Mastercard) completely remove text from their images in favor of a designer expression of their identity.
- Google search offers more and more visual possibilities with more images and an improved user interface. Google images increasingly follow Pinterest.
- Pinterest combines visual search with text search, which should increase its reach.
- Retailers are developing their visual search capabilities, rather than relying on search engines and social networks as intermediaries.

Impact of AI on Image Recognition

We described how image recognition works, but you may still have a lot of questions regarding how to complete those stages. Here is an answer — do image recognition using AI. Artificial intelligence makes all the features of image recognition possible. To give you a better understanding, here are some of them:

Facial recognition

With the help of AI, a facial recognition system maps facial features from an image and then compares this information with a database to find a match. Facial recognition is used by mobile phone makers (as a way to unlock a smartphone), social networks (recognizing people on the picture you upload and tagging them), and so on. However, such systems raise a lot of privacy concerns, as sometimes the data can be collected without a user's permission. Apart from this, even the most advanced systems can't guarantee 100% accuracy. What if a facial recognition system confuses a random user with a criminal? That's not the thing someone wants to happen, but this is still possible. However, technology is constantly evolving, so one day this problem may disappear.

Facial Recognition Trends

- Application of facial recognition models at airport
- Face recognition is now being used at airports to check security and increase alertness. Due to increasing demand for high-resolution 3D facial recognition, thermal facial recognition technologies and image recognition models, this strategy is being applied at major airports around the world.
- Know who your customers are
- In the finance and investment area, one of the most fundamental verification processes is to know who your customers are. As a result of the pandemic, banks were unable to carry out this operation on a large scale in their offices. As a result, face recognition models are growing in popularity as a practical method for recognizing clients in this industry.
- Robotic face recognition system

- In 2020, India implemented an automatic facial recognition system, which will further improve the identity verification process for the National Crime Registration Bureau. It is expected that these systems will be very popular this year.

Object recognition

Object recognition systems pick out and identify objects from the uploaded images (or videos). It is possible to use two methods of deep learning to recognize objects. One is to train the model from scratch, and the other is to use an already trained deep learning model. Based on these models, many helpful applications for object recognition are created. Visual search is probably the most popular application of this technology.

For example, the applications Google Lens identifies the object in the image and gives the user information about this object and search results. As we said before, this technology is especially valuable in e-commerce stores and brands.

Text detection

Everything is obvious here — text detection is about detecting text and extracting it from an image.

Pattern recognition

Pattern recognition means finding and extracting specific patterns in a given image. Those can be textures, facial expressions, etc.

Image analysis

Do you need a summary of a specific image? Then use AI for picture and image analysis. As a result, all the objects of the image (shapes, colors, and so on) will be analyzed, and you will get insightful information about the picture.

Use Cases of Image Recognition

Mobile e-commerce

An excellent example of image recognition is the CamFind API from image Searcher Inc. This technology provides an advanced level of mobile trading. CamFind recognizes items such as watches, shoes, bags, sunglasses, etc., and returns the user's purchase options. Potential buyers can compare products in real-time without visiting websites. Developers can use this image recognition API to create their mobile commerce applications.

Gaming Industry

Recognition models and computer vision technologies have also had a great impact on the gaming industry. It is known that the Microsoft Kinect video game is listed in the Guinness Book of Records as the fastest-selling consumer electronics device. The game is based on computer vision and tracks the human body in real-time.

Healthcare

Detecting brain tumors or strokes and helping people with poor eyesight are some examples of the use of image recognition in the healthcare sector. The study shows that the image recognition algorithm detects lung cancer with an accuracy of 97%.

With the increase in the ability to recognize computer vision, surgeons can use augmented reality in real operations. It can issue warnings, recommendations, and updates depending on what the algorithm sees in the operating system.

Banking

Banks are increasingly using facial recognition to confirm the identity of the customer, who uses internet banking. Banks also use facial recognition " limited access control " to control the entry and access of certain people to certain areas of the facility.

Manufacturing

For pharmaceutical companies, it is important to count the number of tablets or capsules before placing them in containers. To solve this problem, Pharma packaging systems, based in England, has developed a solution that can be used on existing production lines and even

operate as a stand-alone unit. A principal feature of this solution is the use of computer vision to check for broken or partly formed tablets.

Image recognition and models play a huge role in the automotive sector. The main reason for this was to extend site management services, security, and performance. It is also about implementing AI based on deep learning to track people and predict the movement of equipment to avoid dangerous interactions, thus increasing so increasing safety.

Methods and Techniques for Image Processing with AI

Image processing is a method of converting an image into digital form and performing certain operations on it to obtain an improved image or extract useful information from it. This is a type of signal distribution in which the input is an image, such as a video frame or photo, and the output may have an image or features associated with that image.

Let's start with the simplest things — methods of image processing. Currently, there are only two of them: analog and digital. The analog method is used for processing hard copies of images (like printouts). Above all, the mission of the digital one is to manipulate digital images using computer algorithms.

Regarding the techniques, they exist in spades, and we have already mentioned some of them. For instance, image restoration is considered both as a stage and technique of image processing. Here are some of the other techniques:

- Pixelation — turning printed pictures into the digitized ones
- Linear filtering — processing input signals and producing the output ones which are subject to the constraint of linearity
- Edge detection — finding meaningful edges of the image's objects
- Anisotropic diffusion — reducing the image noise without removing crucial parts of the picture
- Principal components analysis — extracting the features of the image

Tools for Image Recognition

Fortunately, you don't have to develop everything from scratch — you can use already existing platforms and frameworks. Cloud Vision API from Google is one of the most popular of them. Features of this platform include image labeling, text detection, Google search, explicit content detection, and others. If you choose this option, you will be charged per image. However, the first 1000 images used each month are free.

The next variant is Amazon Rekognition (yes, we made no mistake, that's Rekognition). It allows adding visual analysis features to your app, integrating face-based user verification, identifying diverse objects, detecting unsafe content, etc. You will be charged for using the platform, but you can still try it for free.

Finally, you can develop a new custom computer vision model and train it. Upload your own labeled images, tag them and improve your classifier — everything is very simple. Just like with Google and Amazon, where you pay only for what you use. A free trial is available as well.

Future of Image Recognition

Due to further research and technological improvements, computer vision will have a wider range of functions in the future.

Computer vision technologies will not only make learning easier but will also be able to distinguish more images than at present. In the future, it can be used in connection with other technologies to create more powerful applications.

Computer vision will play an important role in the development of general artificial intelligence (AGI) and artificial super intelligence (ASI), giving them the ability to process information as well or even better than the human visual system. In addition, by studying the vast number of available visual media, image recognition models will be able to predict the future.

Ocean send erosion

NC State's Coastal & Computational Hydraulics team developed a computer vision model called XBeach to analyze flooding and erosion during hurricanes. Computer vision helps researchers predict and reduce the future impact of severe storms on local communities and sand dunes.

Deforestation

Using computer vision trained in satellite imagery and visual data from Earth, experts can remotely control those ecosystems that are threatened by deforestation. The technology helps identify and analyze emergencies and stop illegal actions before they cause unchangeable damage.

Food production

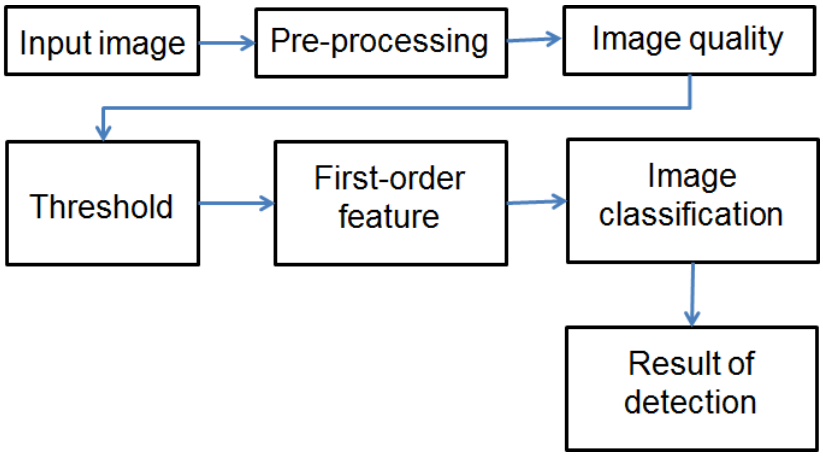
Although modern agriculture still focuses mainly on growing a single crop on a large plot, computer vision software can help farmers manage a greater variety of crops more effectively by informing them what and when to plant. Machine learning can also learn from annotated images to more correctly predict yields and analyze the condition of plants and livestock.

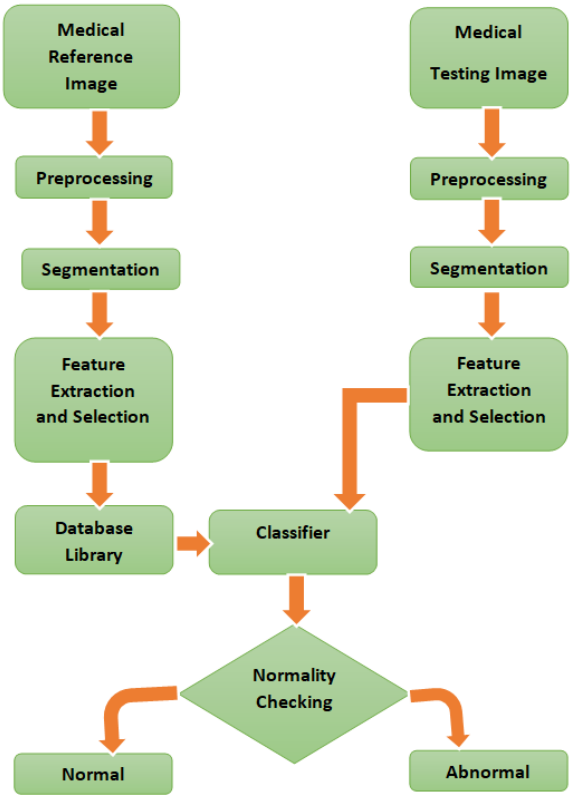
Today, computer vision has benefited enormously from deep learning technologies, excellent development tools, and image recognition models, comprehensive open source databases, and fast and inexpensive computing. Image recognition has found wide application in various industries and enterprises, from self-driving cars and electronic commerce to industrial automation and medical imaging analysis.

