UNIT-2

INTENSITY TRANSFORMATIONS & SPATIAL FILTERING

Image Enhancement technique is to improve the quality of an image even if the degradation is available. This can be achieved by increasing the dominance of some features or decreasing the ambiguity between different regions. Image enhancement approaches fall into two broad categories: spatial domain methods and frequency domain methods. The term *spatial domain* refers to the image plane itself, and approaches in this category are based on direct manipulation of pixels in an image. *Frequency domain* processing techniques are based on modifying the Fourier transform of an image.

The two principal categories of spatial domain are Intensity transformations and spatial filtering.

- The Intensity transformations operate on single pixel of an image, for the purpose of contrast manipulation and image thresholding.
- Spatial filtering deals with performing operations such as image sharpening by working in neighborhood of every pixel in an image.

2.1 The Basics of Intensity transformations and Spatial filtering

The term *spatial domain* refers to the aggregate of pixels composing an image. The Spatial domain processes will be denoted by the expression

$$g(x,y) = T[f(x, y)]$$

Where f(x, y) is the input image, g(x, y) is the processed image, and T is an operator on f, defined over some neighborhood of (x, y). The operator can apply to a single image or to a set of images, such as performing the pixel-by-pixel sum of a sequence of images for noise reduction. The following figure shows the basic implementation of spatial domain on a single image.

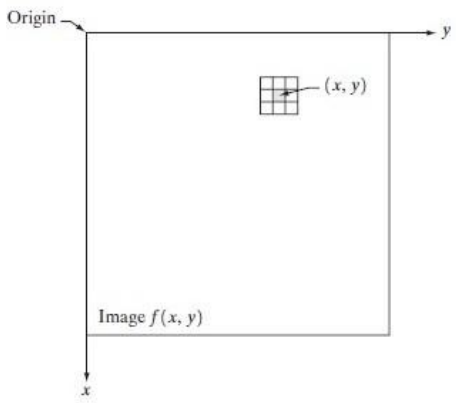


Fig: A 3×3 neighborhood about a point (x, y) in an image in the spatial domain.

The point (x, y) is an arbitrary location in the image and the small region shown containing the point is a neighborhood of (x, y). The neighborhood is rectangular, centered on (x, y) and much smaller in size than the image.

The process consists of moving the origin of the neighborhood from pixel to pixel and applying the operator T to the pixels in the neighborhood to yield the output at that location. Thus for any specific location (x, y) the value of the output image *g* at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) in *f*. This procedure is called spatial filtering, in which the neighborhood, along with a predefined operation is called a spatial filter. The smallest possible neighborhood is of size 1×1. In this case, *g* depends only on the value of *f* at a single point (x, y) and T becomes an intensity transformation or gray level mapping of the form

$$s = T(r)$$

Where s and r are variables represents the intensity of *g* and *f* at any point (x, y). The effect of applying the transformation T(r) to every pixel of *f* to generate the corresponding pixels in g would produce an image of higher contrast than the original by darkening the levels below *m* and brightening the levels above *m* in the original image. This is known as *contrast stretching* (Fig.(a)), the values of *r* below *m* are compressed by the transformation function into a narrow range of *s*, toward black. The opposite effect takes place for values of *r* above *m*. In the limiting case shown in Fig.(b), T(r) produces a two-level (binary) image. A mapping of this form is called a *thresholding* function. Hence the enhancement at any point in an image depends only on the gray level at that point, and the techniques in this category are referred to as *point processing*.

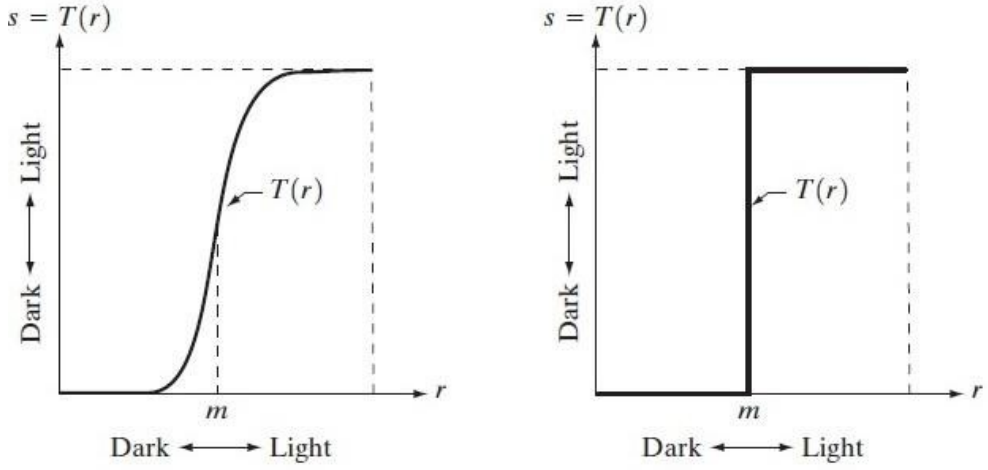


Fig: Intensity transformation functions (a) Contrast Stretching (b) Thresholding

2.2 Intensity transformation Functions

There are three basic types of functions used frequently for image enhancement: linear (negative and identity transformations), logarithmic (log and inverse-log transformations), and power-law (*n*th power and *n*th root transformations).The identity function is the trivial case in which output intensities are identical to input intensities. It is included in the graph only for completeness.

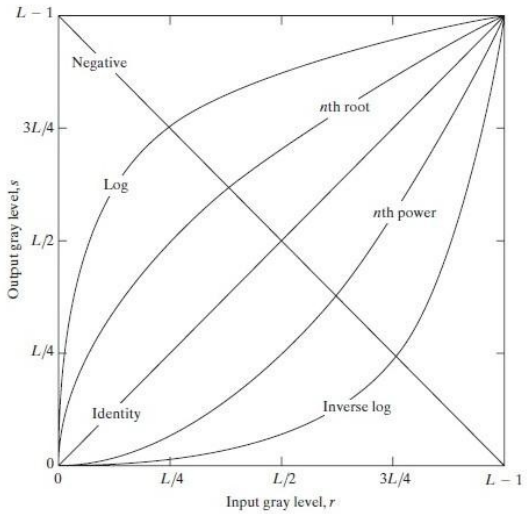


Fig: Some basic Intensity transformation functions used for image enhancement.

Image Negatives

The negative of an image with gray levels in the range [0, L-1]is obtained by using the negative transformation which is given by the expression

$$s = L - 1 - r$$

Reversing the intensity levels of an image in this manner produces the equivalent of a photographic negative. This type of processing is particularly suited for enhancing white or gray detail embedded in dark regions of an image, especially when the black areas are dominant in size.

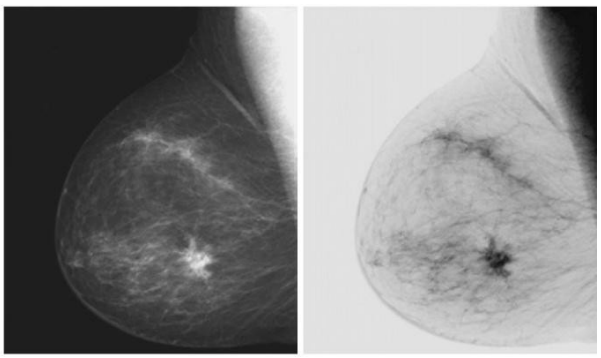


Fig: (a) Original digital mammogram. (b) Negative image obtained using the negative transformation

Log Transformations

The general form of the log transformation is

$$S=c \log (1+r)$$

where *c* is a constant, and it is assumed that $r \geq 0$.The shape of the log curve shows that this transformation maps a narrow range of low gray-level values in the input image into a wider range of output levels. The opposite is true of higher values of input levels. We would use a transformation of this type to expand the values of dark pixels in an image while compressing the higher-level values. The opposite is true of the inverse log transformation. The log transformation function has an important characteristic that it compresses the dynamic range of images with large variations in pixel values. Log transformation is basically employed in Fourier transform.

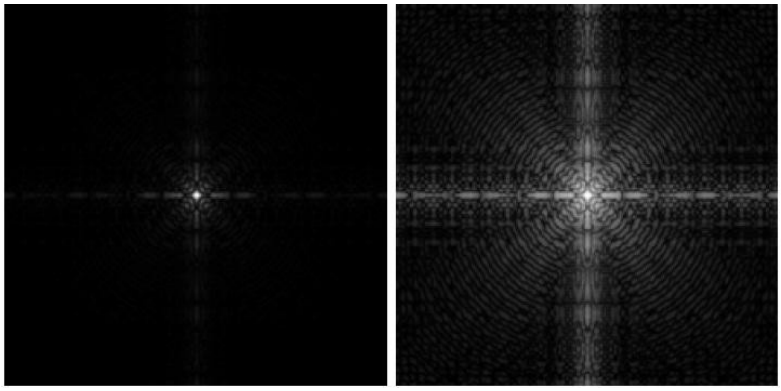


Fig: (a) Fourier spectrum. (b) Result of applying the log transformation

Power –Law Transformations

Power-law transformations have the basic form $s = cr^{\gamma}$

Where *c* and γ are positive constants. However Plots of *s* versus *r* for various values of γ are shown in the following figure.

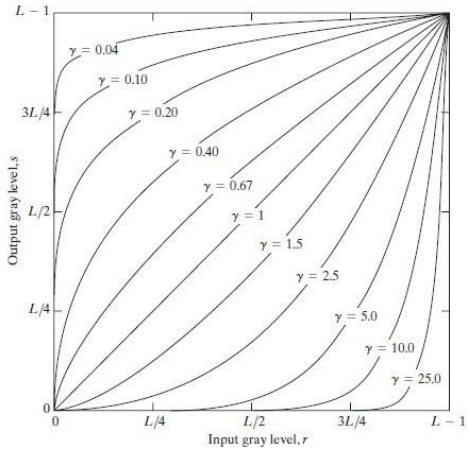


Fig: Plots of the equation $s=cr^{\gamma}$ for various values of g (c=1 in all cases).

The curves generated with values of $\gamma>1$ have exactly the opposite effect as those generated with values of $\gamma<1$. It reduces to the identity transformation when $c=\gamma=1$. The power law transformation used in a variety of devices for image capture, printing, and display. The exponent in the power-law equation is referred to as *gamma is* used to correct this power-law response phenomena is called *gamma correction*.

Piecewise-Linear Transformation Functions

The principal advantage of piecewise linear functions is that these functions can be arbitrarily complex. But their specification requires considerably more user input. These transformations are of 3 types.

Contrast Stretching:

Contrast Stretching is a process that expands the range of intensity values in an image, in order to utilize the dynamic range of intensity values. It is one of the simplest piecewise linear function. Low contrast images can result from poor illumination. The following figure shows that typical transformation used for contrast stretching.

