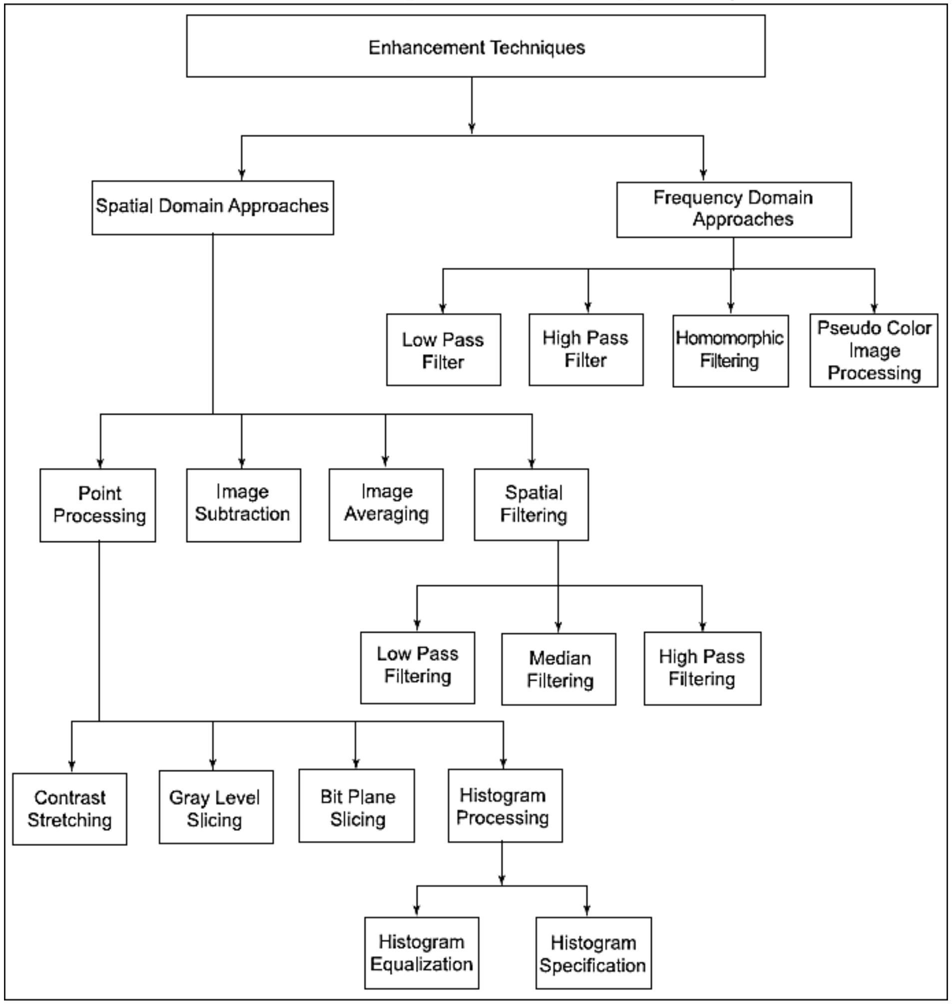
Chapter 9 **Frequency Approach and Transform Domain**

SPATIAL DOMAIN AND FREQUENCY DOMAIN APPROACHES:

In the spatial domain method, the pixel composing of image details are considered and the various procedures are directly applied on these pixels. The image processing functions in the spatial domain may be expressed as: $g(x,y) = T[f(x,y)]$

Where f(x, y) is the input image, g(x, y) is the processed output image and T represents an operation on 'f' defined over some neighborhood of (x, y). Sometimes T can also be used to operate on a set of input images.



The spatial technique methods are simple and easy to implement and also the speed of operation is high. In spite of these advantages, there are certain situations in which the spatial domain filters are not easily addressable. Under such circumstances it is more appealing and intuitive to use the frequency domain filtering approach. All the frequency domain filters are based on computing the Fourier transform of an image to be enhanced. Then the result is multiplied with a filter transfer function and the inverse Fourier transform is applied to the product so that it results in the enhanced image.