Design and execute image classification

Generally, image classification is comprised of learning (training), and classification steps. During the first step (that is, learning or training phase), a classification algorithm builds a model by learning from a labeled land use/cover training data set. In the second step, the model is used for prediction. More importantly, classification accuracy is also performed using an independent test set.

In k-Nearest Neighbors (KNN), Artificial Neural Networks (ANN), single Decision Trees (DT), Support Vector Machines (SVM) and Random Forest (RF) machine learning classifiers will be used for image classification. These are some of the most common classifiers, which are used for image processing and classification. Rather, the aim is to provide a brief review with a focus on practical supervised machine learning for remote sensing image classification. Below is a brief summary of the selected machine learning classifiers.





The aim of **pre-processing** is an improvement of the image data that suppresses unwanted distortions or enhances some image features important for further processing.

Image segmentation

In digital image processing and computer vision, image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.

Feature extraction

In machine learning, pattern recognition, and image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction.

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

What is Feature Extraction?

Feature extraction is a part of the dimensionality reduction process, in which, an initial set of the raw data is divided and reduced to more manageable groups. So when you want to process it will be easier. The most important characteristic of these large data sets is that they

have a large number of variables. These variables require a lot of computing resources to process them. So Feature extraction helps to get the best feature from those big data sets by select and combine variables into features, thus, effectively reducing the amount of data. These features are easy to process, but still able to describe the actual data set with the accuracy and originality.

The recognition and classification of images is what enables many of the most impressive accomplishments of artificial intelligence. Yet how do computers learn to detect and classify images? In this article, we'll cover the general methods that computers use to interpret and detect images and then take a look at some of the most popular methods of classifying those images.

Classification

Image classification is the process of categorizing and labeling groups of pixels or vectors within an image based on specific rules. The categorization law can be devised using one or more spectral or textural characteristics. Two general methods of classification are 'supervised' and 'unsupervised'.

Pixel-Level vs. Object-Based Classification

Image classification techniques can mainly be divided into two different categories: pixel-based classification and object-based classification.

Pixels are the base units of an image, and the analysis of pixels is the primary way that image classification is done. However, classification algorithms can either use just the spectral information within individual pixels to classify an image or examine spatial information (nearby pixels) along with the spectral information. Pixel-based classification methods utilize only spectral information (the intensity of a pixel), while object-based classification methods take into account both pixel spectral information and spatial information.

There are different classification techniques used for pixel-based classification. These include minimum-distance-to-mean, maximum-likelihood, and minimum-Mahalanobis-distance. These methods require that the means and variances of the classes are known, and they all operate by examining the "distance" between class means and the target pixels.

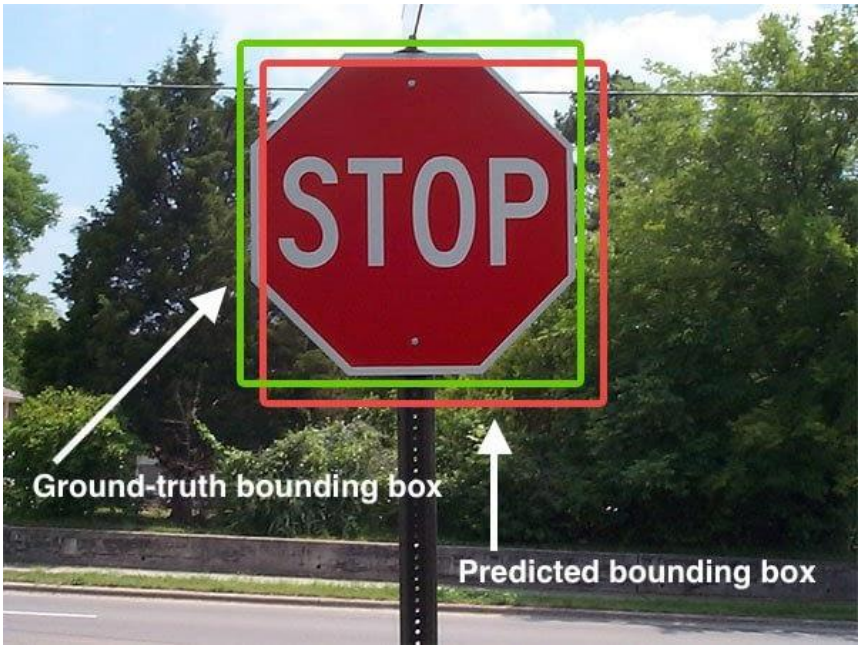
Pixel-based classification methods are limited by the fact that they can't use information from other nearby pixels. In contrast, object-based classification methods can include other pixels and therefore they also use spatial information to classify items. Note that "object" just refers to contiguous regions of pixels and not whether or not there is a target object within that region of pixels.

Preprocessing Image Data For Object Detection

The most recent and reliable image classification systems primarily use object-level classification schemes, and for these approaches image data must be prepared in specific ways. The objects/regions need to be selected and preprocessed.

Before an image, and the objects/regions within that image, can be classified the data that comprises that image has to be interpreted by the computer. Images need to be preprocessed and readied for input into the classification algorithm, and this is done through object detection. This is a critical part of readying the data and preparing the images to train the machine learning classifier.

Object detection is done with a variety of methods and techniques. To begin with, whether or not there are multiple objects of interest or a single object of interest impacts how the image preprocessing is handled. If there is just one object of interest, the image undergoes image localization. The pixels that comprise the image have numerical values that are interpreted by the computer and used to display the proper colors and hues. An object known as a bounding box is drawn around the object of interest, which helps the computer know what part of the image is important and what pixel values define the object. If there are multiple objects of interest in the image, a technique called object detection is used to apply these bounding boxes to all the objects within the image.



Another method of preprocessing is image segmentation. Image segmentation functions by dividing the whole image into segments based on similar features. Different regions of the image will have similar pixel values in comparison to other regions of the image, so these pixels are grouped together into image masks that correspond to the shape and boundaries of the relevant objects within the image. Image segmentation helps the computer isolate the features of the image that will help it classify an object, much like bounding boxes do, but they provide much more accurate, pixel-level labels.

After the object detection or image segmentation has been completed, labels are applied to the regions in question. These labels are fed, along with the values of the pixels comprising the object, into the machine learning algorithms that will learn patterns associated with the different labels.

Machine Learning Algorithms

Once the data has been prepared and labeled, the data is fed into a machine learning algorithm, which trains on the data. We'll cover some of the most common kinds of machine learning image classification algorithms below.

K-Nearest Neighbors

K-Nearest Neighbors is a classification algorithm that examines the closest training examples and looks at their labels to ascertain the most probable label for a given test example. When it comes to image classification using KNN, the feature vectors and labels of the training

images are stored and just the feature vector is passed into the algorithm during testing. The training and testing feature vectors are then compared against each other for similarity.

KNN-based classification algorithms are extremely simple and they deal with multiple classes quite easily. However, KNN calculates similarity based on all features equally. This means that it can be prone to misclassification when provided with images where only a subset of the features is important for the classification of the image.

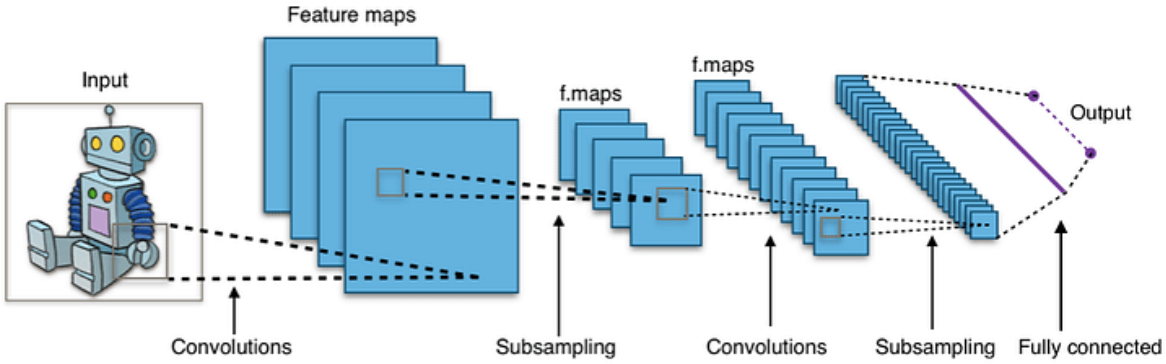
Support Vector Machines

Support Vector Machines are a classification method that places points in space and then draws dividing lines between the points, placing objects in different classes depending on which side of the dividing plane the points fall on. Support Vector Machines are capable of doing nonlinear classification through the use of a technique known as the kernel trick. While SVM classifiers are often very accurate, a substantial drawback to SVM classifiers is that they tend to be limited by both size and speed, with speed suffering as size increases.