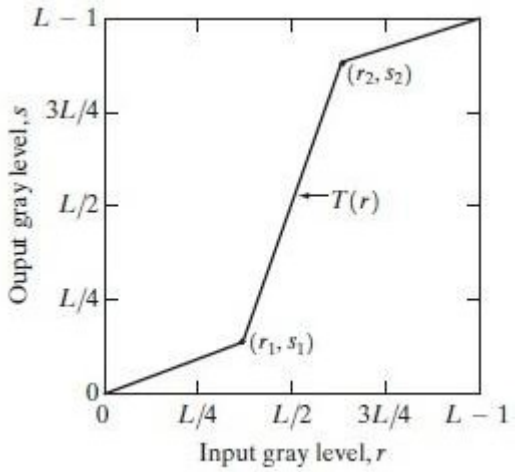


Fig: Form of transformation function

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=0and 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 that the function is single valued and monotonically increasing. This condition preserves the order of gray levels, thus preventing the creation of intensity artifacts in the processed image.

Intensity-Level Slicing:

The process of highlighting a specific range of intensities in an image is known as Intensity-Level Slicing. There are two basic approaches can be adopted for Intensity-Level Slicing.

- One approach is to display a high value for all gray levels in the range of interest and a low value for all other intensities. This transformation produces a binary image.
- The second approach, based on the transformation, brightens the desired range of gray levels but preserves the background and gray-level in the image without change.

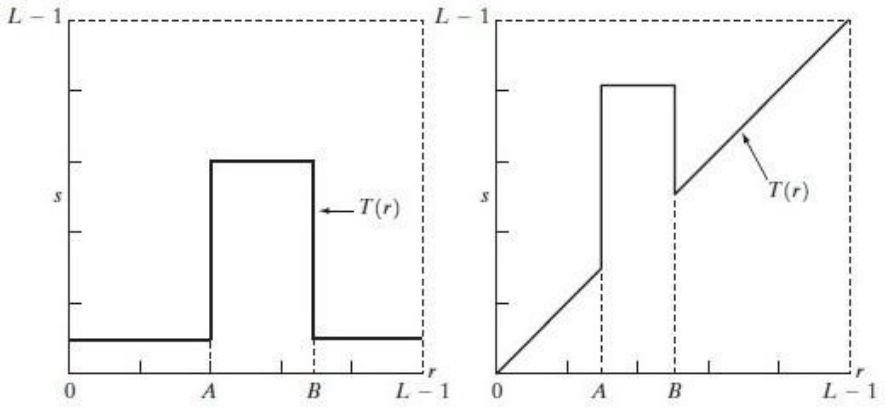


Fig: (a) The transformation highlights range [A, B] of gray levels and reduces all others to a constant level. (b) The transformation highlights range [A, B] but preserves all other levels.

Bit-Plane Slicing:

Instead of highlighting gray-level ranges, highlighting the contribution made to total image appearance by specific bits might be desired. Suppose that each pixel in an image is represented by 8 bits. Imagine that the image is composed of eight 1-bit planes, ranging from bit-plane 0 for the least significant bit to bit plane 7 for the most significant bit. In terms of 8-bit bytes, plane 0 contains all the lowest order bits in the bytes comprising the pixels in the image and plane 7 contains all the high-order bits. The following figure shows the various bit planes for an image.

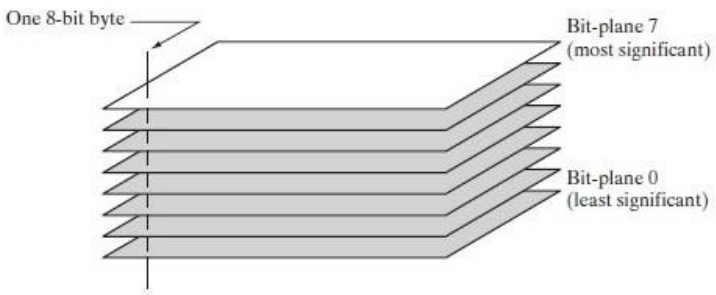


Fig: Bit Plane Representation of an 8-bit image.

The higher-order bits contain the majority of the visually significant data. The other bit planes contribute to more subtle details in the image. Separating a digital image into its bit planes is useful for analyzing the relative importance played by each bit of the image, a process that aids in determining the adequacy of the number of bits used to quantize each pixel.

2.3 Histogram Processing

Histogram of an image is defined as the representation including relative frequency of occurrence of various gray levels in the image. In general, 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. A normalized histogram can be obtained by dividing each of its values by the total number of pixels in the image and it is given by the equation,

$$p(r_k) = \frac{n_k}{M N} \qquad \text{for } k = 0, 1, 2, \ldots..L-1$$

Histograms are the basis for numerous spatial domain processing techniques. Histogram manipulation can be used effectively for image enhancement and also is quite useful in other image processing applications, such as image compression and segmentation. Histograms are simple to calculate in software and also lend themselves to economic hardware implementations, thus making them a popular tool for real-time image processing.

The purpose of histogram is to classify the image falls in which category. Generally images are classified as follows,

- **Dark images:** The components of the histogram are concentrated on the low side of the intensity scale.
- **Bright images:** The components of the histogram are biased towards the high side of the intensity scale.
- **Low contrast:** An image with low contrast has a histogram that will be narrow and will be centered toward the middle of the gray scale.
- **High contrast:** The components of histogram in the high-contrast image cover a broad range of the gray scale.

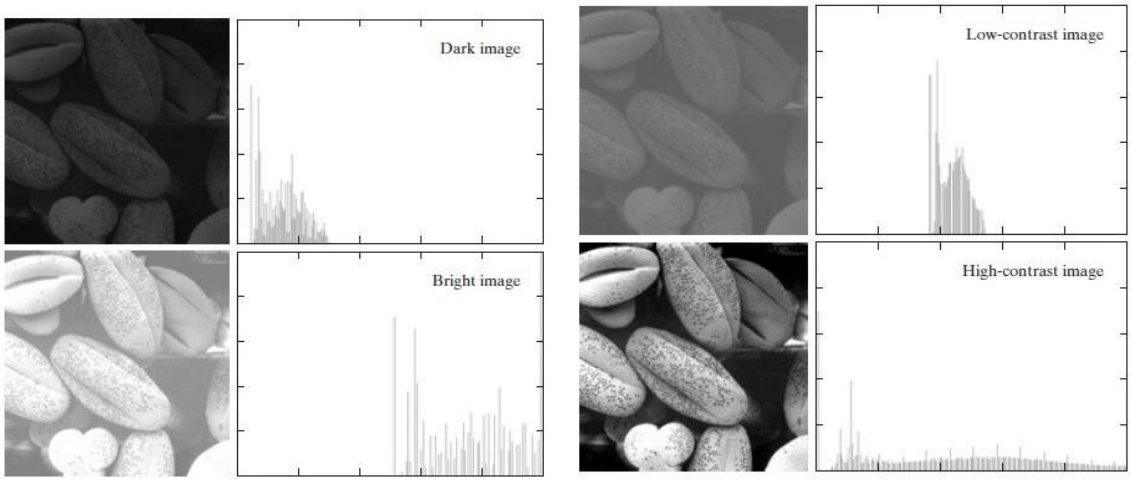


Fig: Dark, light, low contrast, high contrast images, and their corresponding histograms.

Histogram Equalization

Let the variable r represent the gray levels of the image to be enhanced. We assume that r has been normalized to the interval [0, *L*-1], with r=0 representing black and r=*L*-1 representing white. For any r the conditions, then transformations of the form,

$$s = T(r) \qquad 0 \le r \le L\text{-}1$$

It produces an output intensity level s for every pixel in the input image having intensity *r*. Assume that the transformation function T(r) satisfies the following conditions:

(a) T(r) is single-valued and monotonically increasing function in the interval $0 \le r \le L\text{-}1$

(b) $0 \le T(r) \le L\text{-}1$ for $0 \le r \le L\text{-}1$

The transformation function should be single valued so that the inverse transformations should exist. T(r) is monotonically increasing condition guarantees that the output intensity values never be less than corresponding input values. The second conditions guarantee that the output gray levels will be in the same range as the input levels. The gray levels of the image may be viewed as random variables in the interval [0, *L*-1]. The most fundamental descriptor of a random variable is its probability density function (PDF). Let $P_r(r)$ and $P_s(s)$ denote the probability density functions of random variables r and s respectively. If pr(r) and T(r) are known and $T^{-1}(s)$ satisfies condition (a), then the probability density function $P_s(s)$ of the transformed variable *s* can be obtained by

$$P_s(s) = P_r(r) \frac{dr}{ds}$$

Thus, the probability density function of the transformed variable, *s* is determined by the PDF of the input gray-level and by the chosen transformation function. A transformation function of particular importance in image processing has the form

$$s = T(r) = \int_0^r P_r(w)dw$$

Where w is a dummy variable of integration. The right side of this equation is recognized as the cumulative distribution function (CDF) of random variable *r*. Since probability density functions are always positive, and recalling that the integral of a function is the area under the function, it follows that this transformation function is single valued and monotonically increasing, and, therefore, satisfies condition (a). Similarly, the integral of a probability density function for variables in the range [0, *L*-1], so condition (b) is satisfied. Using this definition of T we see that the derivative of s with respect to r is,

$$\begin{aligned}\frac{ds}{dr} &= \frac{dT(r)}{dr} \\ &= \frac{d}{dr}\left[\int_0^r p_r(w)\,dw\right] \\ &= p_r(r).\end{aligned}$$

Substituting this result dr/ds

$$\begin{aligned}p_s(s) &= p_r(r)\left|\frac{dr}{ds}\right| \\ &= p_r(r)\left|\frac{1}{p_r(r)}\right| \\ &= 1 \qquad 0 \le s \le 1.\end{aligned}$$

Hence the $P_s(s)$ is a uniform probability function and independent of the form $P_r(r)$. For discrete values we deal with probabilities and summations instead of probability density functions and integrals. The probability of occurrence of gray level r_k in an image is approximated by

$$Pr(r) = \frac{n_k}{MN} \quad k = 0, 1, 2, \ldots\ldots L\text{-}1$$

Where MN is the total number of pixels in the image, n_k is the number of pixels that have gray level r_k and *L* is the number of possible intensity levels in the image. The discrete version of the transformation function given is

$$s_k = T_k(r) = \sum_{j=0}^{k} P_r(r_j)$$

Thus, a processed (output) image is obtained by mapping each pixel with level r_k in the input image into a corresponding pixel with level s_k in the output image. Hence the transformation is called *histogram equalization* or *histogram linearization.*

Histogram Matching

Histogram equalization automatically determines a transformation function that seeks to produce an output image that has a uniform histogram. This is a good approach because the results from this technique are predictable and the method is simple to implement. However in some applications the enhancement on a uniform histogram is not the best approach. In particular, it is useful sometimes to be able to specify the shape of the histogram that we wish the processed image to have. The method used to generate a processed image that has a specified histogram is called *histogram matching* or *histogram specification*.