The low-pass filter is used to smooth the image and remove the high-frequency components related to noise. Smoothing effect is achieved in the frequency domain by attenuating a specified range of high-frequency components in the transformed image.

There are mainly three steps in frequency domain approach:

1. Transform the image to its frequency representation
2. Perform image processing
3. Compute inverse transform back to the spatial domain

Frequencies in an Image:

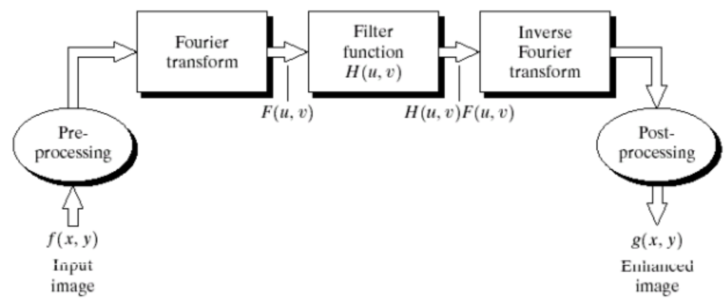
Any spatial or temporal signal has an equivalent frequency representation

What do frequencies mean in an image?

- High frequencies correspond to pixel values that change rapidly across the image (e.g. text, texture, leaves, etc.)
- Strong low frequency components correspond to large scale features in the image (e.g. a single, homogenous object that dominates the image)

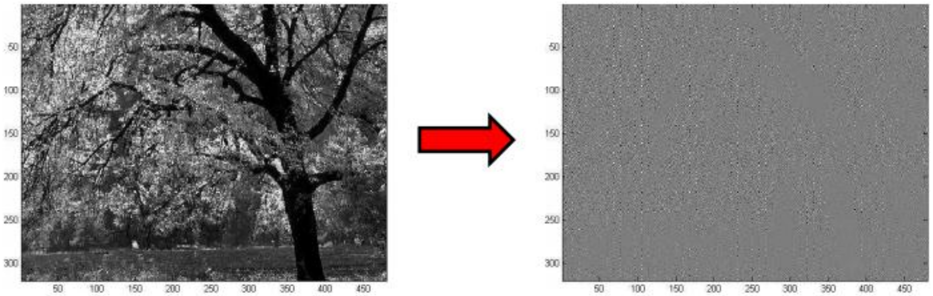
Procedure for Filtering in the Frequency Domain:

Basic steps :

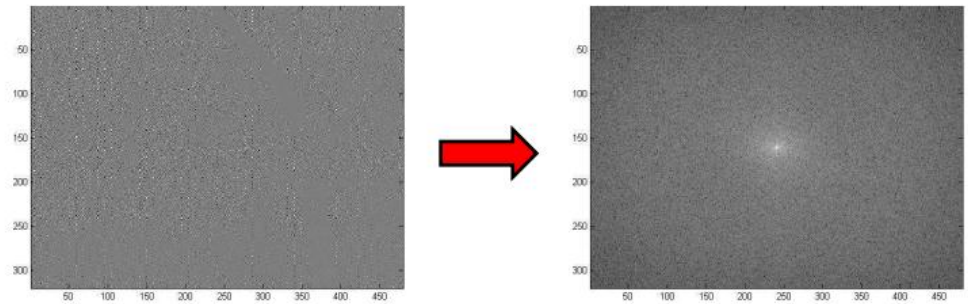


1. Multiply image *f(x,y)* by coefficients $(-1)^{x+y}$ to center the spectrum
2. Compute discret two-dimensional Fourier transform *F(u,v)*,
3. Multiply spectrum *F(u,v)* by filter function (transfer function) *H(u,v)*
4. Compute **inverse** discret two-dimensional Fourier transform $F^{-1}(u,v)$,
5. Extract real part of the result
6. Multiply the result by coefficients $(-1)^{x+y}$ to obtain the final result