Multi-Layer Perceptrons (Neural Nets)

Multi-layer perceptrons, also called neural network models, are machine learning algorithms inspired by the human brain. Multilayer perceptrons are composed of various layers that are joined together with each other, much like neurons in the human brain are linked together. Neural networks make assumptions about how the input features are related to the data's classes and these assumptions are adjusted over the course of training. Simple neural network models like the multi-layer perceptron are capable of learning non-linear relationships, and as a result, they can be much more accurate than other models. However, MLP models suffer from some notable issues like the presence of non-convex loss functions.

Deep Learning Algorithms (CNNs)



The most commonly used image classification algorithm in recent times is the Convolutional Neural Network (CNNs). CNNs are customized versions of neural networks that combine the multilayer neural networks with specialized layers that are capable of extracting the features most important and relevant to the classification of an object. CNNs can automatically discover, generate, and learn features of images. This greatly reduces the need to manually label and segment images to prepare them for machine learning algorithms. They also have an advantage over MLP networks because they can deal with non-convex loss functions.

Convolutional Neural Networks get their name from the fact that they create "convolutions". CNNs operate by taking a filter and sliding it over an image. You can think of this as viewing sections of a landscape through a moveable window, concentrating on just the features that

are viewable through the window at any one time. The filter contains numerical values which are multiplied with the values of the pixels themselves. The result is a new frame, or matrix, full of numbers that represent the original image. This process is repeated for a chosen number of filters, and then the frames are joined together into a new image that is slightly smaller and less complex than the original image. A technique called pooling is used to select just the most important values within the image, and the goal is for the convolutional layers to eventually extract just the most salient parts of the image that will help the neural network recognize the objects in the image.

Convolutional Neural Networks are comprised of two different parts. The convolutional layers are what extract the features of the image and convert them into a format that the neural network layers can interpret and learn from. The early convolutional layers are responsible for extracting the most basic elements of the image, like simple lines and boundaries. The middle convolutional layers begin to capture more complex shapes, like simple curves and corners. The later, deeper convolutional layers extract the high-level features of the image, which are what is passed into the neural network portion of the CNN, and are what the classifier learns.

Object detection and object localization systems using machine learning

It can be challenging for beginners to distinguish between different related computer vision tasks.

For example, image classification is straight forward, but the differences between object localization and object detection can be confusing, especially when all three tasks may be just as equally referred to as object recognition.

Image classification involves assigning a class label to an image, whereas object localization involves drawing a bounding box around one or more objects in an image. Object detection is more challenging and combines these two tasks and draws a bounding box around each object of interest in the image and assigns them a class label. Together, all of these problems are referred to as object recognition and will discover a gentle introduction to the problem of object recognition and state-of-the-art deep learning models designed to address it.

Object recognition is refers to a collection of related tasks for identifying objects in digital images.

Region-Based Convolutional Neural Networks, or R-CNNs, are a family of techniques for addressing object localization and recognition tasks, designed for model performance.

You Only Look Once, or YOLO, is a second family of techniques for object recognition designed for speed and real-time use.

What is Object Recognition?

Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects in digital photographs.

Image classification involves predicting the class of one object in an image. Object localization refers to identifying the location of one or more objects in an image and drawing abounding box around their extent. Object detection combines these two tasks and localizes and classifies one or more objects in an image.

When a user or practitioner refers to “object recognition“, they often mean “object detection“. As such, we can distinguish between these three computer vision tasks:

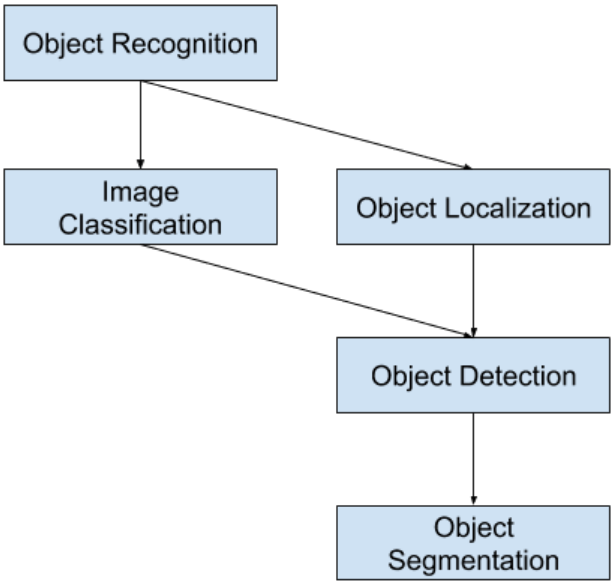
- Image Classification: Predict the type or class of an object in an image.
  Input: An image with a single object, such as a photograph.
  Output: A class label (e.g. one or more integers that are mapped to class labels).
- Object Localization: Locate the presence of objects in an image and indicate their location with a bounding box.

Input: An image with one or more objects, such as a photograph.
Output: One or more bounding boxes (e.g. defined by a point, width, and height).

- Object Detection: Locate the presence of objects with a bounding box and types or classes of the located objects in an image.
  Input: An image with one or more objects, such as a photograph.
  Output: One or more bounding boxes (e.g. defined by a point, width, and height), and a class label for each bounding box.

One further extension to this breakdown of computer vision tasks is object segmentation, also called "object instance segmentation" or "semantic segmentation," where instances of recognized objects are indicated by highlighting the specific pixels of the object instead of a coarse bounding box.

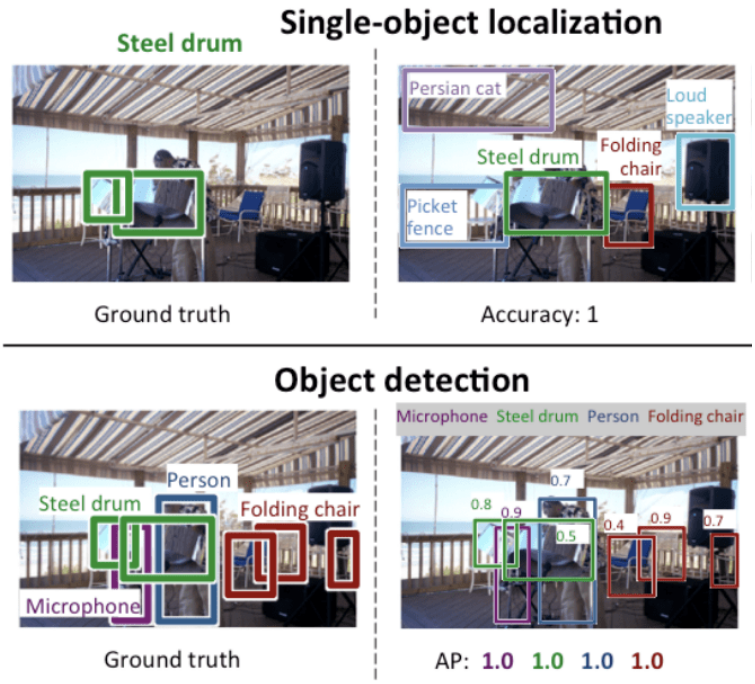
From this breakdown, we can see that object recognition refers to a suite of challenging computer vision tasks.



Overview of Object Recognition Computer Vision Tasks

- **Image classification**: Algorithms produce a list of object categories present in the image.
- **Single-object localization**: Algorithms produce a list of object categories present in the image, along with an axis-aligned bounding box indicating the position and scale of one instance of each object category.
- **Object detection**: Algorithms produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each object category.

We can see that "Single-object localization" is a simpler version of the more broadly defined "Object Localization," constraining the localization tasks to objects of one type within an image, which we may assume is an easier task.



Comparison Between Single Object Localization and Object Detection.Taken From: ImageNet Large Scale Visual Recognition Challenge.

The performance of a model for image classification is evaluated using the mean classification error across the predicted class labels. The performance of a model for single-object localization is evaluated using the distance between the expected and predicted bounding box for the expected class. Whereas the performance of a model for object recognition is evaluated using the precision and recall across each of the best matching bounding boxes for the known objects in the image.

Now that we are familiar with the problem of object localization and detection, let's take a look at some recent top-performing deep learning models.

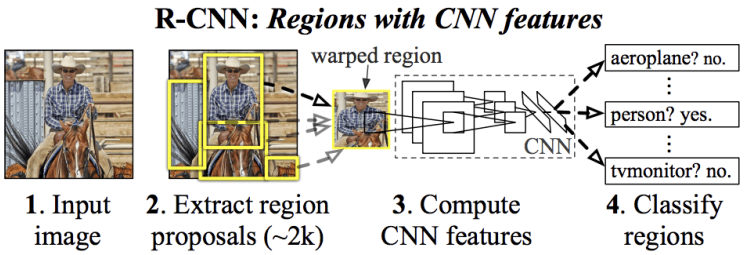
R-CNN Model Family

R-CNN

Their proposed R-CNN model is comprised of three modules; they are:

- Module 1: Region Proposal. Generate and extract category independent region proposals, e.g. candidate bounding boxes.
- Module 2: Feature Extractor. Extract feature from each candidate region, e.g. using a deep convolutional neural network.
- Module 3: Classifier. Classify features as one of the known class, e.g. linear SVM classifier model.

The architecture of the model is summarized in the image below,



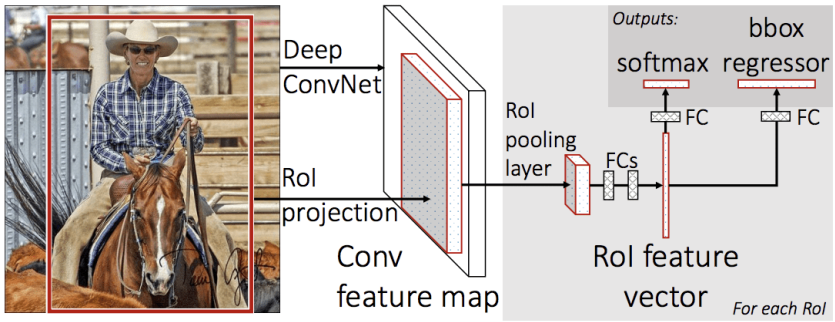
Summary of the R-CNN Model Architecture Taken from Rich feature hierarchies for accurate object detection and semantic segmentation.