Let us consider a continuous gray levels *r* and *z* (considered continuous random variables), and let $p_r(r)$ and $p_z(z)$ denote their corresponding continuous probability density functions. Where *r* and *z* denote the gray levels of the input and output (processed) images respectively. We can estimate $p_r(r)$ from the given input image, while $p_z(z)$ is the *specified* probability density function that we wish the output image to have. Let *s* be a random variable with the property,

$$s = T(r) = \int_0^r P_r(w)dw$$

Where w is a dummy variable of integration. This is a continuous version of histogram equalization. Now we define a random variable *z* with the property

$$G(z) = \int_0^z P_z(t)dt = s$$

Where *t* is a dummy variable of integration, the two equations that G(z)=T(r) and, therefore, that *z* must satisfy the condition

$$Z = G^{-1}(s) = G^{-1}[T(r)]$$

The transformation T(r) can be obtained once $p_r(r)$ has been estimated from the input image. Similarly, the transformation function G(z) can be obtained when $p_z(z)$ is given. Assuming that G^{-1} exists and show that an image with a specified probability density function can be obtained from an input image by using the following procedure:

- Obtain the transformation function T(r) by using $s = T(r) = \int_0^r P_r(w)dw$
- Obtain the transformation function G (z) by using $G(z) = \int_0^z P_z(t)dt = s$
- Obtain the inverse transformation function G^{-1} by using $Z = G^{-1}(s) = G^{-1}[T(r)]$

- Obtain the output image by applying $Z= G^{-1}(s) = G^{-1}[T(r)]$ to all the pixels in the input image. The result of this procedure will be an image whose gray levels, *z*, have the specified probability density function $p_z(z)$.

Local Histogram Processing

The histogram processing methods are *global*, in the sense that pixels are modified by a transformation function based on the intensity distribution of an entire image. Although this global approach is suitable for overall enhancement, there are cases in which it is necessary to enhance details over small areas in an image. The number of pixels in these areas may have negligible influence on the computation of a global transformation whose shape does not necessarily guarantee the desired local enhancement. The solution is to devise transformation functions based on the intensity distribution or other properties in the neighborhood of every pixel in the image.

The histogram processing techniques previously described are easily adaptable to local enhancement. The procedure is to define a square or rectangular neighborhood and move the center of this area from pixel to pixel. At each location, the histogram of the points in the neighborhood is computed and either a histogram equalization or histogram specification transformation function is obtained. This function is finally used to map the intensity of the pixel centered in the neighborhood. The center of the neighborhood region is then moved to an adjacent pixel location and the procedure is repeated. Since only one new row or column of the neighborhood changes during a pixel-to-pixel translation of the region, updating the histogram obtained in the previous location with the new data introduced at each motion step is possible. This approach has obvious advantages over repeatedly computing the histogram over all pixels in the neighborhood region each time the region is moved one pixel location. Another approach used some times to reduce computation is to utilize non overlapping regions, but this method usually produces an undesirable checkerboard effect.

2.4 Enhancement Using Arithmetic/Logic Operations

Arithmetic/logic operations involving images are performed on a pixel-by-pixel basis between two or more images. For an example, subtraction of two images results in a new image whose pixel at coordinates (x, y) is the difference between the pixels in that same location in the two images being subtracted. Depending on the hardware and/or software being used, the actual mechanics of implementing arithmetic/logic operations can be done sequentially, one pixel at a time, or in parallel, where all operations are performed simultaneously.

Logic operations similarly operate on a pixel-by-pixel basis. Basically we can implement the AND, OR, and NOT logic operators because these three operators are *functionally complete*. Any other logic operator can be implemented by using only these three basic functions. The logic operations on gray-scale images, pixel values are processed as strings of binary numbers.

Image Subtraction

The difference between two images f(x, y) and h(x, y), expressed as

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

It is obtained by computing the difference between all pairs of corresponding pixels from *f* and *h*. The key usefulness of subtraction is the enhancement of *differences* between images. Generally the higher-order bit planes of an image carry a significant amount of visually relevant detail, while the lower planes contribute more to fine (often imperceptible) detail.

Image Averaging

Consider a noisy image g(x, y) formed by the addition of noise η (x, y) to an original image f(x, y); that is,

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

If an image $\bar{g}(x, y)$ is formed by averaging *K* different noisy images,

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

$$E\{\bar{g}(x, y)\} = f(x, y)$$

and

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

where $E\{\bar{g}(x, y)\}$ is the expected value of $\bar{g}$, and $\sigma^2_{\bar{g}(x,y)}$ and $\sigma^2_{\bar{\eta}(x,y)}$ are the variances of $\bar{g}$ and η, all at coordinates (x, y). The standard deviation at any point in the average image is

$$\sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{K}}\sigma_{\eta(x,y)}.$$

As *K* increases, the variability (noise) of the pixel values at each location (x, y) decreases. Because $E\{\bar{g}(x, y)\} = f(x, y)$, this means that $\bar{g}(x, y)$ approaches f(x, y) as the number of noisy images used in the averaging process increases. The images $g_i(x, y)$ must be registered (aligned) in order to avoid the introduction of blurring and other artifacts in the output image.

2.5 Basics of Spatial Filtering

Spatial Filter consists of a neighborhood and a predefined operation that is performed on the image pixels by the neighborhood. Filtering creates anew pixel with coordinates equal to the coordinates of the center of the neighborhood, and whose value is the result of the filtering operation. A processed image is generated as the center of the filter visits each pixel in the input image. If the operation performed on the image pixels is linear, then the filter is called a *linear spatial filter* otherwise *nonlinear*.

The mechanics of spatial filtering are illustrated in the following figure. The process consists simply of moving the filter mask from point to point in an image. At each point (x, y), the *response* g(x, y) of the filter at that point is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask.

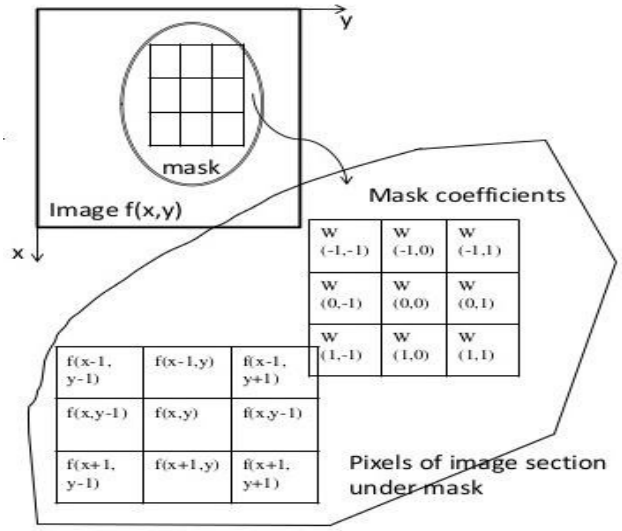


Fig: The mechanics of linear spatial filtering using a 3×3 filter mask.

For the 3×3 mask shown in the figure, the result (or response), g(x, y) of linear filtering with the filter mask at a point (x, y) in the image is

$$g(x, y) = w(-1,-1)f(x-1, y-1) + w(-1,0)f(x-1, y) + \cdots + w(0,0)f(x, y) + \cdots + w(1,0)f(x+1, y) + w(1,1)f(x+1, y+1)$$

It is the sum of products of the mask coefficients with the corresponding pixels directly under the mask. Observe that the coefficient w (0, 0) coincides with image value f(x, y), indicating that the mask is centered at (x, y) when the computation of the sum of products takes place. For a mask of size m×n, we assume that m=2a+1 and n=2b+1, where a and b are nonnegative integers.

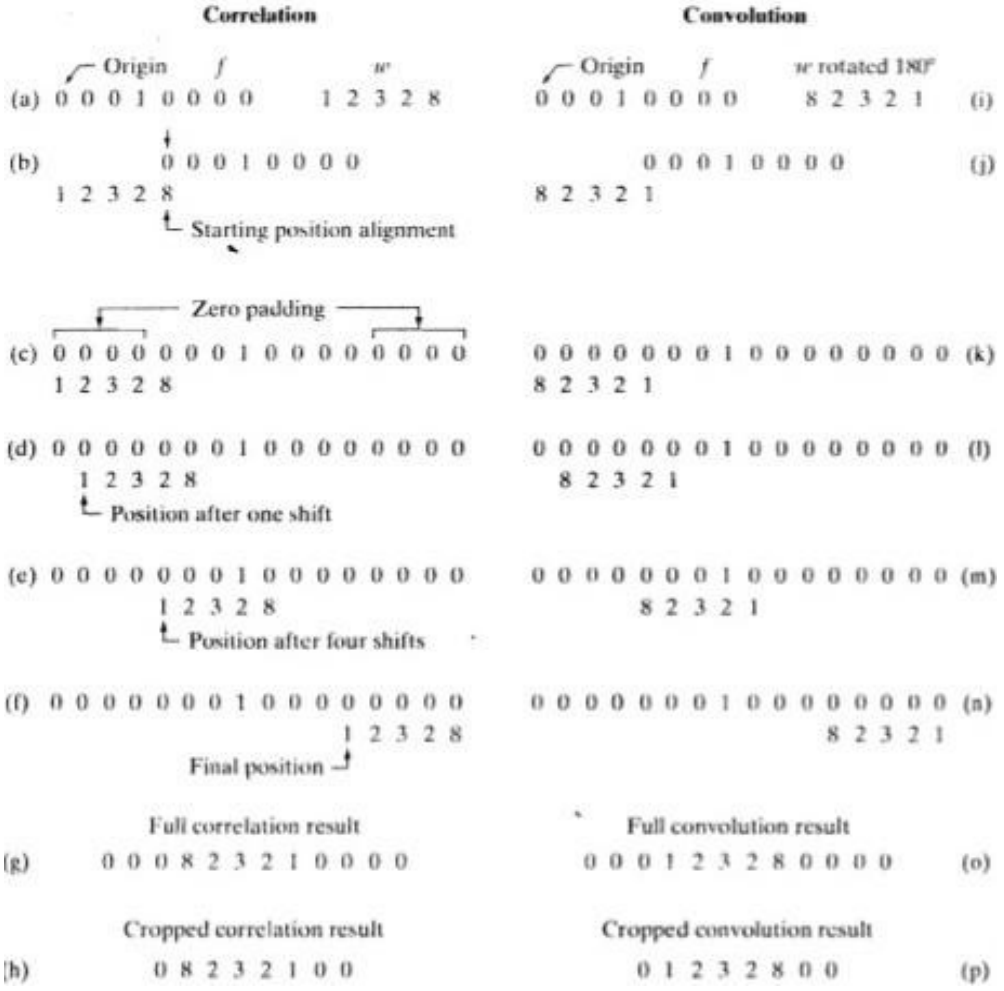
In general, linear filtering of an image f of size M×N with a filter mask of size m×n is given by the expression:

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

Where x and y are varied so that each pixel in w visits every pixel in f

Spatial Correlation and Convolution

Correlation is the process of moving a filter mask over the image and computing the sum of products at each location. The mechanism of convolution is the same except that the filter is first rotated by 180^0. The difference between the correlation and convolution can be explained with a 1-D image as follows.



The correlation of a filter w (x, y) of size m×n with an image is given by the equation

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

The convolution of a filter *w* (x, y) of size m×n with an image is given by the equation

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

2.6 Smoothing Spatial Filters

Smoothing filters are used for blurring and for noise reduction. Blurring is used in preprocessing steps, such as removal of small details from an image prior to object extraction, and bridging of small gaps in lines or curves. Noise reduction can be accomplished by blurring with a linear filter and also by nonlinear filtering.

Smoothing Linear Filters

The output (response) of a smoothing, linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask. These filters are called *averaging filters* and also are referred to a *lowpass filters*.