1. Multiply the input image by $(-1)^{x+y}$ to center the transform



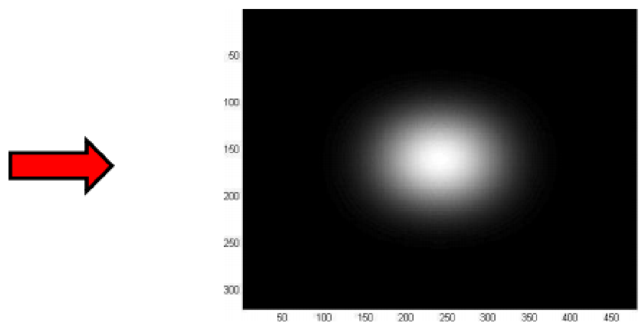
2. Compute the DFT F(u,v) of the resulting image



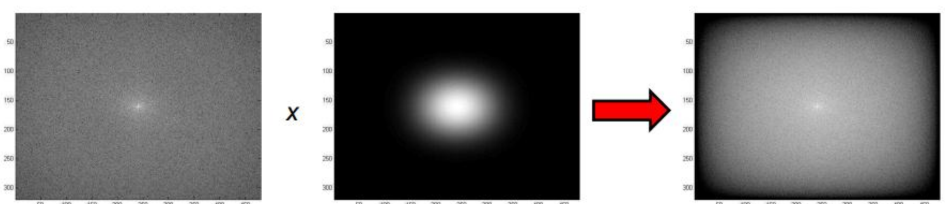
3. Multiply F(u,v) by a filter G(u,v)

$$\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} / 256$$

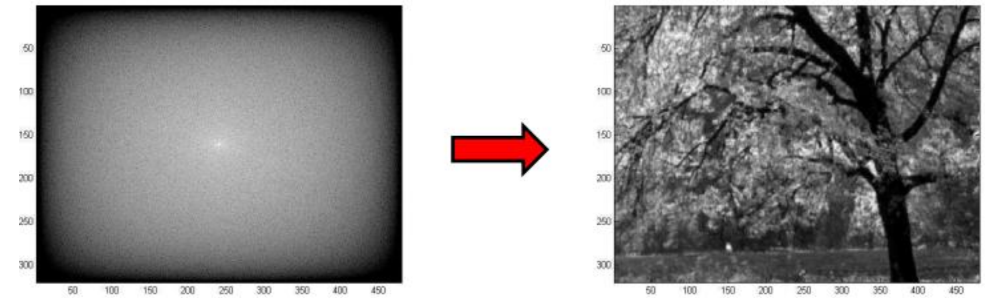
g(x,y)

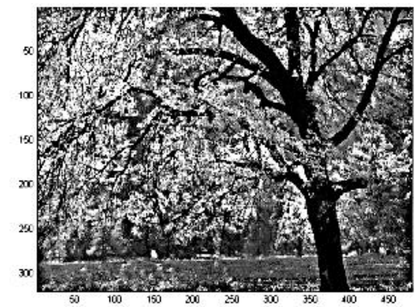


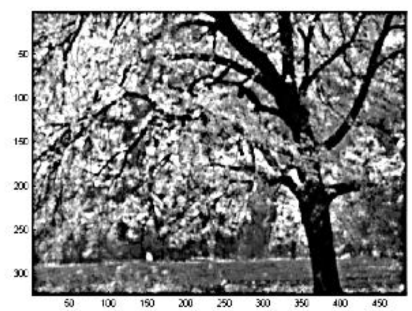
G(u,v)



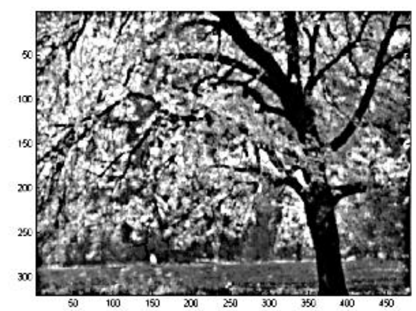
4. Computer the inverse DFT transform h*(x,y)
5. Obtain the real part h(x,y) of 4
6. Multiply the result by $(-1)^{x+y}$