A computer vision technique is used to propose candidate regions or bounding boxes of potential objects in the image called "*selective search*," although the flexibility of the design allows other region proposal algorithms to be used.
It is a relatively simple and straightforward application of CNNs to the problem of object localization and recognition. A downside of the approach is that it is slow, requiring a CNN-based feature extraction pass on each of the candidate regions generated by the region proposal algorithm. This is a problem as the paper describes the model operating upon approximately 2,000 proposed regions per image at test-time.

Fast R-CNN

- **Training is a multi-stage pipeline**. Involves the preparation and operation of three separate models.
- **Training is expensive in space and time**. Training a deep CNN on so many region proposals per image is very slow.
- **Object detection is slow**. Make predictions using a deep CNN on so many region proposals is very slow.

Fast R-CNN is proposed as a single model instead of a pipeline to learn and output regions and classifications directly. The architecture of the model takes the photograph a set of region proposals as input that are passed through a deep convolutional neural network. A pre-trained CNN, such as a VGG-16, is used for feature extraction. The end of the deep CNN is a custom layer called a Region of Interest Pooling Layer, or RoI Pooling, that extracts features specific for a given input candidate region.
The output of the CNN is then interpreted by a fully connected layer then the model bifurcates into two outputs, one for the class prediction via a softmax layer, and another with a linear output for the bounding box. This process is then repeated multiple times for each region of interest in a given image.
The architecture of the model is summarized in the image below, taken from the paper.



The model is significantly faster to train and to make predictions, yet still requires a set of candidate regions to be proposed along with each input image.

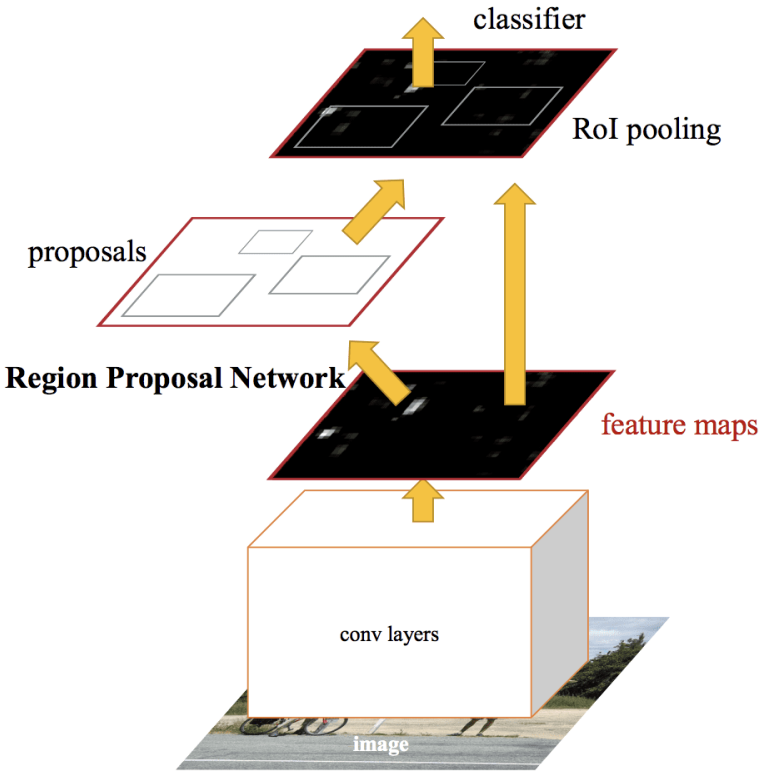
Faster R-CNN

The architecture was designed to both propose and refine region proposals as part of the training process, referred to as a Region Proposal Network, or RPN. These regions are then used in concert with a Fast R-CNN model in a single model design. These improvements both reduce the number of region proposals and accelerate the test-time operation of the model to near real-time with then state-of-the-art performance.

Although it is a single unified model, the architecture is comprised of two modules:

- **Module 1: Region Proposal Network**. Convolutional neural network for proposing regions and the type of object to consider in the region.
- **Module 2: Fast R-CNN**. Convolutional neural network for extracting features from the proposed regions and outputting the bounding box and class labels.
  Both modules operate on the same output of a deep CNN. The region proposal network acts as an attention mechanism for the Fast R-CNN network, informing the second network of where to look or pay attention.

The architecture of the model is summarized in the image below, taken from the paper.



The RPN works by taking the output of a pre-trained deep CNN, such as VGG-16, and passing a small network over the feature map and outputting multiple region proposals and a class prediction for each. Region proposals are bounding boxes, based on so-called anchor boxes or pre-defined shapes designed to accelerate and improve the proposal of regions. The class prediction is binary, indicating the presence of an object, or not, so-called “objectness” of the proposed region.

A procedure of alternating training is used where both sub-networks are trained at the same time, although interleaved. This allows the parameters in the feature detector deep CNN to be tailored or fine-tuned for both tasks at the same time. At the time of writing, this Faster R-CNN architecture is the pinnacle of the family of models and continues to achieve near state-of-the-art results on object recognition tasks.

What is transfer learning? Exploring the popular deep learning approach

Transfer learning is the reuse of a pre-trained model on a new problem. It's currently very popular in deep learning because it can train deep neural networks with comparatively little data. This is very useful in the data science field since most real-world problems typically do not have millions of labeled data points to train such complex models. We'll take a look at what transfer learning is, how it works, why and when you it should be used. Additionally, we'll cover the different approaches of transfer learning and provide you with some resources on already pre-trained models.

Transfer learning, used in machine learning, is the reuse of a pre-trained model on a new problem. In transfer learning, a machine exploits the knowledge gained from a previous task to improve generalization about another. For example, in training a classifier to predict whether an image contains food, you could use the knowledge it gained during training to recognize drinks.

In transfer learning, the knowledge of an already trained machine learning model is applied to a different but related problem. For example, if you trained a simple classifier to predict whether an image contains a backpack, you could use the knowledge that the model gained during its training to recognize other objects like sunglasses.

With transfer learning, we basically try to exploit what has been learned in one task to improve generalization in another. We transfer the weights that a network has learned at "task A" to a new "task B."

The general idea is to use the knowledge a model has learned from a task with a lot of available labeled training data in a new task that doesn't have much data. Instead of starting the learning process from scratch, we start with patterns learned from solving a related task.

Transfer learning is mostly used in computer vision and natural language processing tasks like sentiment analysis due to the huge amount of computational power required.

Transfer learning isn't really a machine learning technique, but can be seen as a "design methodology" within the field, for example, active learning. It is also not an exclusive part or study-area of machine learning. Nevertheless, it has become quite popular in combination with neural networks that require huge amounts of data and computational power.

Why Use Transfer Learning

Transfer learning has several benefits, but the main advantages are saving training time, better performance of neural networks (in most cases), and not needing a lot of data.

Usually, a lot of data is needed to train a neural network from scratch but access to that data isn't always available — this is where transfer learning comes in handy. With transfer learning a solid machine learning model can be built with comparatively little training data because the model is already pre-trained. This is especially valuable in natural language processing because mostly expert knowledge is required to create large labeled datasets.

Additionally, training time is reduced because it can sometimes take days or even weeks to train a deep neural network from scratch on a complex task.

Approaches to Transfer Learning

1. TRAINING A MODEL TO REUSE IT

Imagine you want to solve task A but don't have enough data to train a deep neural network. One way around this is to find a related task B with an abundance of data. Train the deep neural network on task B and use the model as a starting point for solving task A. Whether you'll need to use the whole model or only a few layers depends heavily on the problem you're trying to solve.

If you have the same input in both tasks, possibly reusing the model and making predictions for your new input is an option. Alternatively, changing and retraining different task-specific layers and the output layer is a method to explore.

2. USING A PRE-TRAINED MODEL

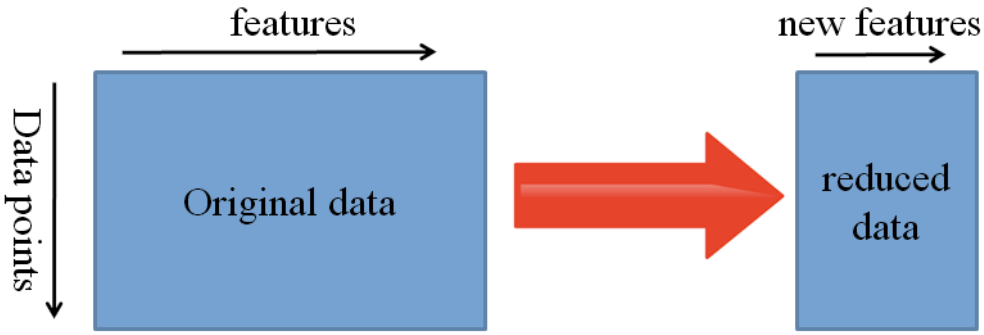
The second approach is to use an already pre-trained model. There are a lot of these models out there, so make sure to do a little research. How many layers to reuse and how many to retrain depends on the problem.

Keras, for example, provides nine pre-trained models that can be used for transfer learning, prediction, feature extraction and fine-tuning. You can find these models, and also some brief tutorials on how to use them, here. There are also many research institutions that release trained models.