The smoothing filters can replace the value of every pixel in an image by the average of the gray levels in the neighborhood defined by the filter mask, this process results in an image with reduced "sharp" transitions in gray levels for noise reduction. However, edges are characterized by sharp transitions in gray levels, so averaging filters have the undesirable side effect that they blur edges. A major use of averaging filters is in the reduction of "irrelevant" detail in an image. The following 3×3 smoothing filter yields the standard average of the pixels under the mask and can be obtained by substituting the coefficients of the mask into

$$R = \frac{1}{9} \sum_{i=1}^{9} z_i$$

It is the average of the gray levels of the pixels in the 3×3 neighborhood defined by the mask. An m×n mask would have a normalizing constant equal to 1/mn. A spatial averaging filter in which all coefficients are equal is called a *box filter*.

$\frac{1}{9} \times$

1	1	1
1	1	1
1	1	1

Average Filter Mask

$\frac{1}{16} \times$

1	2	1
2	4	2
1	2	1

Weighted Average Filter Mask

The second mask is called *weighted average* which is used to indicate that pixels are multiplied by different coefficients. In this mask the pixel at the center of the mask is multiplied by a higher value than any other, thus giving this pixel more importance in the calculation of the average. The other pixels are inversely weighted as a function of their distance from the center of the mask. The diagonal terms are further away from the center than the orthogonal neighbors and, thus, are weighed less than these immediate neighbors of the center pixel.

The general implementation for filtering an M×N image with a weighted averaging filter of size m×n (m and *n* odd) is given by the expression

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

The complete filtered image is obtained by applying the above equation for x=0, 1, 2, ……. M-1 and y=0, 1, 2,……., N-1. The denominator is simply the sum of the mask coefficients and, therefore, it is a constant that needs to be computed only once. This scale factor is applied to all the pixels of the output image after the filtering process is completed.

Order-Statistics Filters

Order-statistics filters are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result. The best-known example in this category is the "*Median filter"*. It replaces the value of a pixel by the median of the gray levels in the neighborhood of that pixel. Median filters are quite popular because, they provide excellent noise-reduction capabilities. They are effective in the presence of *impulse noise*, also called *salt-and-pepper noise* because of its appearance as white and black dots superimposed on an image.

In order to perform median filtering at a point in an image, we first sort the values of the pixel in question and its neighbors, determine their median, and assign this value to that pixel. For example, in a 3×3 neighborhood the median is the 5th largest value, in a 5×5 neighborhood the 13th largest value, and so on. When several values in a neighborhood are the same, all equal values are grouped. For example, suppose that a 3×3 neighborhood has values (10, 20, 20, 20, 15, 20, 20, 25, 100). These values are sorted as (10, 15, 20, 20, 20, 20, 20, 25, 100), which results in a median of 20. Thus, the principal function of median filters is to force points with distinct gray levels to be more like their neighbors.

The median represents the 50th percentile of a ranked set of numbers, but the ranking lends itself to many other possibilities. For example, using the 100th percentile filter is called *max filter*, which is useful to find the brightest points in an image. The 0th percentile filter is the *min filter*, used for the opposite purpose.

2.7 Sharpening Spatial Filters

The principal objective of sharpening is to highlight fine detail in an image or to enhance detail that has been blurred, either in error or as a natural effect of a particular method of image acquisition. It includes applications ranging from electronic printing and medical imaging to industrial inspection and autonomous guidance in military systems.

As smoothing can be achieved by integration, sharpening can be achieved by spatial differentiation. The strength of response of derivative operator is proportional to the degree of discontinuity of the image at that point at which the operator is applied. Thus image differentiation enhances edges and other discontinuities and deemphasizes the areas with slow varying grey levels.

The derivatives of a digital function are defined in terms of differences. There are various ways to define these differences. A basic definition of the first-order derivative of a one-dimensional image f(x) is the difference

$$\frac{df}{dx} = f(x+1) - f(x)$$

The first order derivative must satisfy the properties such as,

- Must be zero in the areas of constant gray-level values.
- Must be nonzero at the onset of a gray-level step or ramp.
- Must be nonzero along ramps.

Similarly, we define a second-order derivative as the difference

$$\frac{d^2f}{dx^2} = f(x+1) + f(x-1) - 2f(x)$$

The second order derivative must satisfy the properties such as,

- Must be zero in the areas of constant gray-level values.
- Must be nonzero at the onset and end of a gray-level step or ramp.
- Must be zero along ramps of constant slope.

Use of Second Derivatives for Enhancement–The Laplacian

The second order derivatives in image processing are implemented by using the *Laplacian operator.* The Laplacian for an image f(x, y), is defined as

$$A^2 f = \frac{d^2 f}{dx^2} + \frac{d^2 f}{dy^2}$$

From the definition, the second order derivative in x-direction is

$$\frac{d^2 f}{dx^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

Similarly in y-direction is

$$\frac{d^2 f}{dy^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

Then

$$A^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

This equation can be implemented using the mask shown in the following, which gives an isotropic result for rotations in increments of 90°.The diagonal directions can be incorporated in the definition of the digital Laplacian by adding two more terms to the above equation, one for each of the two diagonal directions. The form of each new term is the same but the coordinates are along the diagonals. Since each diagonal term also contains a –2f(x, y) term, the total subtracted from the difference terms now would be –8f(x, y). The mask used to implement this new definition is shown in the figure. This mask yields isotropic results for increments of 45°. The other two masks are also used frequently in practice.

0	1	0	1	1	1
1	−4	1	1	−8	1
0	1	0	1	1	1
0	−1	0	−1	−1	−1
−1	4	−1	−1	8	−1
0	−1	0	−1	−1	−1

a b
c d

Fig.(a) Filter mask used to implement the digital Laplacian (b) Mask used to implement an extension of this equation that includes the diagonal neighbors. (c)&(d) Two other Implementations of the Laplacian.

The Laplacian with the negative sign gives equivalent results. Because the Laplacian is a derivative operator, it highlights gray-level discontinuities in an image and deemphasizes regions with slowly varying gray levels. This will tend to produce images that have grayish edge lines and other discontinuities, all superimposed on a dark, featureless background. Background features can be "recovered" while still preserving the sharpening effect of the Laplacian operation simply by adding the original and Laplacian images. If the definition uses a negative center coefficient, then we *subtract*, rather than add, then Laplacian image is used to obtain a sharpened result. Thus, the basic way in which we use the Laplacian for image enhancement is as follows:

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is negative} \\ f(x,y) + \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is positive.} \end{cases}$$

Unsharp masking and High-Boost filtering

A process that has been used for many years in the publishing industry to sharpen images consists of subtracting a blurred version of an image from the original image. This process, called *unsharp masking*, consists of following steps

- Blur the original image.
- Subtract the blurred image from the original.
- Add the mask to the original.

Let $\bar{f}(x,y)$ denotes the blurred image, unsharp masking is expressed as

$$g_{mask}(x,y) = f(x,y) - \bar{f}(x,y)$$

Then we add a weighted portion of the mask to the original image

$$g(x,y) = f(x,y) + k * g_{mask}(x,y)$$ Where k is a weighted coefficient.

When k=1, which acts as unsharp masking

When k>1, It is referred as High-Boost Filtering.

When k<1, It de-emphasizes the contribution of the unsharp mask.

Use of First Derivatives for Enhancement–The Laplacian

The first derivatives in image processing are implemented using the magnitude of the gradient. The gradient of *f* at coordinates (x, y) is defined as the two-dimensional column *vector*

$$\nabla f \equiv \text{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The magnitude of vector Af denoted as *M(x, y)*

$$M(x, y) = mag(\text{A}f) = \sqrt{(g_x^2 + g_y^2)}$$

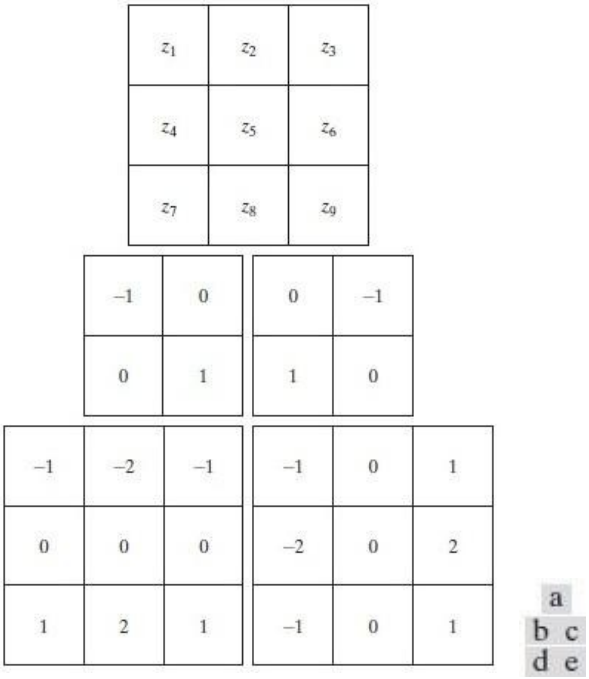
The components of the gradient vector itself are linear operators, but the magnitude of this vector obviously is not because of the squaring and square root. The computational burden of implementing the above equation over an entire image is not trivial, and it is common practice to approximate the magnitude of the gradient by using absolute values instead of squares and square roots:

$$M(x, y) \equiv |g_x| + |g_y|$$

This equation is simpler to compute and it still preserves relative changes in gray levels, but the isotropic feature property is lost in general. However, as in the case of the Laplacian, the isotropic properties of the digital gradient are preserved only for a limited number of rotational increments that depend on the masks used to approximate the derivatives. As it turns out, the most popular masks used to approximate the gradient give the same result only for vertical and horizontal edges and thus the isotropic properties of the gradient are preserved only for multiples of 90°.