$f(x,y) * g(x,y)$



$F(u,v)G(u,v)$

Low pass filters: Low-pass filtering smooth a signal or image

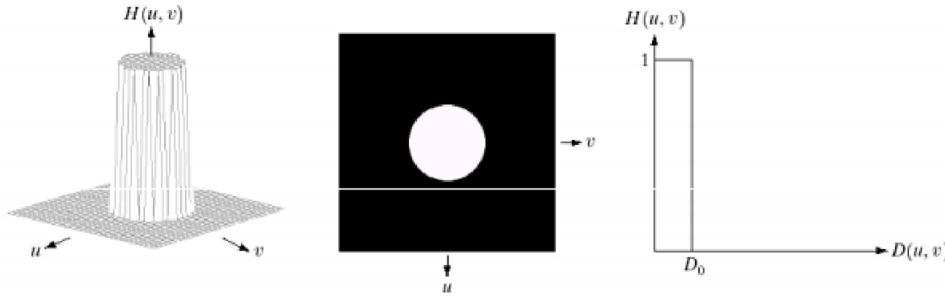
Ideal low pass filters:

A 2-D lowpass filter that passes without attenuation all frequencies within a circle of radius D0 from the origin and "cuts off" all frequencies outside this circle is called and ideal lowpass filter; it is specified by the function:

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

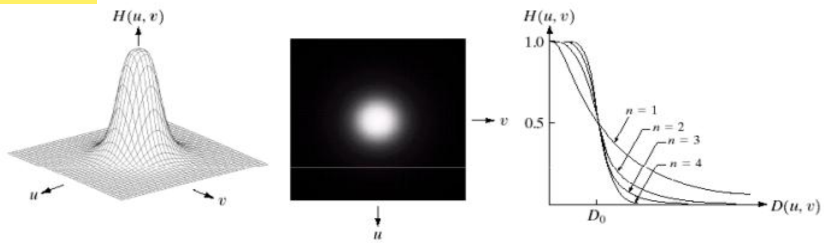
Where D_0 is a positive constant and $D(u,v)$ is the distance between a point (u,v) in the frequency domain and the center of the frequency rectangle; that is: $D(u,v) = [(u-\frac{P}{2})^2 + (v-\frac{Q}{2})^2]^{1/2}$

Where P and Q are padded sizes.



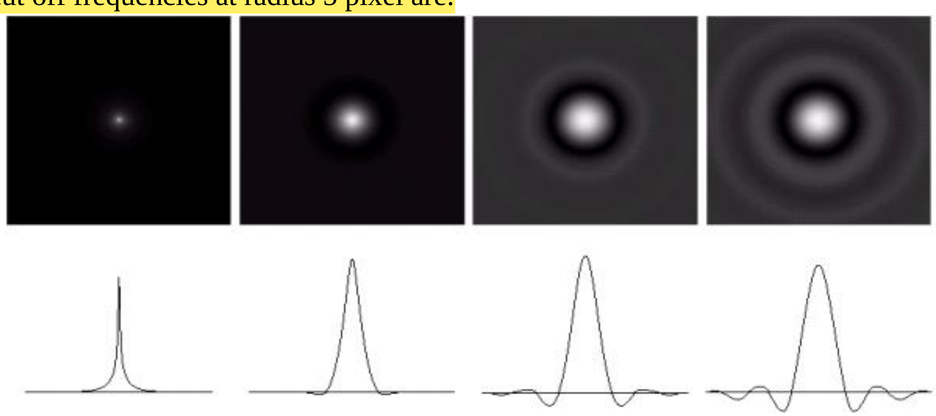
Butterworth Lowpass Filters:

The transfer function of a Butterworth lowpass filter (BLPF) of order n, and with cutoff frequency at a distance D_0 from the origin, is defined as: $H(u,v) = \frac{1}{1+[D(u,v)/D_0]^{2n}}$



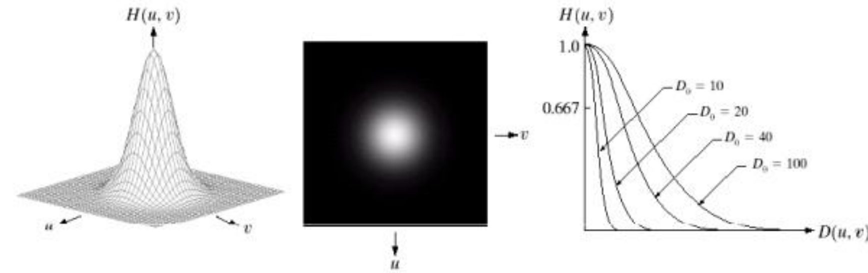
$$H(u,v) = \frac{1}{1+(D(u,v)/D_0)^{2n}} \qquad D(u,v) = \sqrt{(u-M/2)^2 + (v-N/2)^2}$$

Spatial representation of Butterworth low pass filter of order 1, 2, 5 and 20 with cut off frequencies at radius 5 pixel are:



Gaussian Lowpass filter:

The Gaussian lowpass filter in two dimensions is given by: $H(u,v) = e^{-D^2(u,v)/2\sigma^2}$ where $D(u,v)$ is the distance from the center of the frequency rectangle. σ is the measure of spread about the center.



$$H(u,v) = e^{-D^2(u,v)/2D_0^2} \qquad D_0 \leftrightarrow \sigma \qquad D(u,v) = \sqrt{(u-M/2)^2 + (v-N/2)^2}$$

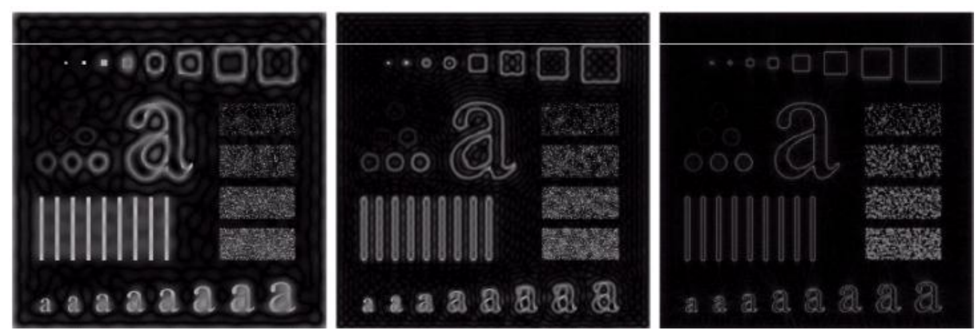
For $D(u,v)=D_0$ transfer function reach the values 0.607

Ideal Highpass filters:

A 2-D ideal Highpass filter (IHPF) is defined as: $H(u,v) = \begin{cases} 0 & if\ D(u,v) \leq D_0 \\ 1 & if\ D(u,v) > D_0 \end{cases}$

where D_0 is the cutoff frequency and $D(u,v)$ is the distance between a point (u,v).

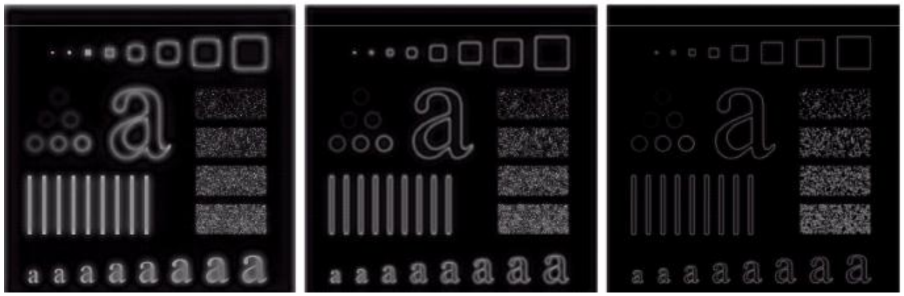
Results of ideal highpass filter with D0 = 15 and 30