This type of transfer learning is most commonly used throughout deep learning.

3. FEATURE EXTRACTION

Another approach is to use deep learning to discover the best representation of your problem, which means finding the most important features. This approach is also known as representation learning, and can often result in a much better performance than can be obtained with hand-designed representation.



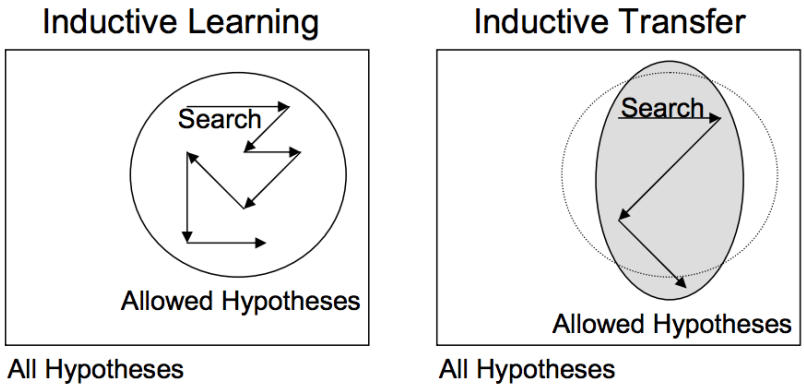
In machine learning, features are usually manually hand-crafted by researchers and domain experts. Fortunately, deep learning can extract features automatically. Of course, this doesn't mean feature engineering and domain knowledge isn't important anymore — you still have to decide which features you put into your network. That said, neural networks have the ability to learn which features are really important and which ones aren't. A representation learning algorithm can discover a good combination of features within a very short timeframe, even for complex tasks which would otherwise require a lot of human effort.

The learned representation can then be used for other problems as well. Simply use the first layers to spot the right representation of features, but don't use the output of the network because it is too task-specific. Instead, feed data into your network and use one of the intermediate layers as the output layer. This layer can then be interpreted as a representation of the raw data.

This approach is mostly used in computer vision because it can reduce the size of your dataset, which decreases computation time and makes it more suitable for traditional algorithms, as well.

How transferable are features in deep neural networks?

This form of transfer learning used in deep learning is called inductive transfer. This is where the scope of possible models (model bias) is narrowed in a beneficial way by using a model fit on a different but related task.



How to Use Transfer Learning?

You can use transfer learning on your own predictive modeling problems.

Two common approaches are as follows:

1. Develop Model Approach
2. Pre-trained Model Approach

Develop Model Approach

1. **Select Source Task**. You must select a related predictive modeling problem with an abundance of data where there is some relationship in the input data, output data, and/or concepts learned during the mapping from input to output data.
2. **Develop Source Model**. Next, you must develop a skillful model for this first task. The model must be better than a naive model to ensure that some feature learning has been performed.
3. **Reuse Model**. The model fit on the source task can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model, depending on the modeling technique used.
4. **Tune Model**. Optionally, the model may need to be adapted or refined on the input-output pair data available for the task of interest.

Pre-trained Model Approach

1. **Select Source Model**. A pre-trained source model is chosen from available models. Many research institutions release models on large and challenging datasets that may be included in the pool of candidate models from which to choose from.
2. **Reuse Model**. The model pre-trained model can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model, depending on the modeling technique used.
3. **Tune Model**. Optionally, the model may need to be adapted or refined on the input-output pair data available for the task of interest.

This second type of transfer learning is common in the field of deep learning.

Examples of Transfer Learning with Deep Learning

Let's make this concrete with two common examples of transfer learning with deep learning models.

Transfer Learning with Image Data

It is common to perform transfer learning with predictive modeling problems that use image data as input.

This may be a prediction task that takes photographs or video data as input.

For these types of problems, it is common to use a deep learning model pre-trained for a large and challenging image classification task such as the ImageNet 1000-class photograph classification competition.

The research organizations that develop models for this competition and do well often release their final model under a permissive license for reuse. These models can take days or weeks to train on modern hardware.

These models can be downloaded and incorporated directly into new models that expect image data as input.

Three examples of models of this type include:

- Oxford VGG Model
- Google Inception Model
- Microsoft ResNet Model

This approach is effective because the images were trained on a large corpus of photographs and require the model to make predictions on a relatively large number of classes, in turn, requiring that the model efficiently learn to extract features from photographs in order to perform well on the problem.

In their Stanford course on Convolutional Neural Networks for Visual Recognition, the authors caution to carefully choose how much of the pre-trained model to use in your new model.

[Convolutional Neural Networks] features are more generic in early layers and more original-dataset-specific in later layers

— Transfer Learning, CS231n Convolutional Neural Networks for Visual Recognition

Transfer Learning with Language Data

It is common to perform transfer learning with natural language processing problems that use text as input or output.

For these types of problems, a word embedding is used that is a mapping of words to a high-dimensional continuous vector space where different words with a similar meaning have a similar vector representation.

Efficient algorithms exist to learn these distributed word representations and it is common for research organizations to release pre-trained models trained on very large corpa of text documents under a permissive license.

Two examples of models of this type include:

- Google's word2vec Model
- Stanford's GloVe Model

These distributed word representation models can be downloaded and incorporated into deep learning language models in either the interpretation of words as input or the generation of words as output from the model.

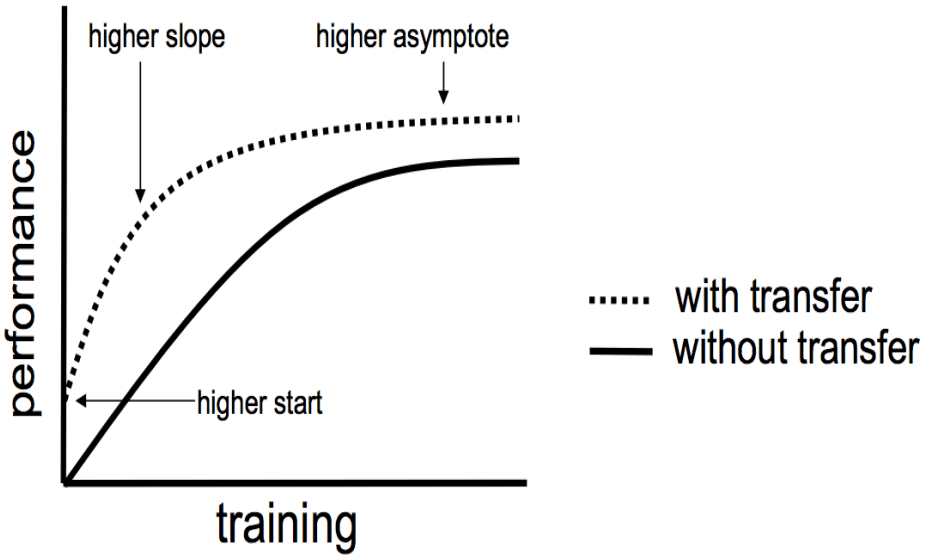
When to Use Transfer Learning?

Transfer learning is an optimization, a shortcut to saving time or getting better performance.

In general, it is not obvious that there will be a benefit to using transfer learning in the domain until after the model has been developed and evaluated.

Lisa Torrey and Jude Shavlik in transfer learning describe three possible benefits to look for when using transfer learning:

1. **Higher start**. The initial skill (before refining the model) on the source model is higher than it otherwise would be.
2. **Higher slope**. The rate of improvement of skill during training of the source model is steeper than it otherwise would be.
3. **Higher asymptote**. The converged skill of the trained model is better than it otherwise would be.



Ideally, you would see all three benefits from a successful application of transfer learning. It is an approach to try if you can identify a related task with abundant data and you have the resources to develop a model for that task and reuse it on your own problem, or there is a pre-trained model available that you can use as a starting point for your own model.
On some problems where you may not have very much data, transfer learning can enable you to develop skillful models that you simply could not develop in the absence of transfer learning. The choice of source data or source model is an open problem and may require domain expertise and/or intuition developed via experience.

Using Deep learning for image processing

Many of the tools use AI for solving complex image processing tasks. In fact, improvements in AI and machine learning are one of the reasons for the impressive progress in computer vision technology that we can see today.
Most effective machine learning models for image processing use neural networks and deep learning. Deep learning uses neural networks for solving complex tasks similarly to the way the human brain solves them.
Different types of neural networks can be deployed for solving different image processing tasks, from simple binary classification (whether an image does or doesn't match a specific criteria) to instance segmentation. Choosing the right type and architecture of a neural network plays an essential part in creating an efficient AI-based image processing solution.

Convolutional Neural Network

Convolutional Neural Networks (ConvNets or CNNs) are a class of deep learning networks that were created specifically for image processing. However, CNNs have been successfully applied on various types of data, not only images. In these networks, neurons are organized and connected similarly to how neurons are organized and connected in the human brain. In contrast to other neural networks, CNNs require fewer preprocessing operations. Plus, instead of using hand-engineered filters (despite being able to benefit from them), CNNs can learn the necessary filters and characteristics during training.

CNNs are multilayered neural networks that include input and output layers as well as a number of hidden layer blocks which consist of:

- Convolutional layers – Responsible for filtering the input image and extracting specific features such as edges, curves, and colors
- Pooling layers – Improve the detection of unusually placed objects
- Normalization (ReLU) layers – Improve network performance by normalizing the inputs of the previous layer
- Fully connected layers – Layers in which neurons have full connections to all activations in the previous layer (similar to regular neural networks)

All CNN layers are organized in three dimensions (weight, height, and depth) and have two components:

- Feature extraction
- Classification

In the first component, the CNN runs multiple convolutions and pooling operations in order to detect features it will then use for image classification.