Let us denote the intensities of image points in a 3×3 region shown in figure (a). For example, the center point, z_5 , denotes f(x, y), z_1 denotes f(x-1, y-1), and so on. The simplest approximations to a first-order derivative that satisfy the conditions stated are $g_x = (z_8 - z_5)$ and $g_y = (z_6 - z_5)$. Two other definitions proposed by Roberts [1965] in the early development of digital image processing use cross differences:

$$g_x = (z_9 - z_5) \text{ and } g_y = (z_8 - z_6)$$



a
b c
d e

Then we compute the gradient and its absolute values as

$$M(x, y) = \left[(z_9 - z_5)^2 + (z_8 - z_6)^2\right]^{1/2}$$

and

$$M(x, y) \approx |z_9 - z_5| + |z_8 - z_6|$$

This equation can be implemented with the two masks shown in figure (b) and (c).These masks are referred to as the *Roberts cross-gradient operators*. Masks of even size are difficult to implement. The smallest filter mask in which we are interested is of size 3×3. An approximation using absolute values, still at point z_5 , but using a 3×3 mask, is

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

These equations can be implemented using the masks shown in figure (d) and (e). The difference between the third and first rows of the 3×3 image region approximates the derivative in the *x*-direction, and the difference between the third and first columns approximates the derivative in the *y*-direction. The masks shown in figure (d) and (e) are referred as *Sobel operator.* The magnitude of gradient by using these masks is

$$M(x, y) \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

FILTERING IN THE FREQUENCY DOMAIN

2.8. Basic Concepts

Complex Numbers:

A complex number, C, is defined as

$$C == R + jI$$

Where R and I are real numbers and j is an imaginary number equal to t $j = \sqrt{-1}$. Here, R denotes the *real part* of the complex number and I its *imaginary part.* Real numbers are a subset of complex numbers in which $I = 0$. The *conjugate* of a complex number C denoted by C*, is defined as

$$C\cdot = R - jI$$

Sometimes, it is useful to represent complex numbers in polar coordinates,

$$C = |C|(\cos\theta + j\sin\theta)$$

Where $|C| = \sqrt{R^2 + I^2}$ and $\theta = \tan^{-1}(I / R)$

Fourier series:

Let a function $f(t)$ of a continuous variable t that is periodic with period, T, can be expressed as the sum of sines and cosines multiplied by appropriate coefficients. This sum, known as a *Fourier series,* has the form

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{j\frac{2\pi n}{T}t}$$

where

$$c_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t)\, e^{-j\frac{2\pi n}{T}t}\, dt \qquad \text{for } n = 0, \pm 1, \pm 2, \ldots$$

Impulses and Their Sifting Property:

A *unit impulse* of a continuous variable t located at $t = 0$, is *defined* as

$$\delta(t) = \begin{cases} 1 & \text{if } t = 0 \\ 0 & \text{if } t \neq 0 \end{cases}$$

and also to satisfy the identity

$$\int_{-\infty}^{\infty} \delta(t)\, dt = 1$$

An impulse *sifting property* with respect to integration is

$$\int_{-\infty}^{\infty} f(t)\,\delta(t)\,dt = f(0)$$

The sifting property involves an impulse located at an arbitrary point t_o, denoted by $\delta(t - t_o)$ *is defined as*

$$\int_{-\infty}^{\infty} f(t)\,\delta(t - t_0)\,dt = f(t_0)$$

Let x represent a *discrete* variable, the *unit discrete impulse,* $\delta(x)$ is defined as

$$\delta(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases}$$

and also satisfies the condition

$$\sum_{x=-\infty}^{\infty} \delta(x) = 1$$

The sifting property for discrete variables has the form

$$\sum_{x=-\infty}^{\infty} f(x)\,\delta(x) = f(0)$$

Generally a discrete impulse located at $x=x_0$

$$\sum_{x=-\infty}^{\infty} f(x)\,\delta(x - x_0) = f(x_0)$$

The Fourier Transform of Functions of One Continuous Variable:

The *Fourier transform* of a continuous function $f(t)$ of a continuous variable, t, is *defined* by the equation,

$$F(\mu) = \int_{-\infty}^{\infty} f(t)\,e^{-j2\pi\mu t}\,dt$$

and its Inverse Fourier transform

$$f(t) = \int_{-\infty}^{\infty} F(\mu)\,e^{j2\pi\mu t}\,d\mu$$

Convolution:

The convolution of two continuous functions, *f(t)* and *h(t),* of one *continuous* variable, *t,* is *defined* as

$$f(t) \star h(t) = \int_{-\infty}^{\infty} f(\tau)\, h(t-\tau)\, d\tau$$

and its Fourier Transform is

$$\Im\{f(t) \star h(t)\} = \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} f(\tau)\, h(t-\tau)\, d\tau\right] e^{-j2\pi\mu t}\, dt$$

$$= \int_{-\infty}^{\infty} f(\tau)\left[\int_{-\infty}^{\infty} h(t-\tau)\, e^{-j2\pi\mu t}\, dt\right] d\tau$$

$$\Im\{f(t) \star h(t)\} = \int_{-\infty}^{\infty} f(\tau)\left[H(\mu)\, e^{-j2\pi\mu\tau}\right] d\tau$$

$$= H(\mu)\int_{-\infty}^{\infty} f(\tau)\, e^{-j2\pi\mu\tau}\, d\tau$$

$$= H(\mu)\, F(\mu)$$

Hence the *convolution theorem* is written as

$$f(t) \star h(t) \Leftrightarrow H(\mu)\, F(\mu)$$

and

$$f(t)h(t) \Leftrightarrow H(\mu) \star F(\mu)$$

Sampling and the Fourier Transform of Sampled Functions

Sampling is the process of converting a Continuous function into a sequence of discrete values. Let us consider a continuous function, *f(t),* that we wish to sample at uniform intervals (ΔT) of the independent variable *t.* We assume that sampling is to multiply *f(t)* by a *sampling function* equal to a train of impulses Δ*T* units apart is defined as

$$\tilde{f}(t) = f(t)s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} f(t)\,\delta(t - n\Delta T)$$

Where $\hat{f}(t)$ *denotes* the sampled function and each component of this summation is an impulse weighted by the value of *f(t)* at the location of the impulse. The *value* of each sample is then given by the "strength" of the weighted impulse, which we obtain by integration. That is, the value, *f k,* of an arbitrary sample in the sequence is given by

$$f_k = \int_{-\infty}^{\infty} f(t)\,\delta(t - k\Delta T)\,dt$$

$$= f(k\Delta T)$$

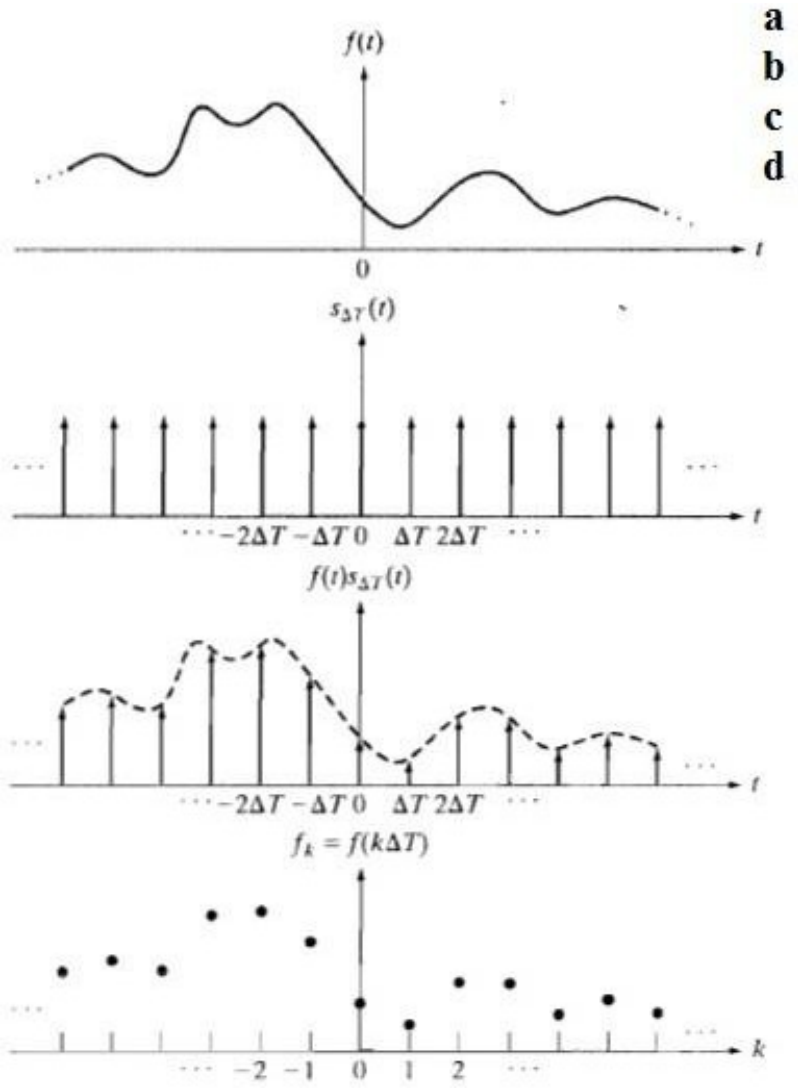


Fig: (a) A continuous function. (b) Train of impulses used to model the sampling process.(c) Sampled function formed as the product of (a) and (b). (d) Sample values obtained by integration and using the sifting prope.rty of the impulse.

The Fourier transform of sampled function is

$$\begin{aligned}\tilde{F}(\mu) &= \Im\{\tilde{f}(t)\} \\ &= \Im\{f(t)s_{\Delta T}(t)\} \\ &= F(\mu) \star S(\mu)\end{aligned}$$

Where

$$S(\mu) = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\delta\left(\mu - \frac{n}{\Delta T}\right)$$

Hence the convolution is

$$\begin{aligned}\tilde{F}(\mu) &= F(\mu) \star S(\mu) \\ &= \int_{-\infty}^{\infty} F(\tau)S(\mu - \tau)\,d\tau \\ &= \frac{1}{\Delta T}\int_{-\infty}^{\infty} F(\tau)\sum_{n=-\infty}^{\infty}\delta\left(\mu - \tau - \frac{n}{\Delta T}\right)d\tau \\ &= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty} F(\tau)\delta\left(\mu - \tau - \frac{n}{\Delta T}\right)d\tau \\ &= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} F\left(\mu - \frac{n}{\Delta T}\right)\end{aligned}$$

The summation in the last line shows that the Fourier transform of the sampled function is an *infinite, periodic* sequence of *copies* of the transform of the original, continuous function. The separation between copies is determined by the value of $1/\Delta T$.

The quantity $1/\Delta T$, is the sampling rate used to generate the sampled function. The sampling rate was high enough to provide sufficient separation between the periods and thus preserve the integrity of $F(\mu)$ is known as *over-sampling.* If the sampling rate was just enough to preserve $F(\mu)$ is known as critically-sampling. The sampling rate was below the minimum required to maintain distinct copies of $F(\mu)$ and thus failed to preserve the original transform is known as *under-sampling.*

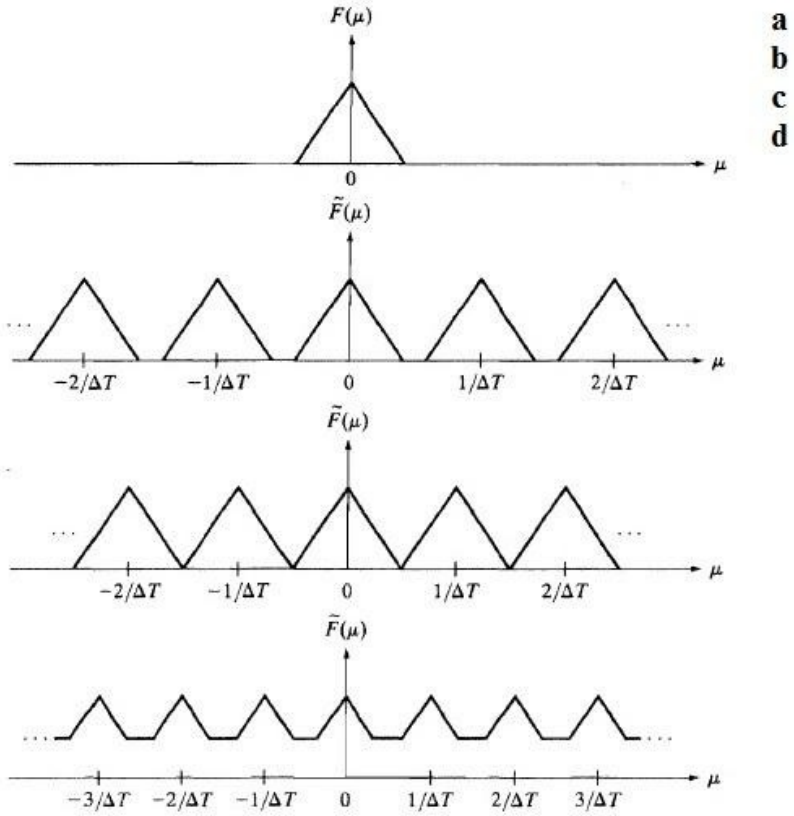


Fig. (a) Fourier transform of a band-limited function. (b)-(d) Transforms of the corresponding sampled function under the conditions of over-sampling, critically sampling, and under-sampling, respectively.

Sampling Theorem:

A function *f(t)* whose Fourier transform is zero for values of frequencies outside a finite interval $[-\mu_{max}, \mu_{max}]$ about the origin is called a *band-limited* function. We can recover *f(t)* from its sampled version- if we can isolate a copy of $F(\mu)$ from the periodic sequence of copies of this function contained in $\hat{F}(\mu)$. $\hat{F}(\mu)$ is a *continuous, periodic* function with period $1/\Delta T$. This implies that we can recover *f(t)* from that single period by using the inverse Fourier transform. Extracting from $\hat{F}(\mu)$ a single period that is equal to $F(\mu)$ is possible if the separation between copies is sufficient with separation period

$$\frac{1}{\Delta T} > 2\mu_{max}$$

This equation indicates that a continuous, band-limited function can be recovered completely from a set of its samples if the samples are acquired at a rate exceeding twice the highest frequency content of the function. This result is known as the *sampling theorem.*

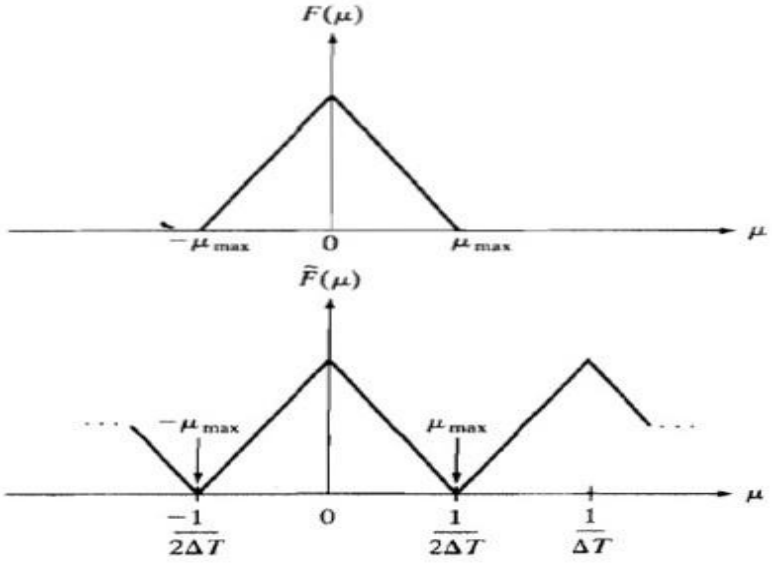


Fig. (a) Transform of a band-limited function. (b) Transform resulting from critically sampling the same function.

2.9. Extension to Functions of Two Variables

The 2-D Impulse and Its Sifting Property:

The impulse, $\delta(t, z)$, of two continuous variables, t and z, is defined as in

$$\delta(t, z) = \begin{cases} \infty & \text{if } t = z = 0 \\ 0 & \text{otherwise} \end{cases}$$

and

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(t, z)\, dt\, dz = 1$$

The 2-D impulse exhibits the *sifting property* under integration,

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t, z)\,\delta(t, z)\, dt\, dz = f(0, 0)$$

or, more generally for an impulse located at coordinates (t_0, z_0),

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t, z)\,\delta(t - t_0, z - z_0)\, dt\, dz = f(t_0, z_0)$$

For discrete variables x and y, the 2-D discrete impulse is defined as

$$\delta(x, y) = \begin{cases} 1 & \text{if } x = y = 0 \\ 0 & \text{otherwise} \end{cases}$$

and its sifting property is

$$\sum_{x=-\infty}^{\infty}\sum_{y=-\infty}^{\infty} f(x, y)\,\delta(x, y) = f(0, 0)$$

Where *f(x, y)* is a function of discrete variables *x* and *y*. For an impulse located at coordinates $(x_0, y0)$ the sifting property is

$$\sum_{x=-\infty}^{\infty}\sum_{y=-\infty}^{\infty} f(x, y)\delta(x - x_0, y - y_0) = f(x_0, y_0)$$

The 2-D Continuous Fourier Transform Pair

Let *f (t, z)* be a continuous function of two continuous variables, *t* and *z*. The two-dimensional, continuous Fourier transform pair is given by the expressions

$$F(\mu, \nu) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t, z)e^{-j2\pi(\mu t + \nu z)}dt\,dz$$

and

$$f(t, z) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(\mu, \nu)e^{j2\pi(\mu t + \nu z)}d\mu\,d\nu$$

Two-Dimensional Sampling and the 2-D Sampling Theorem:

Sampling in two dimensions can be modeled using the sampling function

$$s_{\Delta T\Delta Z}(t, z) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\delta(t - m\Delta T, z - n\Delta Z)$$

Where ΔT and ΔZ are the separations between samples along the *t*- and z-axis of the continuous function *f (t, z)*. Function *f(t, z)* is said to be band-limited if its Fourier Transform is zero outside a rectangle established by the intervals $[-\mu_{max}, \mu_{max}]$ *and* $[-\nu_{max}, \nu_{max}]$ that is,

$$F(\mu, \nu) = 0 \quad \text{for } |\mu| \geq \mu_{\max} \text{ and } |\nu| \geq \nu_{\max}$$

The *two-dimensional sampling theorem* states that a continuous, band-limited function *f(t, z)* can be recovered with no error from a set of its samples if the sampling intervals are,

$$\frac{1}{\Delta T} > 2\mu_{\max}$$

and

$$\frac{1}{\Delta Z} > 2\nu_{\max}$$

The 2-D Discrete Fourier Transform and Its Inverse

The discrete Fourier transform and its inverse Fourier transform of an image *f(x, y)* of size M×N is defined as

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

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

The Fourier spectrum, Phase angle and power spectrum as

$$|F(u,v)| = \left[R^2(u,v) + I^2(u,v)\right]^{1/2}$$

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

and

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

Where *R (u, v) and I (u,v)* are the real and imaginary parts of *F(u,v)*. Some properties of Fourier transform are listed below,

Name	DFT Pairs
1) Symmetry properties	See Table 4.1
2) Linearity	$af_1(x,y) + bf_2(x,y) \Leftrightarrow aF_1(u,v) + bF_2(u,v)$
3) Translation (general)	$f(x,y)e^{j2\pi(u_0x/M+v_0y/N)} \Leftrightarrow F(u-u_0, v-v_0)$ $f(x-x_0, y-y_0) \Leftrightarrow F(u,v)e^{-j2\pi(ux_0/M+vy_0/N)}$
4) Translation to center of the frequency rectangle, (*M*/2, *N*/2)	$f(x,y)(-1)^{x+y} \Leftrightarrow F(u-M/2, v-N/2)$ $f(x-M/2, y-N/2) \Leftrightarrow F(u,v)(-1)^{u+v}$
5) Rotation	$f(r, \theta+\theta_0) \Leftrightarrow F(\omega, \varphi+\theta_0)$ $x = r\cos\theta \quad y = r\sin\theta \quad u = \omega\cos\varphi \quad v = \omega\sin\varphi$
6) Convolution theorem†	$f(x,y) \star h(x,y) \Leftrightarrow F(u,v)H(u,v)$ $f(x,y)h(x,y) \Leftrightarrow F(u,v) \star H(u,v)$

	Name	DFT Pairs
7)	Correlation theorem†	$f(x, y) \star h(x, y) \Leftrightarrow F^*(u, v)H(u, v)$ $f^*(x, y)h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$
8)	Discrete unit impulse	$\delta(x, y) \Leftrightarrow 1$
9)	Rectangle	$\text{rect}[a, b] \Leftrightarrow ab\frac{\sin(\pi ua)}{(\pi ua)}\frac{\sin(\pi vb)}{(\pi vb)}e^{-j\pi(ua+vb)}$
10)	Sine	$\sin(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $j\frac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) - \delta(u - Mu_0, v - Nv_0)\right]$
11)	Cosine	$\cos(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $\frac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) + \delta(u - Mu_0, v - Nv_0)\right]$
	The following Fourier transform pairs are derivable only for continuous variables, denoted as before by t and z for spatial variables and by μ and ν for frequency variables. These results can be used for DFT work by sampling the continuous forms.	
12)	*Differentiation* (The expressions on the right assume that $f(\pm\infty, \pm\infty) = 0$.)	$\left(\frac{\partial}{\partial t}\right)^m\left(\frac{\partial}{\partial z}\right)^n f(t, z) \Leftrightarrow (j2\pi\mu)^m(j2\pi\nu)^n F(\mu, \nu)$ $\frac{\partial^m f(t, z)}{\partial t^m} \Leftrightarrow (j2\pi\mu)^m F(\mu, \nu); \frac{\partial^n f(t, z)}{\partial z^n} \Leftrightarrow (j2\pi\nu)^n F(\mu, \nu)$
13)	*Gaussian*	$A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)} \Leftrightarrow Ae^{-(\mu^2+\nu^2)/2\sigma^2}$ (A is a constant)

2.10. The Basics of Filtering in the Frequency Domain

Filtering in the frequency domain are based on modifying the Fourier Transform of an image and then computing the inverse transform to obtain the processed result.

$$g(x,y) = F^{-1}[H(u,v)F(u,v)]$$

Fundamental steps involved in the Frequency Domain are:

1. Given an input image *f(x,y)* of size *M×N*, obtain padding parameters *P* and *Q*. Typically, *P=2M* and *Q=2N*.
2. Form a padded image $f_p(x,y)$ of size *P×Q* by appending the necessary number of zeros to f(x,y).
3. Multiply $f_p(x,y)$ by $(-1)^{x+y}$ to centre its transform.
4. Compute the DFT, *F(u,v),* of the image from step 3.
5. Generate a real, symmetric filter function, *H(u,v),* of size *P×Q* with centre at coordinates *(P/2,Q/2)*. Form the product *G(u,v)=H(u,v)F(u,v)* using array multiplication.

6. Obtain the processed image:

$$g_p(x,y) = \text{real}\left[IDFT\left[G(u,v)\right]\right](-1)^{x+y}$$

7. Obtain the final processed result, *g(x,y),* by extracting the *M*×*N* region from the top, left quadrant of $g_p(x,y)$.

2.11. Image Smoothing using Frequency Domain Filters:

Smoothing is achieved in the frequency domain filtering by attenuating a specified range of high frequency components in the transform of a given image.

Ideal Low pass Filter

The ideal low pass filter that passes without attenuation all frequencies within a circle of radius *D0* from the origin and "cuts off" all frequencies outside this circle. The 2-D low pass filter

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

Where *D(u,v)* is the distance between a point *(u,v)* and the centre of the frequency rectangle:

$$D(u,v) = \left[(u - P/2)^2 + (v - Q/2)^2\right]^{1/2}$$

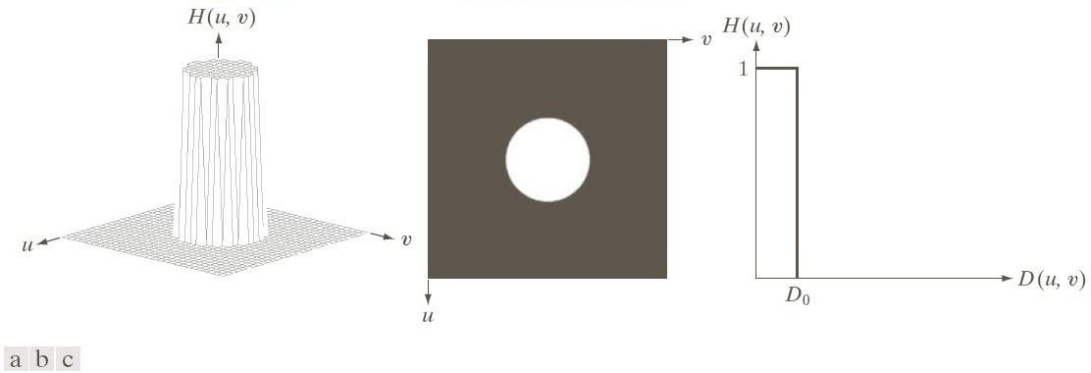


Fig: 3.3 (a) Perspective plot of an ideal Low pass Filter transfer function. (b) Filter displayed as an image. (c) Filter Radial cross section.

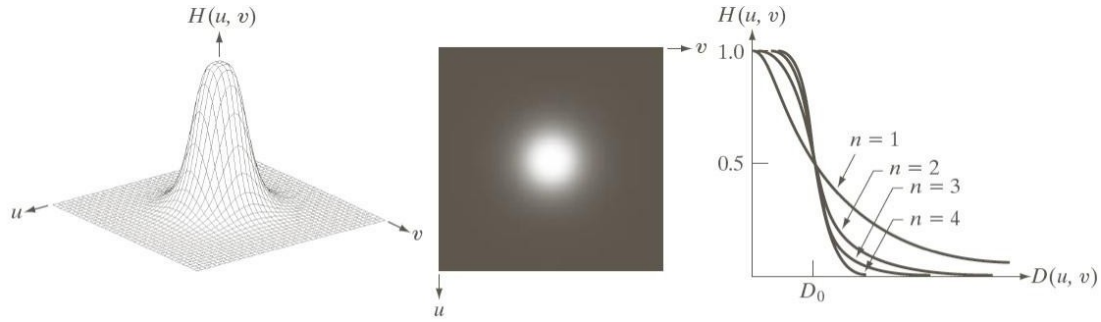
The point of transition between *H(u,v) = 1* and *H(u,v) = 0* is called the *cutoff frequency.* The sharp cutoff frequencies of an ILPF cannot be realized with electronic components and it produces ringing effect where a series of lines decreasing intensity lie parallel to the edges. To avoid this ringing effect Gaussian low-pass or Butterworth low-pass filters are preferred.

Butterworth Low pass Filter:

The transfer functions of a Butterworth low-pass filter (BLPF) of order *n* and with cutoff frequency at a distance D_0 from the origin:

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

If n increases, the filter becomes sharper with increased ringing in the spatial domain. For n=1 it produces no ringing effect and n=2 ringing is present but imperceptible.



a b c

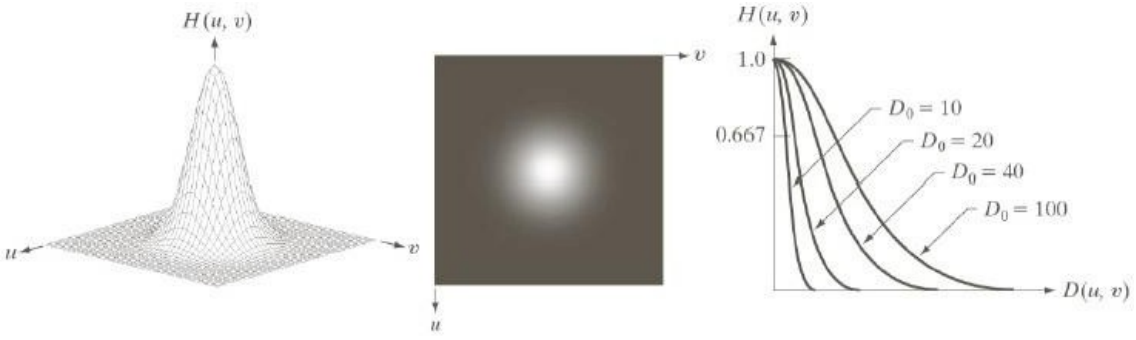
Fig: 3.4 (a) Perspective plot of a Butterworth Low pass Filter transfer function. (b) Filter displayed as an image. (c) Filter Radial cross section of orders through n=1 to 4.

Gaussian Low pass Filter:

The transfer function of a 2D Gaussian low-pass filter (GLPF) is defined as

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

The Gaussian LPF transfer function is controlled by the value of cut-off frequency D_0. The advantage of the Gaussian filter is that it never causes ringing effect.



a b c

Fig. (a) Perspective plot of a Butterworth Low pass Filter transfer function. (b) Filter displayed as an image. (c) Filter Radial cross section for various values of D_0.

2.12. Image Sharpening using Frequency Domain Filters:

Image sharpening can be achieved in the frequency domain by High-pass filtering which attenuates the low frequency components without disturbing high-frequency components of the Fourier transform of the image. A high-pass H_{HP} filter can be obtained from a given low-pass H_{LP} filter by

$$H_{HP}(u, v) = 1 - H_{LP}(u, v)$$

Ideal High pass Filter

A 2-D *Ideal High pass Filter* (IHPF) is defined as:

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

The IHPF is the opposite of the ILPF in the sense that it sets to zero all frequencies inside a circle of radius D_o while passing, without attenuation all frequencies ouside the circle.

Butterworth High pass Filter

A 2-D Butterworth High pass Filter of order n and cutoff frequency D_o is defined as

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

The order n determines the sharpness of the cutoff value and the amount of ringing. The transition into higher values of cutoff frequencies is much smoother with the BHPF.

Gaussian High pass Filter

The transfer function of the Gaussian high pass filter with cutoff frequency locus at a distance D_o from the center of the frequency rectangle is given by

$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$$

The results obtained by using GHPF are more gradual than with the IHPF, BHPF filters. Even the filtering of the smaller objects and thin bars is cleaner with the Gaussian filter.

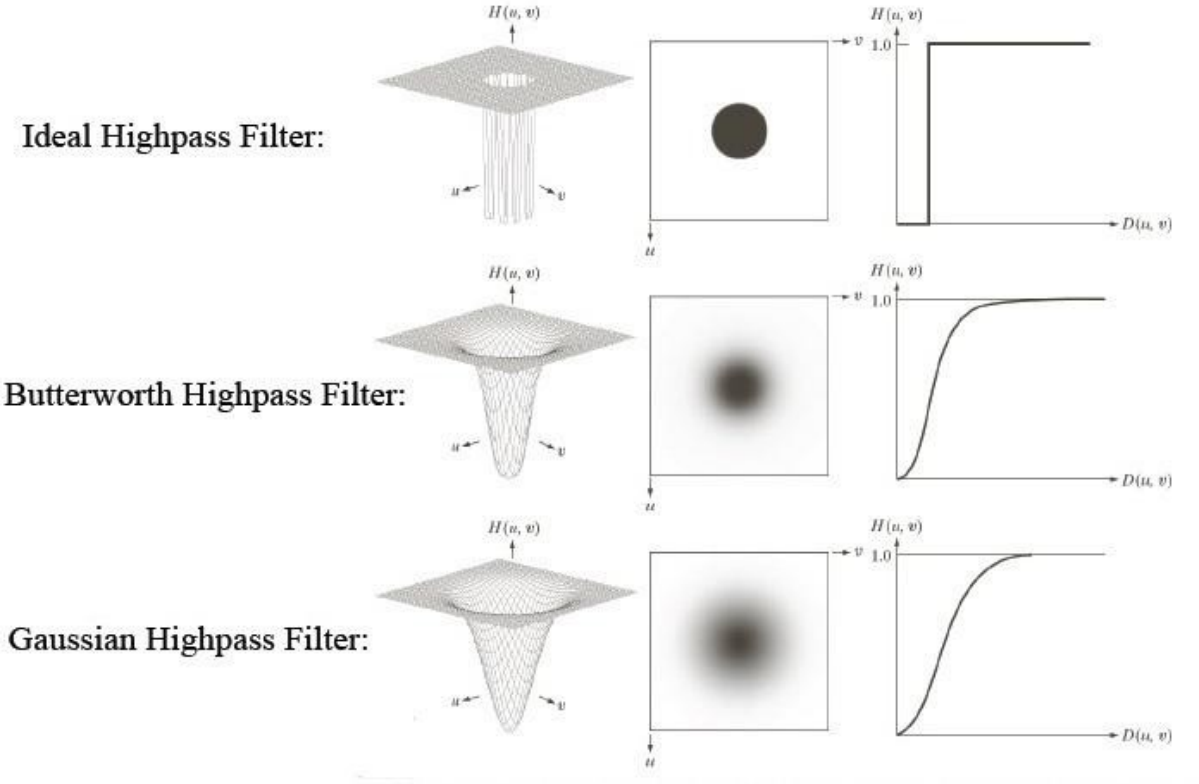


Fig. Perspective plot, Filter displayed as an image and Filter Radial cross section.

2.13. The Laplacian in the Frequency Domain

The Laplacian for an image f(x, y) is defined as

$$A^2 f(x,y) = \frac{d^2 f}{dx^2} + \frac{d^2 f}{dy^2}$$

We know that,

$$\Im\left[\frac{d^n f(x)}{dx^n}\right] = (ju)^n F(u).$$

The Fourier Transform of Laplacian for an image f(x, y) is

$$\Im\left[\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}\right] = (ju)^2 F(u,v) + (jv)^2 F(u,v)$$
$$= -(u^2 + v^2)F(u,v).$$

Then

$$\Im[\nabla^2 f(x,y)] = -(u^2 + v^2)F(u,v),$$

Hence the Laplacian can be implemented in the frequency domain by using the filter

$$H(u,v) = -(u^2 + v^2).$$

In all filtering operations, the assumption is that the origin of F (u, v) has been centered by performing the operation f(x, y) $(-1)^{x+y}$ prior to taking the transform of the image. If f (and F) are of size M X N, this operation shifts the center transform so that (u, v) = (0, 0) is at point (M/2, N/2) in the frequency rectangle. As before, the center of the filter function also needs to be shifted:

$$H(u, v) = -\left[(u - M/2)^2 + (v - N/2)^2\right].$$

The Laplacian-filtered image in the spatial domain is obtained by computing the inverse Fourier transform of H (u, v) F (u, v):

$$\nabla^2 f(x, y) = \Im^{-1}\left\{-\left[(u - M/2)^2 + (v - N/2)^2\right]F(u, v)\right\}.$$

Conversely, computing the Laplacian in the spatial domain and computing the Fourier transform of the result is equivalent to multiplying F(u, v) by H(u, v). Hence this dual relationship in the Fourier-transform-pair notation,

$$\nabla^2 f(x, y) \Leftrightarrow -\left[(u - M/2)^2 + (v - N/2)^2\right]F(u, v).$$

The Enhanced image g(x, y) can be obtained by subtracting the Laplacian from the original image,

$$g(x, y) = f(x, y) - \nabla^2 f(x, y).$$

2.14 The Unsharp Masking, High-Boost Filtering & High-Frequency Emphasis Filtering

Using frequency domain, the filter mask can be defined as

$$g_{mask}(x, y) = f(x, y) - f_{LP}(x, y)$$

With

$$f_{LP}(x, y) = F^{-1}[H_{LP}(u,v)F(u,v)]$$

Where H_{LP} is a low pass filter, $F(u,v)$ is the Fourier transform of $f(x,y)$ and $f_{LP}(x,y)$ is a smoothed image. Then

$$g(x, y) = f(x,y) + k * g_{mask}(x,y)$$

This expression defines unsharp masking when k=1 and high boost filtering when k>1. Using frequency domain the $g(x, y)$ can be expressed as

$$g(x, y) = F^{-1}\{[1 + k*[1 - H_{LP}(u,v)]]F(u,v)\}$$

$$g(x, y) = F^{-1}\{[1 + k*H_{HP}(u,v)]F(u,v)\}$$

The expression contained within the square brackets is called "High-Frequency Emphasis filter". The High pass filter set the dc term to zero, thus reducing the average intensity in the filtered image to 0. The high-frequency emphasis filter does not have this problem because of the 1 that is added to the high pass filter. The constant k gives control over the proportion of high frequencies that influence the final result. Now, the high frequency-emphasis filtering is expressed as,

$$g(x, y) = F^{-1}\{[k_1 + k_2 * H_{HP}(u,v)]F(u,v)\}$$

Where $k_1 \geq 0$ gives controls of the offset from the origin and $k_2 \geq 0$ controls the contribution of high frequencies.

2.15. Homomorphic filtering

The illumination-reflectance model can be used to develop a frequency domain procedure for improving the appearance of an image by simultaneous gray-level range compression and contrast enhancement. An image f(x, y) can be expressed as the product of illumination and reflectance components,

$$f(x, y) = i(x, y)r(x, y).$$

This equation cannot be used directly to operate separately on the frequency components of illumination and reflectance because the Fourier transform of the product of two functions is not separable; in other words,

$$\Im\{f(x, y)\} \neq \Im\{i(x, y)\}\Im\{r(x, y)\}.$$

Suppose, however, that we define

$$\begin{aligned} z(x, y) &= \ln f(x, y) \\ &= \ln i(x, y) + \ln r(x, y). \end{aligned}$$

Then

$$\begin{aligned} \Im\{z(x, y)\} &= \Im\{\ln f(x, y)\} \\ &= \Im\{\ln i(x, y)\} + \Im\{\ln r(x, y)\} \end{aligned}$$

or

$$Z(u, v) = F_i(u, v) + F_r(u, v)$$

Where $F_i(u, v)$ and $F_r(u, v)$ are the Fourier transforms of $\ln i(x, y)$ and $\ln r(x, y)$, respectively. If we process Z (u, v) by means of a filter function H (u, v) then, from

$$\begin{aligned} S(u, v) &= H(u, v)Z(u, v) \\ &= H(u, v)F_i(u, v) + H(u, v)F_r(u, v) \end{aligned}$$

Where the filtered image in the spatial domain is

$$s(x, y) = \Im^{-1}\{S(u, v)\}$$
$$= \Im^{-1}\{H(u, v)F_i(u, v)\} + \Im^{-1}\{H(u, v)F_r(u, v)\}.$$

By letting

$$i'(x, y) = \Im^{-1}\{H(u, v)F_i(u, v)\}$$

and

$$r'(x, y) = \Im^{-1}\{H(u, v)F_r(u, v)\},$$

Now we can express

$$s(x, y) = i'(x, y) + r'(x, y).$$

Finally, z (x, y) was formed by taking the logarithm of the original image f (x, y), and the inverse operation yields the desired enhanced image, denoted by g(x, y).

$$g(x, y) = e^{s(x,y)}$$
$$= e^{i'(x,y)} \cdot e^{r'(x,y)}$$
$$= i_0(x, y)r_0(x, y)$$

Where

$$r_0(x, y) = e^{r'(x,y)}$$

$$i_0(x, y) = e^{i'(x,y)}$$

are the illumination and reflectance components of the output image. The filtering appraoch issummarized as shown in the figure.

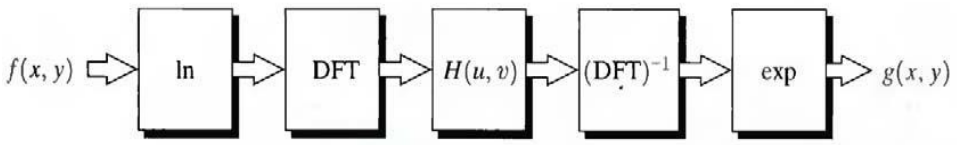


Fig: Summary of steps in Homomorphic Filtering

This enhancement approach is based on a special case of a class of systems known as homomorphic systems. In this particular application, the key to the approach is the separation of the illumination and reflectance components achieved. The homomorphic filter function H (u, v) can then operate on these components separately. The illumination component of an image generally is characterized by slow spatial variations, while the reflectance component tends to vary abruptly, particularly at the junctions of dissimilar objects.

2.16. Selective filtering

The filters Low-Pass and High-Pass are operate over the entire frequency rectangle. There are some other applications in which it is of interest to process specific band of frequencies selective filters are used. They are,

- Band Pass Filter
- Band Reject Filter
- Notch Filters

Band Reject and Band Pass Filter

The Band Reject Filter transfer function is defined as

Ideal	Butterworth	Gaussian
$H(u,v) = \begin{cases} 0 & \text{if } D_0 - \frac{W}{2} \le D \le D_0 + \frac{W}{2} \\ 1 & \text{otherwise} \end{cases}$	$H(u,v) = \frac{1}{1 + \left[\frac{DW}{D^2 - D_0^2}\right]^{2n}}$	$H(u,v) = 1 - e^{-\left[\frac{D^2 - D_0^2}{DW}\right]^2}$

Where „W" is the width of the band, D is the distance *D (u, v)* from the centre of the filter, D_o is the cutoff frequency and n is the order of the Butterworth filter. The band reject filters are very effective in removing periodic noise and the ringing effect normally small

A Band pass filter is obtained from the band reject filter as

$$H_{BP}(u,v) = 1 - H_{BR}(u,v)$$

Band Reject Filter Band Pass Filter

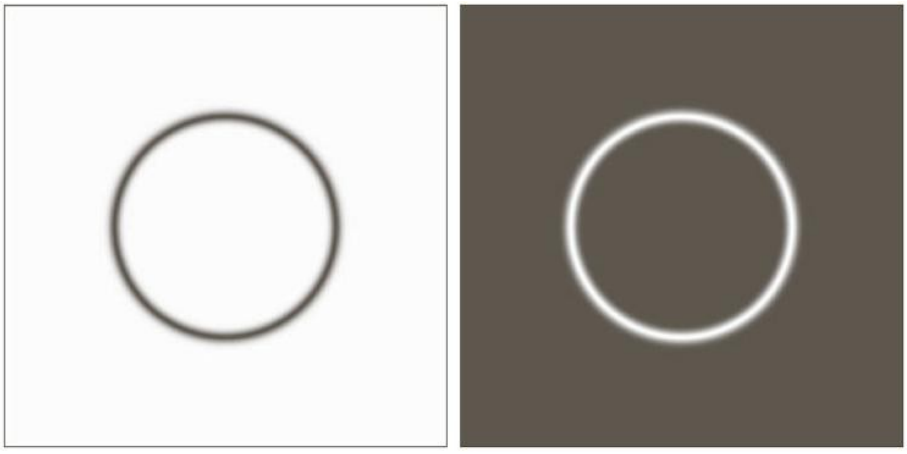


Fig: Band Reject Filter and its corresponding Band Pass Filter

Notch Filters

A Notch filter Reject (or pass) frequencies in a predefined neighborhood about the centre of the frequency rectangle. It is constructed as products of high pass filters whose centers have been translated to the centers of the notches. The general form is defined as

$$H_{NR}(u,v) = \prod_{k=1}^{Q} H_k(u,v) H_{-k}(u,v)$$

centre at (u_k, v_k) centre at $(-u_k, -v_k)$

Where $H_k(u,v)$ and $H_{-k}(u,v)$ are high pass filters whose centers are at (u_k, v_k) and ($-u_k$, $-v_k$) respectively. These centers are specified with respect to the center of the frequency rectangle (M/2, N/2). The distance computations for each filter are defined as

$$D_k(u,v) = \left[(u - M/2 - u_k)^2 + (v - N/2 - v_k)^2\right]^{1/2}$$

$$D_{-k}(u,v) = \left[(u - M/2 + u_k)^2 + (v - N/2 + v_k)^2\right]^{1/2}$$

A *Notch Pass filter* (NP) is obtained from a *Notch Reject filter* (NR) using:

$$H_{NP}(u,v) = 1 - H_{NR}(u,v)$$

For example the Butterworth notch reject filter of order n, containing three notch pairs is defined as

$$H_{NR}(u,v) = \prod_{k=1}^{3} \left[\frac{1}{1 + [D_{0k}/D_k(u,v)]^{2n}}\right] \left[\frac{1}{1 + [D_{0k}/D_{-k}(u,v)]^{2n}}\right]$$

PREVIOUS QUESTIONS

1. What is meant by image enhancement? Explain the various approaches used in image enhancement.
2. a) Explain the Gray level transformation. Give the applications.
   b) Compare frequency domain methods and spatial domain methods used in image enhancement
3. Explain in detail about histogram processing.
4. What is meant by Histogram Equalization? Explain.
5. Explain how Fourier transforms are useful in digital image processing?
6. What is meant by Enhancement by point processing? Explain.

7. Discuss in detail about the procedure involved in Histogram matching.
8. Prove that for continuous signal Histogram equalization results in flat histogram
9. Differentiate the spatial image enhancement and image enhancement in frequency domain?
10. Explain in detail image averaging and image subtraction.
11. What is meant by Histogram of an image? Sketch Histograms of basic image types.
12. Explain about image smoothing using spatial filters
13. Explain the following concepts: Local enhancement.
14. Explain spatial filtering in image enhancement.
15. Explain about fuzzy techniques for intensity transformations.
16. Explain about the basic of filtering in the frequency domain.
17. Explain image smoothing using frequency domain filters.
18. Explain about the discrete Fourier transform (DFT) of one variable and two variables.
19. Explain about selective filtering
20. What is homomorphic filter? How to implement it?
21. Discuss about the properties of 2-D Discrete Fourier transform.
22. Explain the process of sampling in two dimensional functions.
23. Explain the process of two dimensional convolution?