Butterworth Highpass Filters:

The transfer function of a Butterworth Highpass filter (BHPF) of order n, and with cutoff frequency at a distance D_0 is defined as: $H(u,v) = \frac{1}{1+[D_0/D(u,v)]^{2n}}$

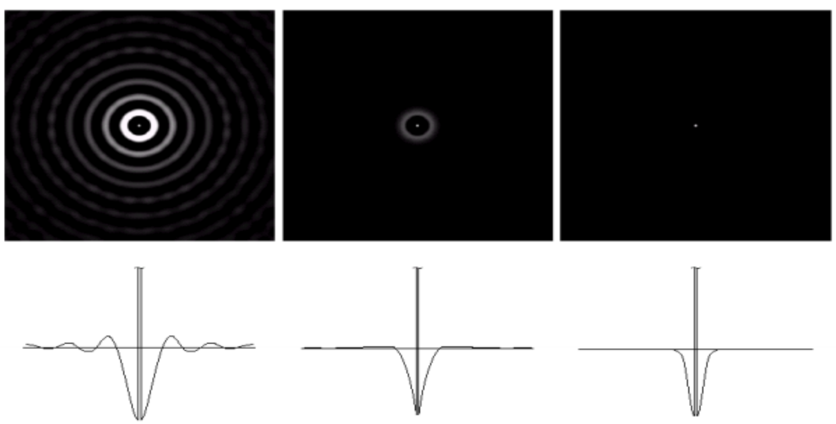
Results of Butterworth highpass filter with D0 = 15 and 30



Gaussian Highpass Filter:

The transfer function of the Gaussian Highpass filter (GHPF) with cutoff frequency locus at a distance D_0 from the center of the frequency rectangle is given by: $H(u,v) = 1 - e^{-D^2(u,v)/2\sigma^2}$

Results of Gaussian highpass filter with D0 = 15 and 30



Spatial representations of the basic highpass filters and corresponding grey-levels profiles:
1) Ideal filter
2) Butterworth filter
3) Gaussian filter

Homomorphic Filter:

An image $f(x,y)$ can be expressed as the product of its illumination, $i(x,y)$ and reflectance $r(x,y)$, components: $f(x,y) = i(x,y)r(x,y)$

This equation can't be used directly to operate on the frequency components o illumination and reflectance because the Fourier transform of a product is not the product of the transforms: $F[f(x,y)] \neq F[i(x,y)]F[r(x,y)]$.

Let we define: $z(x,y) = \ln f(x,y) = \ln i(x,y) + \ln r(x,y)$.

Then the Fourier transform become: $F[z(x,y)] = F[\ln f(x,y)] = F[\ln i(x,y)] + F[\ln r(x,y)]$.

Or, we can write this expression as: $Z(u,v) = F_i(u,v) + F_r(u,v)$.

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*.

We can filter $Z(u,v)$ using a Homomorphic filter $H(u,v)$ so that, $S(u,v) = H(u,v)Z(u,v) = H(u,v)F_i(u,v) + H(u,v)F_r(u,v)$

The filtered image in the spatial domain is:

$$s(x,y) = F[s(u,v)] = F[H(u,v)F_i(u,v)] + F[H(u,v)F_r(u,v)]$$

By defining, $i'(x,y) = F[H(u,v)F_i(u,v)]$ *and* $r'(x,y) = F[H(u,v)F_r(u,v)]$, we can express in the form: $s(x,y) = i'(x,y) + r'(x,y)$

We reverse the process by taking the exponential of the filtered result to form the output image:

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

$i_0(x,y)r_0(x,y)$ *are the illumination and reflectance components of the output i*

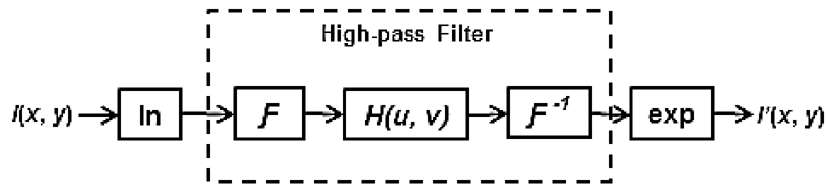
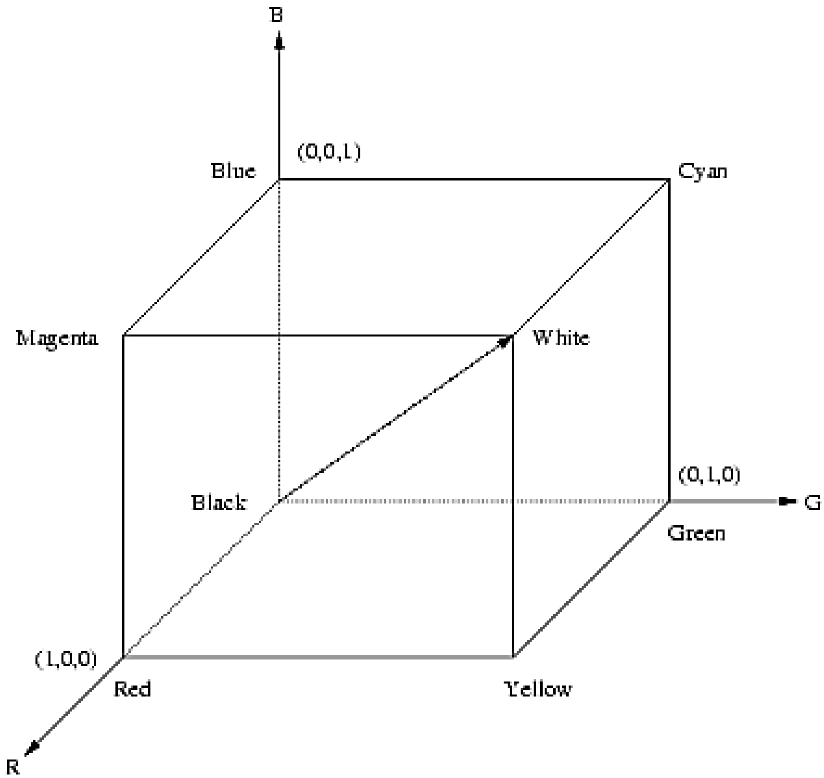


Fig. summary of steps in Homomorphic filtering.

Pseudo color image processing:

Pseudo color image processing consists of assigning colors to gray values based on a specified criterion. The term pseudo or false color is used to differentiate the process of assigning colors to monochrome images from the processes associated with true color images. The principal use of pseudo color is for human visualization and interpretation of fray scale events in an image or sequence of images. One of the principal motivations for using color is the fact that humans can discern thousands of color shades and intensities, compared to only two dozen or so shades of gray.

The RGB color model:



RGB to CMYK conversion formula

The R,G,B values are divided by 255 to change the range from 0..255 to 0..1:

$R' = R/255$; $G' = G/255$; $B' = B/255$

The black key (K) color is calculated from the red (R'), green (G') and blue (B') colors:

$K = 1\text{-max}(R', G', B')$

The cyan color (C) is calculated from the red (R') and black (K) colors:

$C = (1\text{-}R'\text{-}K) / (1\text{-}K)$

The magenta color (M) is calculated from the green (G') and black (K) colors:

$M = (1\text{-}G'\text{-}K) / (1\text{-}K)$

The yellow color (Y) is calculated from the blue (B') and black (K) colors:

$Y = (1\text{-}B'\text{-}K) / (1\text{-}K)$

CMYK to RGB conversion formula

The *R,G,B* values are given in the range of 0..255.