In the second component, using the extracted features, the network algorithm attempts to predict what the object in the image could be with a calculated probability.

CNNs are widely used for implementing AI in image processing and solving such problems as signal processing, image classification, and image recognition. There are numerous types of CNN architectures such as AlexNet, ZFNet, Faster R-CNN, and GoogLeNet/Inception.

The choice of CNN architecture depends on the task at hand. For instance, GoogLeNet shows a higher accuracy for leaf recognition than AlexNet or a basic CNN. At the same time, due to the higher number of layers, GoogLeNet takes longer to run.

Mask R-CNN

Mask R-CNN is a Faster R-CNN-based deep neural network that can be used for separating objects in a processed image or video. This neural network works in two stages:

Segmentation – The neural network processes an image, detects areas that may contain objects, and generates proposals.

Generation of bounding boxes and masks – The network calculates a binary mask for each class and generates the final results based on these calculations.

This neural network model is flexible, adjustable, and provides better performance when compared to similar solutions. However, Mask R-CNN struggles with real-time processing, as this neural network is quite heavy and the mask layers add a bit of performance overhead, especially compared to Faster R-CNN.

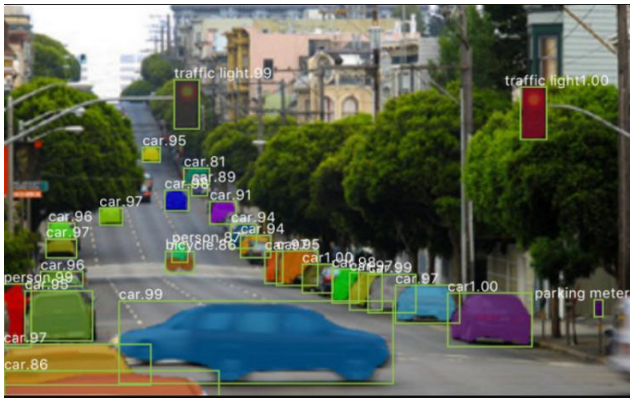


Figure 7. How Mask R-CNN works

Mask R-CNN remains one of the best solutions for instance segmentation. we have applied this neural network architecture and our image processing skills to solve many complex tasks, including the processing of medical image data and medical microscopic data.

Fully convolutional network

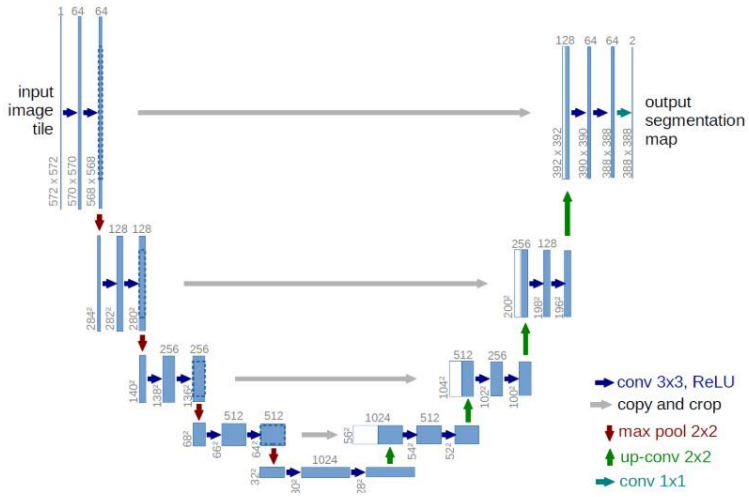
The concept of a fully convolutional network (FCN) was first offered by a team of researchers from the University of Berkeley. The main difference between a CNN and FCN is that the latter has a convolutional layer instead of a regular fully connected layer. As a result, FCNs are able to manage different input sizes. Also, FCNs use downsampling (striped convolution) and upsampling (transposed convolution) to make convolution operations less computationally expensive.

A fully convolutional neural network is the perfect fit for image segmentation tasks when the neural network divides the processed image into multiple pixel groupings which are then labeled and classified. Some of the most popular FCNs used for semantic segmentation are DeepLab, RefineNet, and Dilated Convolutions.

U-Net

U-Net is a convolutional neural network that allows for fast and precise image segmentation. In contrast to other neural networks on our list, U-Net was designed specifically for biomedical image segmentation. Therefore, it comes as no surprise that U-Net is believed to be superior to Mask R-CNN especially in such complex tasks as medical image processing.

U-Net has a U-shaped architecture and has more feature channels in its upsampling part. As a result, the network propagates context information to higher-resolution layers, thus creating a more or less symmetric expansive path to its contracting part.



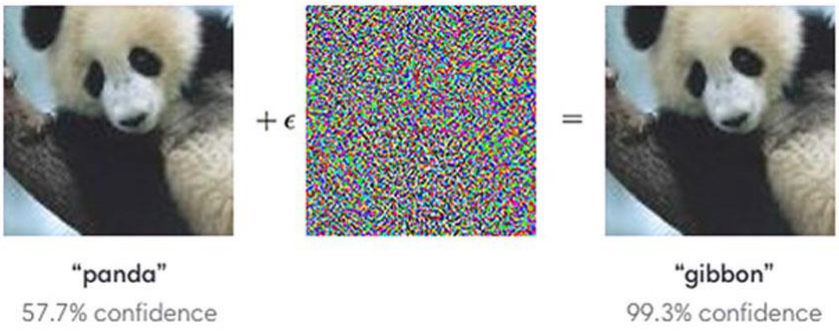
The U-Net neural network architecture

we successfully implemented a system with the U-Net backbone to complement the results of a medical image segmentation solution. This approach allowed us to get more diverse image processing results and permitted us to analyze the received results with two independent systems. Additional analysis is especially useful when a domain specialist feels unsure about a particular image segmentation result.

Generative Adversarial Network

Generative adversarial networks (GANs) are supposed to deal with one of the biggest challenges neural networks face these days: adversarial images.

Panda images are known for causing massive failures in neural networks. For instance, a neural network can be fooled if you add a layer of visual noise called perturbation to the original image. And even though the difference is nearly unnoticeable to the human brain, computer algorithms struggle to properly classify Panda images.



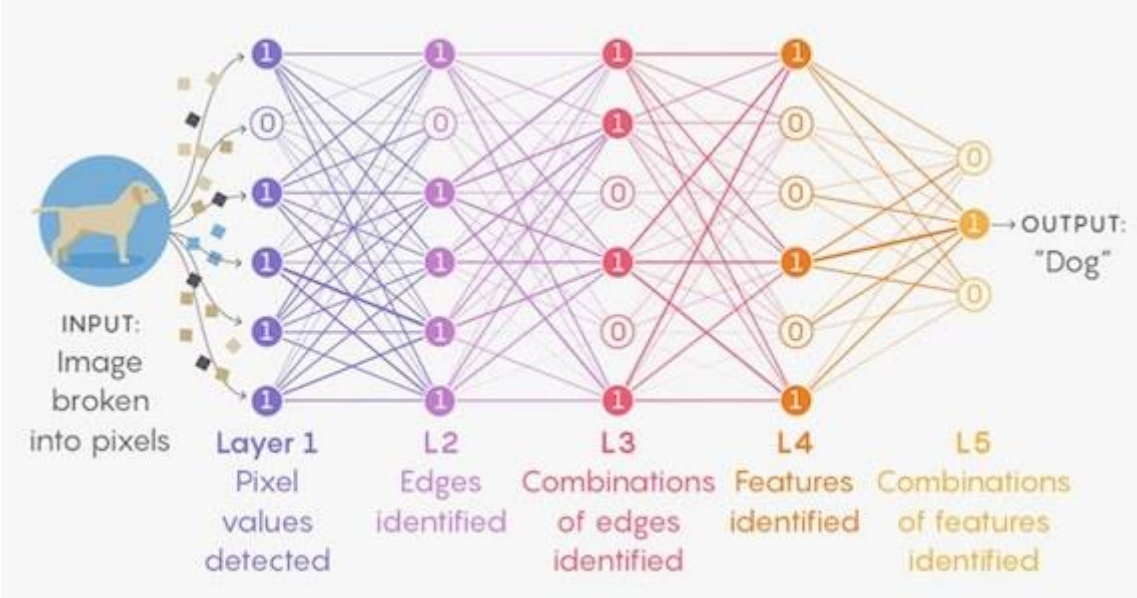
Example of Panda image misclassification

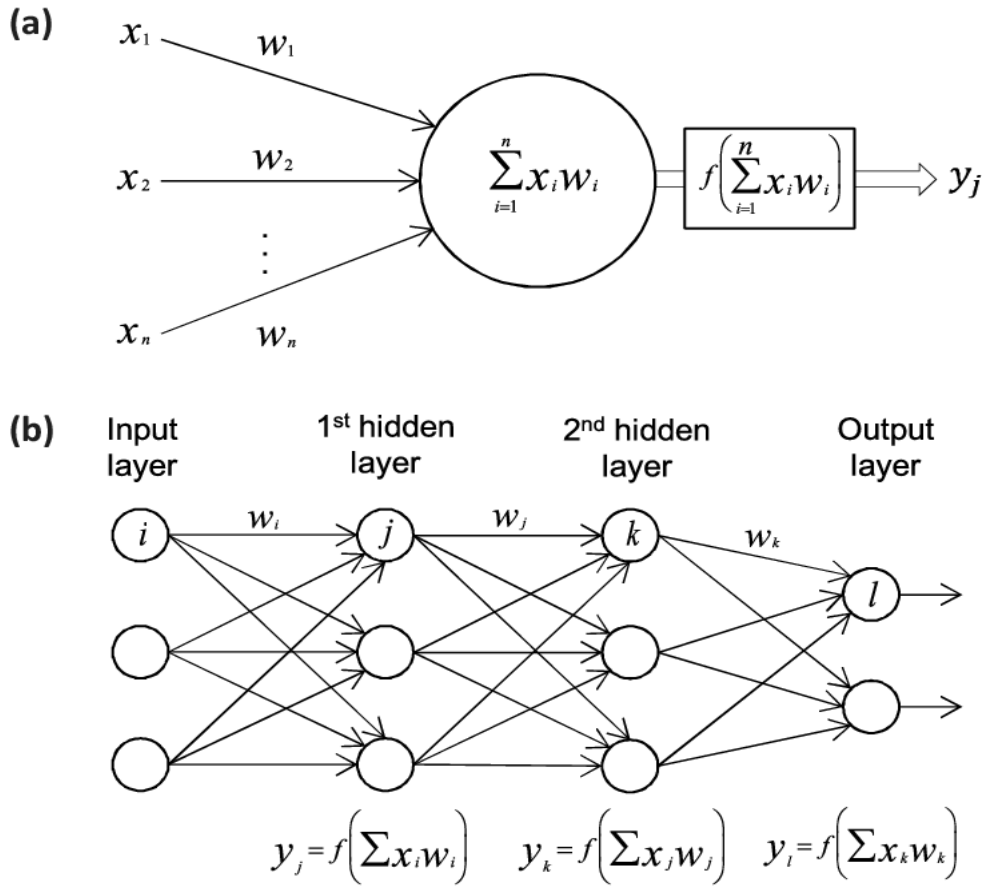
GANs are double networks that include two nets — a generator and a discriminator — that are pitted against each other. The generator is responsible for generating new data and the discriminator is supposed to evaluate that data for authenticity.
Plus, in contrast to other neural networks, GANs can be taught to create new data such as images, music, and prose.

Deep Learning

Deep neural network: the structure

To gain a better understanding of how to build a deep neural network, we need to understand all the elements a neural network contains. A neural network consists of layers, each consisting of nodes. There is an input layer which can be the image, or parts of the image. Then, there are several hidden layers which have the function of extracting image features.





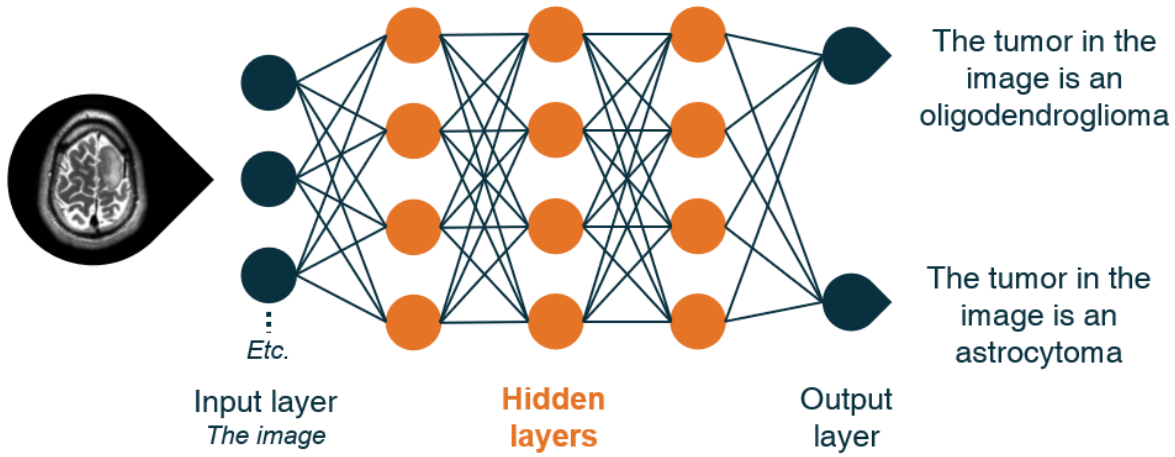


Figure: A deep neural network consists of an input layer, multiple hidden layers and an output layer, all consisting of nodes.

A node represents a basic calculation: it takes an input and calculates an output. The input consists of the output from all nodes in the previous layer (hence the connecting lines between nodes in figure 1). Inside the node, the output gets calculated and it is passed on to the next layer. Training your deep neural network is nothing more than figuring out the optimal output for each node, so that when all nodes are combined, the network gives the right response.

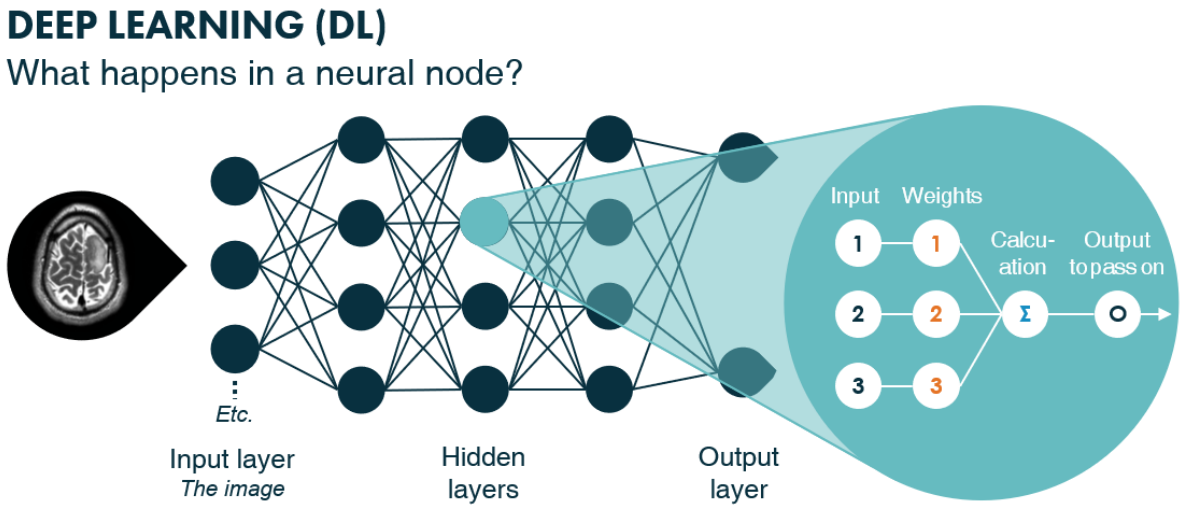


Figure: A node in a hidden layer of a deep neural network takes an input, performs a calculation, and passes on an output to nodes in the next layer.

The calculation consists of two parts: 1) a simple multiplication of all the inputs by the related "weight", and 2) the activation function. In the first part, each input is multiplied with a corresponding weight and the outcomes are summed. The second part, the activation function, is a bit more complex. It allows the network to perform more complex calculations, and thus, solve more advanced problems. After applying these two steps, the output of the node is passed on to multiple nodes in the next layer (to whom it becomes an input). See figure for a visual explanation of the calculation within a neural node. In a traditional neural network, usually the hidden layers are fully connected, meaning each node in the first hidden layer is connected to each node in the second, which is connected to each node in the third hidden layer, etc.

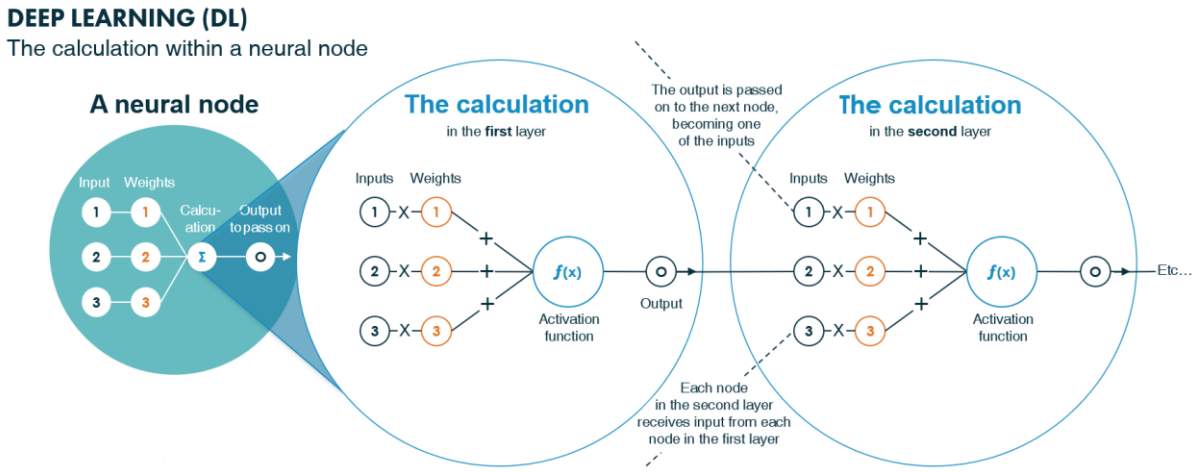


Figure: Each neural node in a deep neural network performs a calculation by taking inputs from nodes in the previous layer, multiplying these by weights, and summing all outcomes. The resulting number gets processed by an activation function to allow for more complex calculations. Lastly, the output is passed on to the nodes in the next layer.

Building a neural network in 3 steps

As we saw in the previous section, the weights of a neural network directly influence how input data is transformed and passed on from one node to the next. Hence, in the end, the weights control the value of the output. The process of determining the value of weights that give the correct output is called training. To get a grasp of how many weights we are dealing with, see figure.

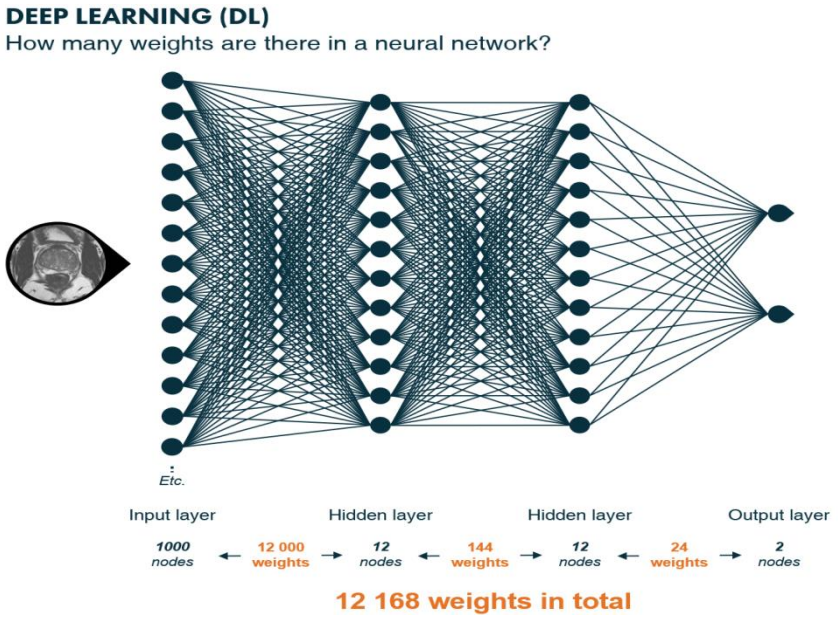


Figure 4: If the input layer (the image) consists of 10x10x10 voxels, you already start with 1000 nodes. Say your neural network contains two hidden layers with each 12 nodes and 2 output nodes, stating whether the input image does or does not contain a tumor. For each connection shown in the figure above, you need a weight, so this very simple network contains 1000 x 12 + 12 x 12 +2*12 = 12 168 weights.

In order to correctly train a deep neural network, example data is needed. This data consists of examples (in the case of deep learning in radiology, these are usually medical images) containing input and the corresponding correct output, i.e. the ground truth. The training procedure is usually performed using 3 independent datasets, each containing different examples: a training set, a validation set, and a test set. The datasets are used in three different steps of the training process.

Step 1: Initial training of the network using the training set

The first step is training of the network, i.e. taking a go at determining the value of each weight. This is done using back propagation.

To train the network using back propagation, you start by assigning all weights to random values. Subsequently, every image in the training set is run through the network. The output of the network, which will be completely random at this stage, is then compared to our ground truth. The difference between the output and the ground truth can be expressed as an error. For example, our network is supposed to compute the probability that our image contains a tumor. We insert an image containing a tumor and the network returns “32% chance of a tumor”. Comparing this response with the ground truth, in this case “100% chance of a tumor”, we can estimate an error of 68%, meaning the algorithm is pretty far off.

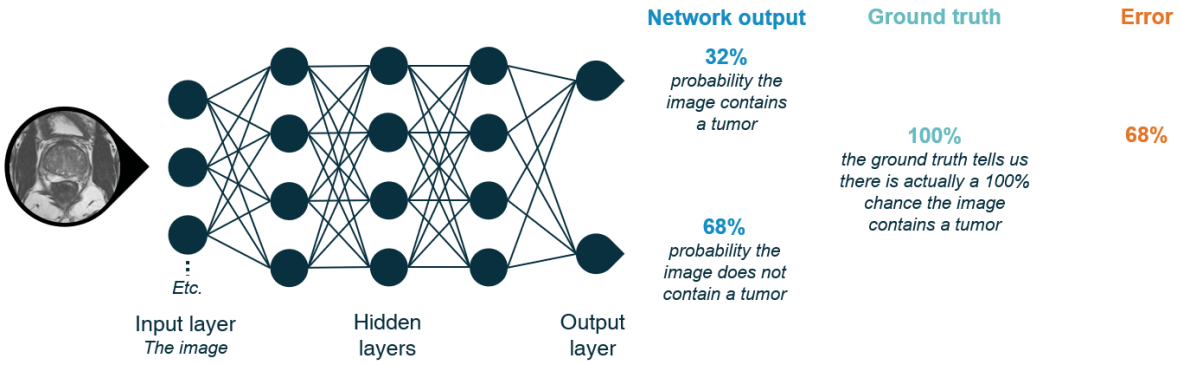


Figure: Calculation of error of one image. The image passes through the network generating an output. In this case, the network returns a 32% probability that a tumor is present in the image (and therefore a probability of 68% of the image not containing a tumor). The error is given by the difference between the ground truth, 100% probability of the image containing a tumor, and the output of the network, 32% probability of the image containing a tumor. Thus, the error is 68%.

The error of each example in the training set is then combined to obtain the total error of the network over the training set, which is nothing more than summing up the error for all images in the training set.

For our network to have an optimal performance, we need to minimize this total error. Stage enter back propagation calculation. This is achieved by adjusting all weights starting at the output layer and working our way back to the input layer. A commonly used method to iteratively adjust the weights, slowly approaching the right values, is called gradient descent. We will discuss this technique in a future blog post!

Step 2: Refining the network using the validation set

The training procedure described above will provide a neural network that has a very small error when applied to the training set. Roughly speaking, we can say that the deeper the network, i.e. the more hidden layers and more nodes it has, the better it is at learning from the training set. What might happen, however, is that the network learns too much from the given examples, or in other words, it over fits. This results in a great performance of the network on the training set, but a poor performance with new data. In other words, the network's performance is not generalizable.

To overcome over fitting, an independent dataset is used to evaluate the network's performance. This is the validation set. This is done during training, so we can check whether the algorithm performs well on the training set, but lousy on the validation set. If this is the case, you have over fitted and you will need to go back to the drawing board to make adjustments. There are several ways to do this. For example, you could:

- Use more data. This is easier said than done, because usually if you had more data at the start, you would have used it already.
- Apply augmentation. If there is not enough original data available, it can be created in an artificial way. Take the original dataset and transform the images such that they are different, but still resemble a credible example. This transformation may be cropping, translating, rotating, resizing, stretching, or randomly deforming them. We will discuss augmentation in a future blog post!
- Apply drop out. This method is applied during training of the network by ignoring random neurons with each iteration during training, i.e. “turning them off”. By doing this, other neurons will pick up the task of the ignored neurons. Hence, instead of the

network becoming too dependent on specific weights between specific neurons, the interdependencies distribute themselves over the whole network.[4]

- Apply regularization to the weights. By introducing other conditions or extra controls on the weights, or actually the total of all weights, you can steer the network away from over fitting.

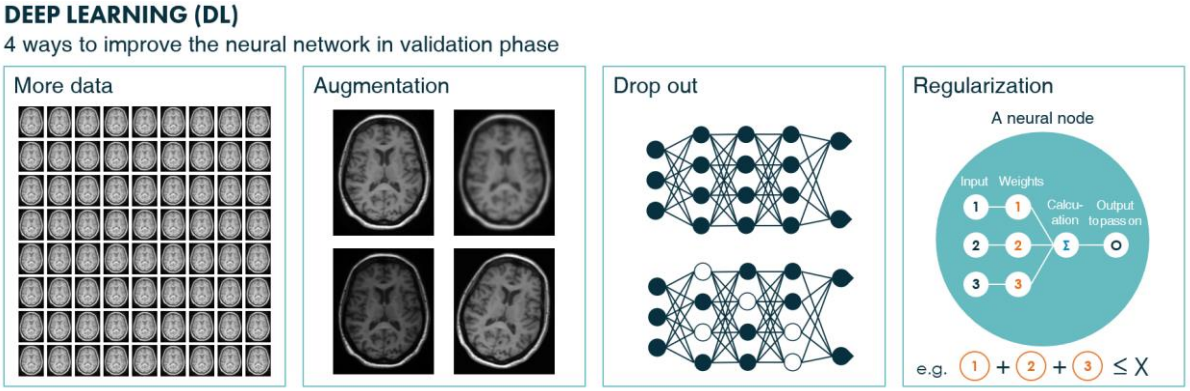


Figure : There are several ways to improve the algorithm in case of overfitting: adding more data to the training set, performing augmentation, applying drop out, or regularization.

Step 3: Using the test set

The very last step, testing, uses a set of data that has been kept apart. This step is very straightforward: the algorithm processes the images in the test set and performance measurements, such as accuracy, recall, or DICE score are calculated. Hence, this is an independent test of how the algorithm performs on data that has not been seen before. After the algorithm performance has been checked using this test set, there is no going back. After all, tweaking the algorithm based on the test results would make the test set everything but unrelated to the training of the network.

TEXT / REFERENCE BOOKS

1. **Rafael C. Gonzalez, Richard E. Woods," Digital Image Processing", Pearson , Second Edition, 2004**
2. **Anil K. Jain, "Fundamentals of Digital Image Processin", Pearson 2002**
3. **Robert J.Schalkoff, "Pattern Recognition Statistical, Structural and Neural Approaches",John Wiley & Sons Inc., New York, 1992**
4. **R.O.Duda, P.E.Hart and D.G.Stork, "Pattern Classification", John Wiley, 2001**
5. **Himanshu Singh. "Practical Machine Learning and Image Processing", Apress, 2019**
6. **François Chollet "Deep Learning with Python", Manning Publications Co., NY, 2018**
7. **Phil Kim. "MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence", Apress, 2017**
8. **Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images**
9. **Valentina Zharkova, Lakhmi C. Jain, "Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images", Springer, 2007**