The red (R) color is calculated from the cyan (C) and black (K) colors:

$R = 255 \times (1\text{-}C) \times (1\text{-}K)$

The green color (G) is calculated from the magenta (M) and black (K) colors:

$G = 255 \times (1\text{-}M) \times (1\text{-}K)$

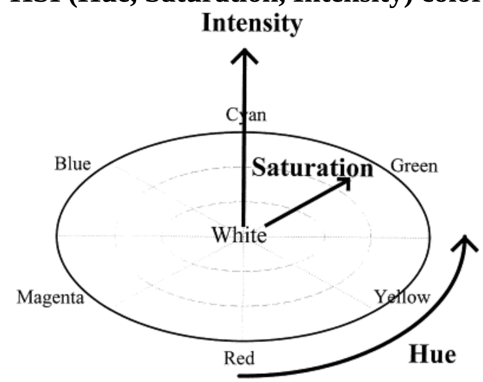
The blue color (B) is calculated from the yellow (Y) and black (K) colors:

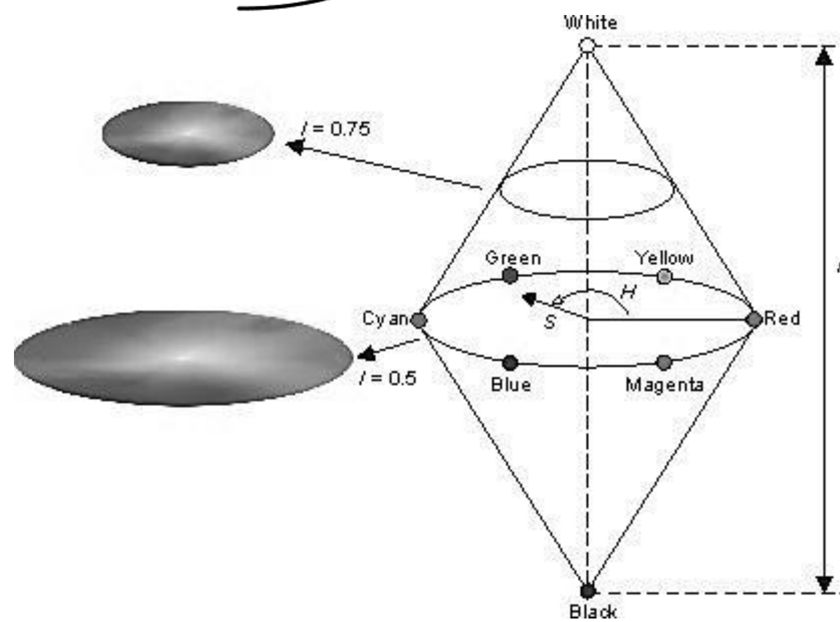
$B = 255 \times (1\text{-}Y) \times (1\text{-}K)$

CMYK color model:

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

HSI (Hue, Saturation, Intensity) color model:





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

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

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

Again converting color from HSI to RGB,
If $0^0 \le \theta < 120^0$ then,

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

$$R = I\left[1 + \frac{S\ cosH}{\cos(60^0 - H)}\right]$$

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

Else if $120^0 \le \theta < 240^0$ then,

$$H = H - 120^0$$

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

$$G = I\left[1 + \frac{S\ cosH}{\cos(60^0 - H)}\right]$$

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

Else if $0^0 \le \theta < 120^0$ then,

$$H = H - 240^0$$

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

$$B = I\left[1 + \frac{S\ cosH}{\cos(60^0 - H)}\right]$$

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

RGB to Gray conversion formula: Standard formula

Y = 0.2126R + 0.7152G + 0.0722B

Luma coding in video systems:

Y=0.299R + 0.587G + 0.114B

Grayscale as single channels of multichannel color images: