ज्ञानेन प्रकाशते जगत्

INDUS UNIVERSITY

Image Processing (EC0703)

Unit-3

B.Tech (Electronics and Communication)

Semester-7

Prof.Hansa Shingrakhia

Academic Year 2019-2020

Contents

In this lecture we will look at image enhancement in the frequency domain

- Jean Baptiste Joseph Fourier
- The Fourier series & the Fourier transform
- Image Processing in the frequency domain
  - Image smoothing
  - Image sharpening
- Fast Fourier Transform

Jean Baptiste Joseph Fourier

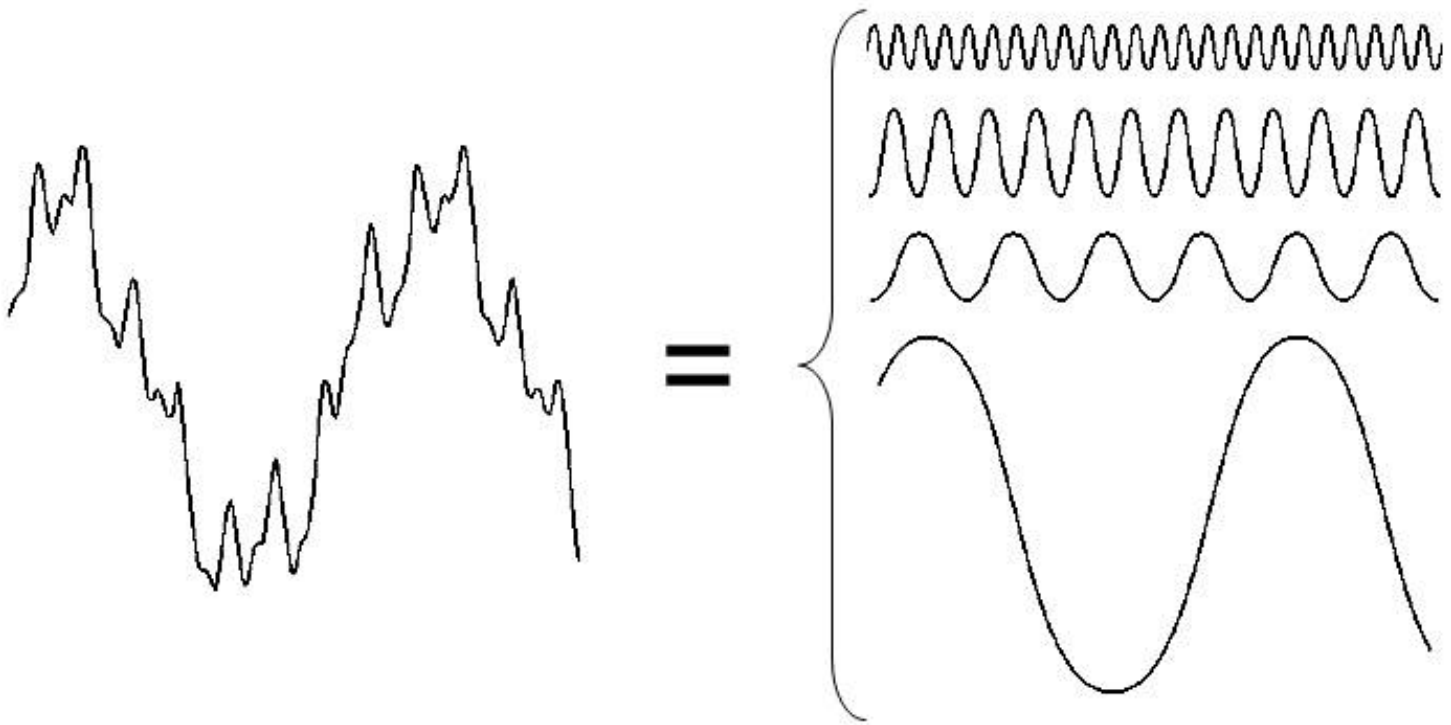
Fourier was born in Auxerre, France in 1768

- Most famous for his work *"La Théorie Analitique de la Chaleur"* published in 1822
- Translated into English in 1878: *"The Analytic Theory of Heat"*

Nobody paid much attention when the work was first published

One of the most important mathematical theories in modern engineering

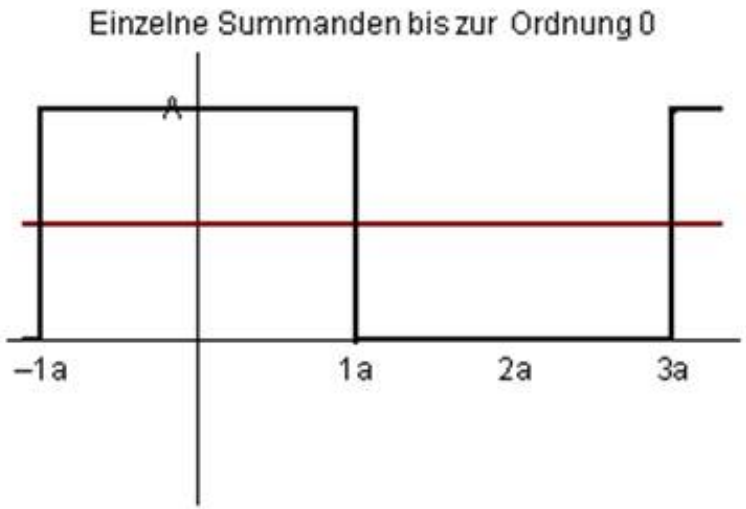
The Big Idea

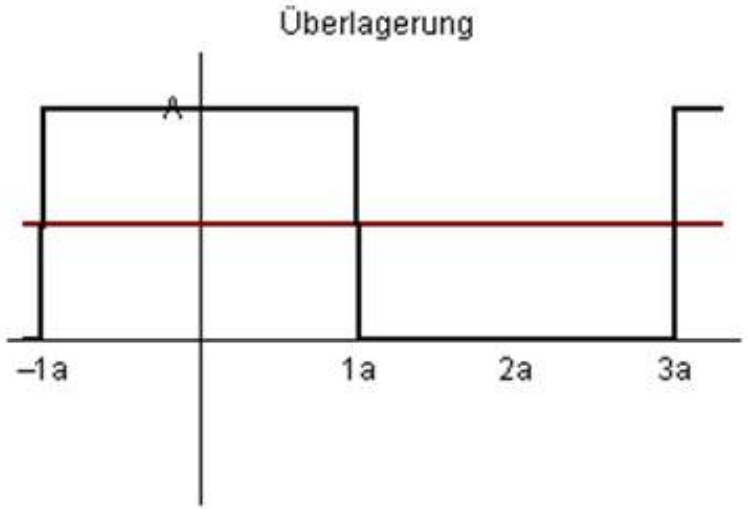


Any function that periodically repeats itself can be expressed as a sum of sines and cosines of different frequencies each multiplied by a different coefficient – a *Fourier series*

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

The Big Idea (cont…)

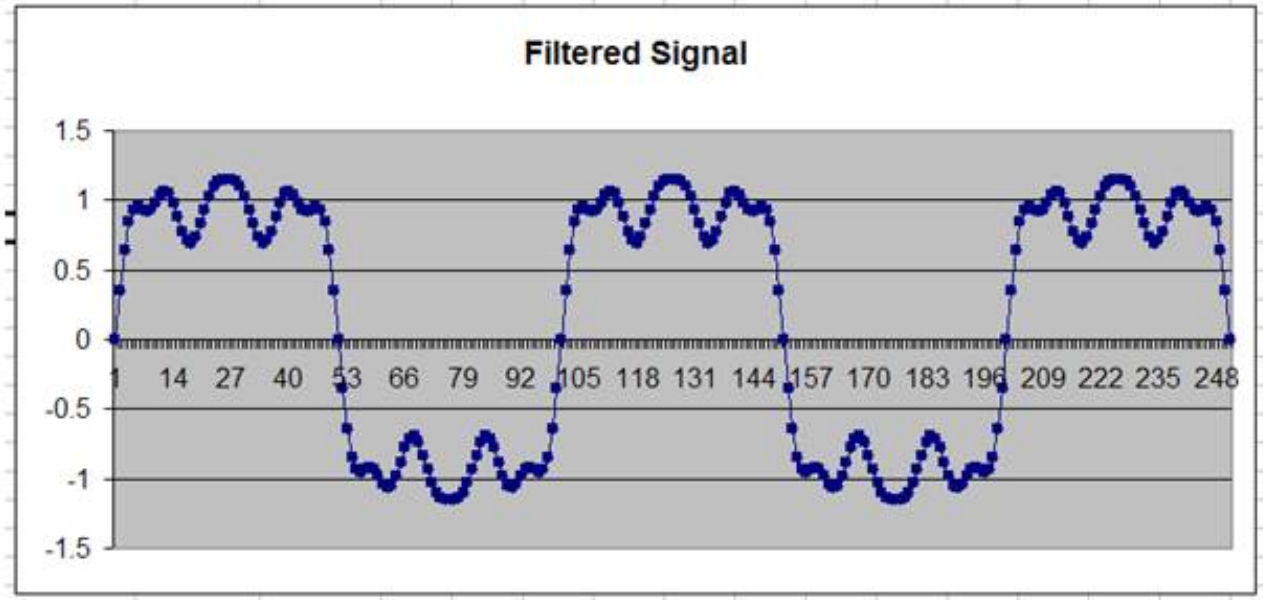


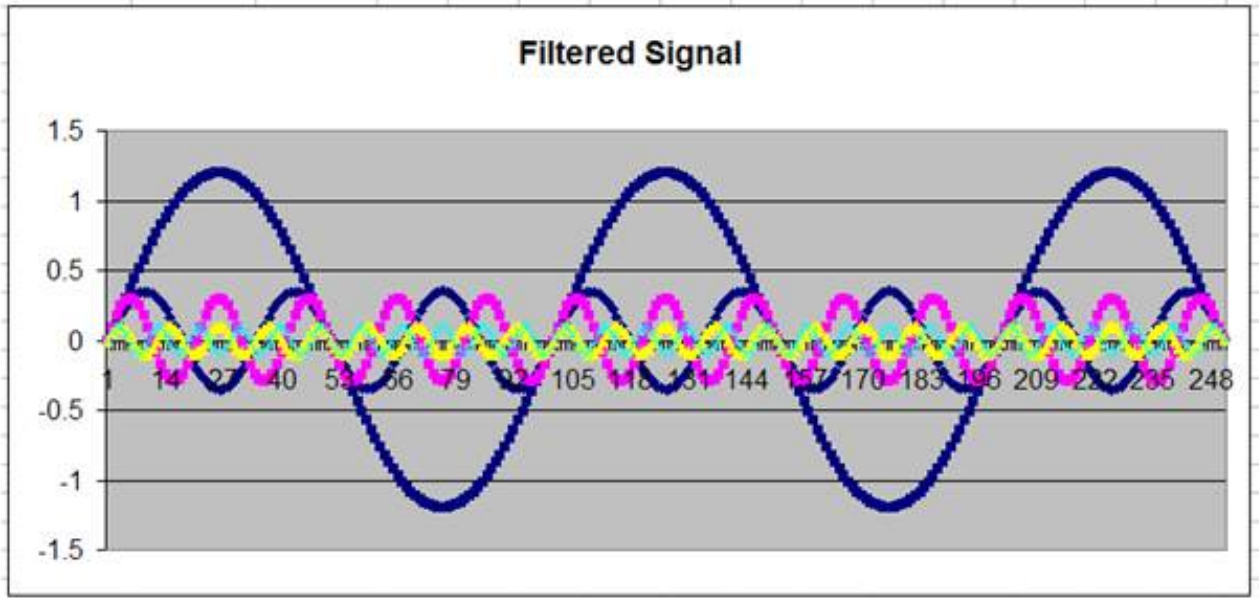


Notice how we get closer and closer to the original function as we add more and more frequencies

Taken from www.tfh-berlin.de/~schwenk/hobby/fourier/Welcome.html

The Big Idea (cont…)





Frequency domain signal processing example in Excel

The Discrete Fourier Transform (DFT)

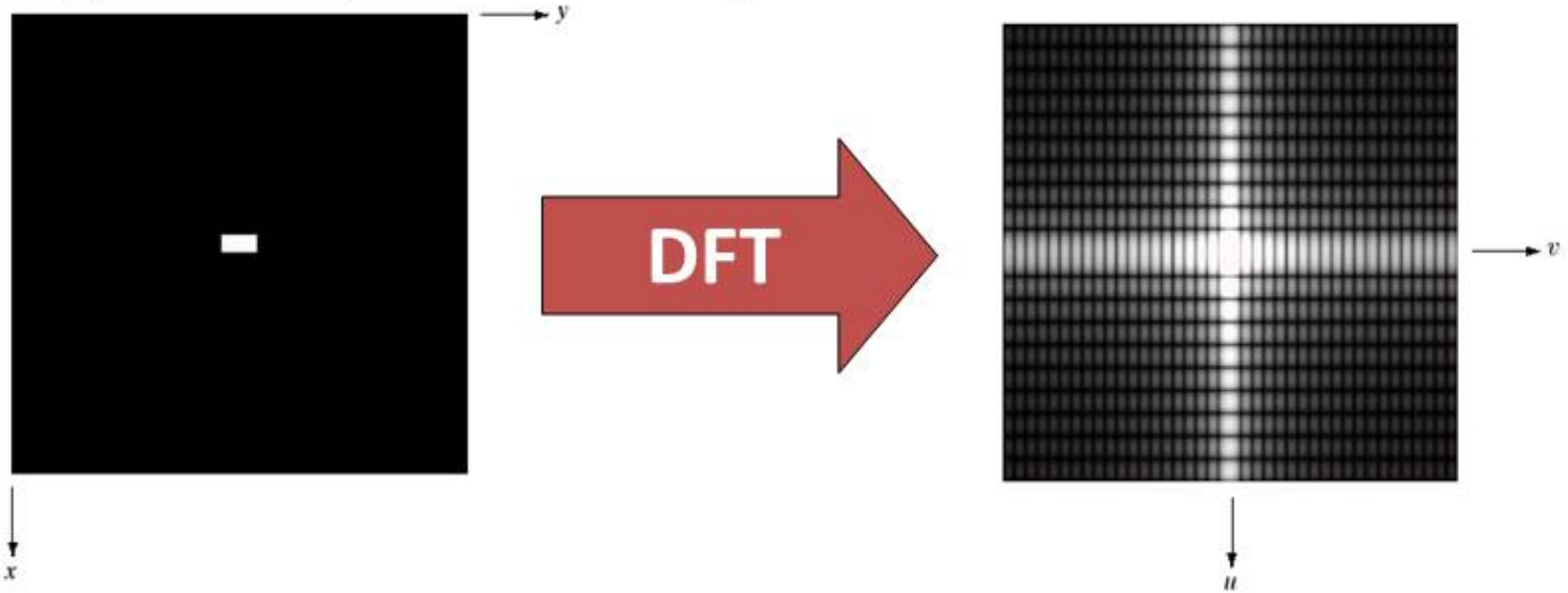
The *Discrete Fourier Transform* of $f(x, y)$, for x = 0, 1, 2...M-1 and y = 0,1,2...N-1, denoted by $F(u, v)$, is given by the equation:

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

for u = 0, 1, 2...M-1 and v = 0, 1, 2...N-1.

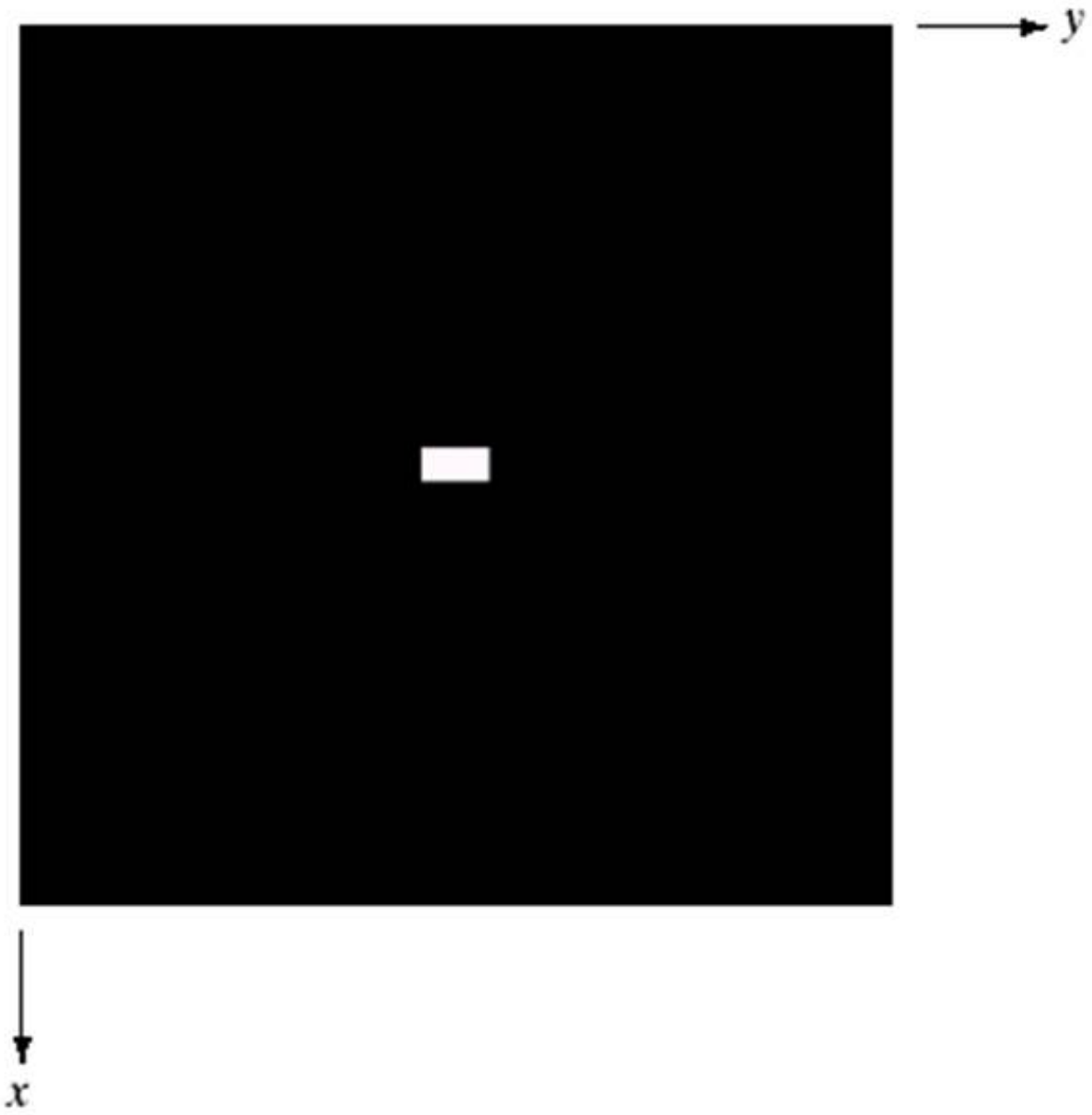
DFT & Images

The DFT of a two dimensional image can be visualised by showing the spectrum of the images component frequencies



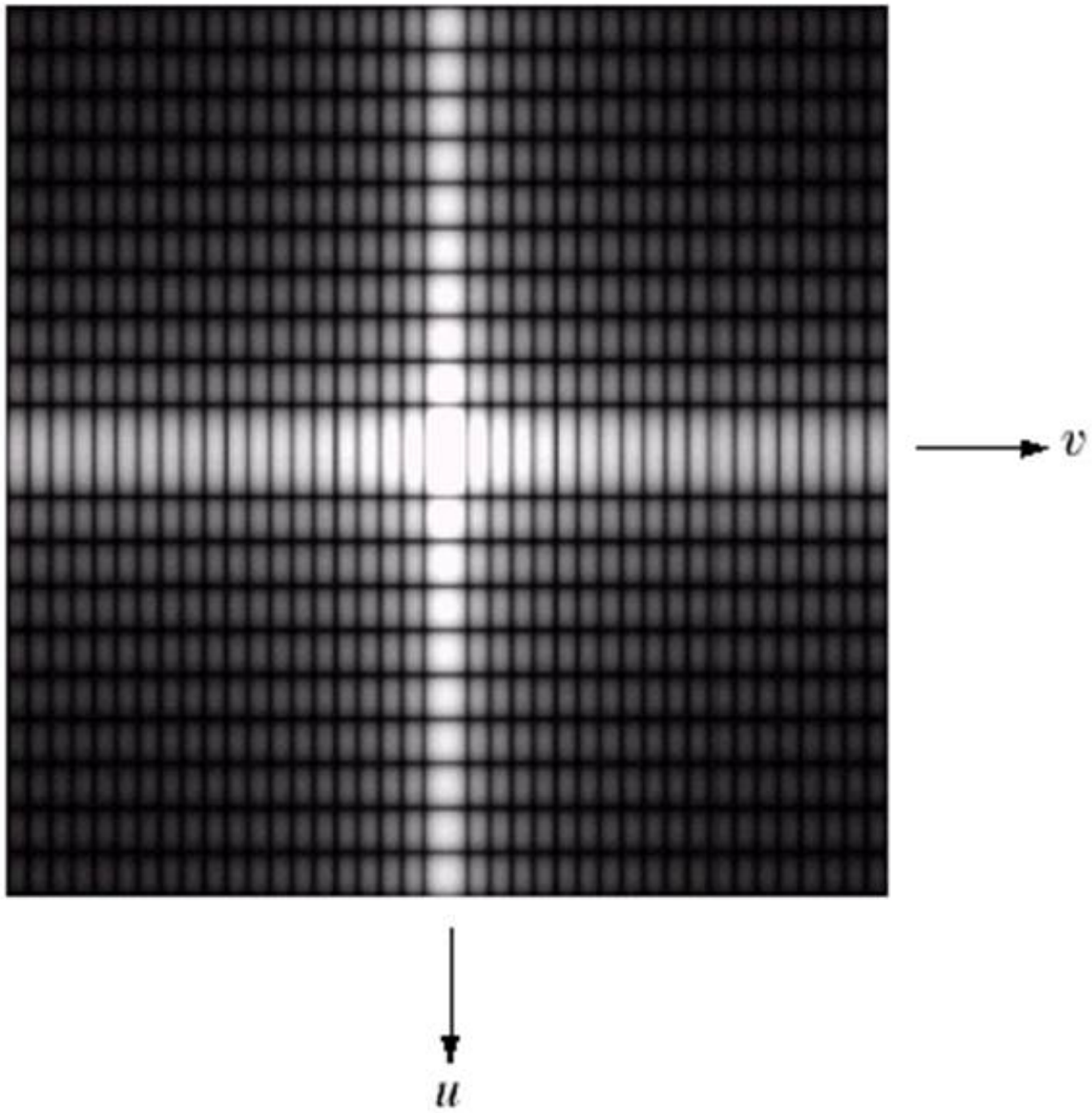
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

DFT & Images



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

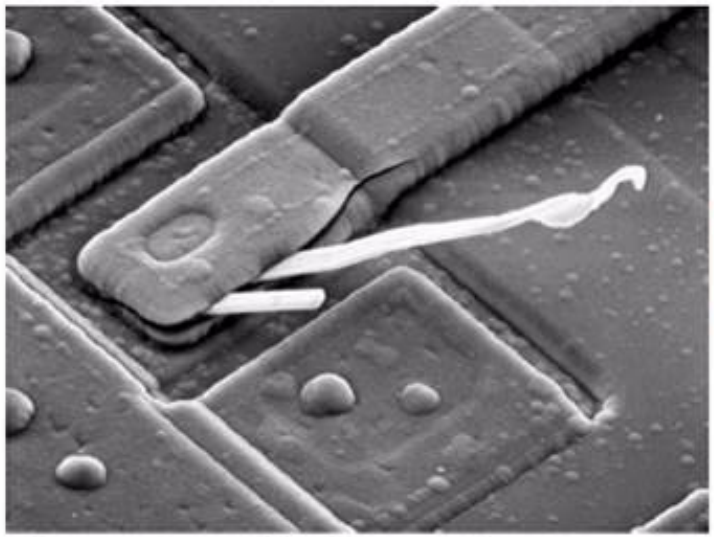
DFT & Images



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

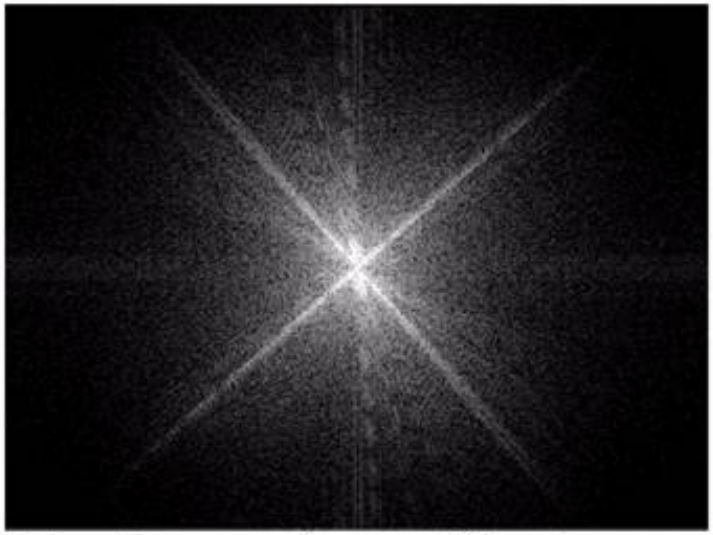
DFT & Images (cont…)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)



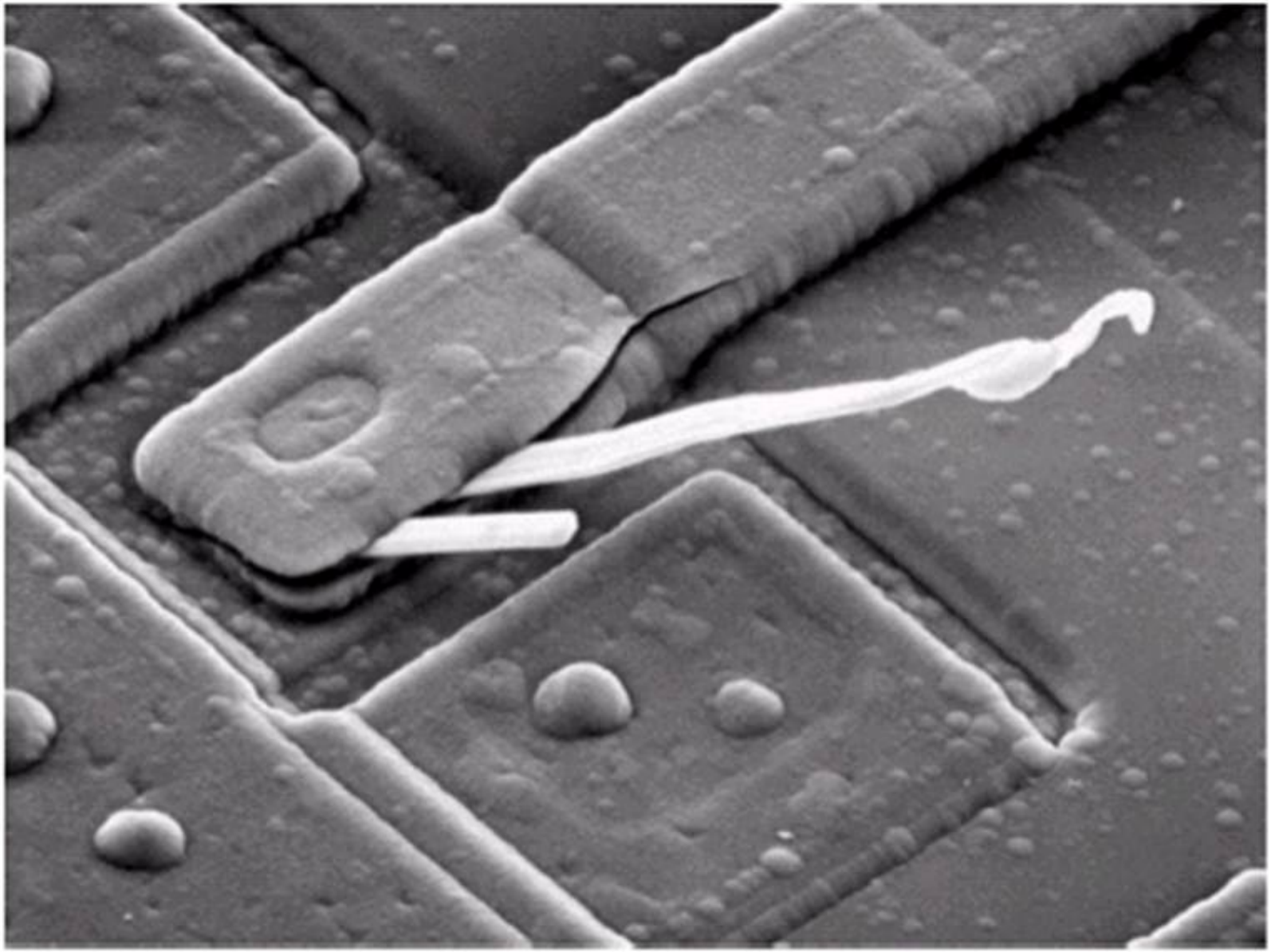
Scanning electron microscope image of an integrated circuit magnified ~2500 times

DFT



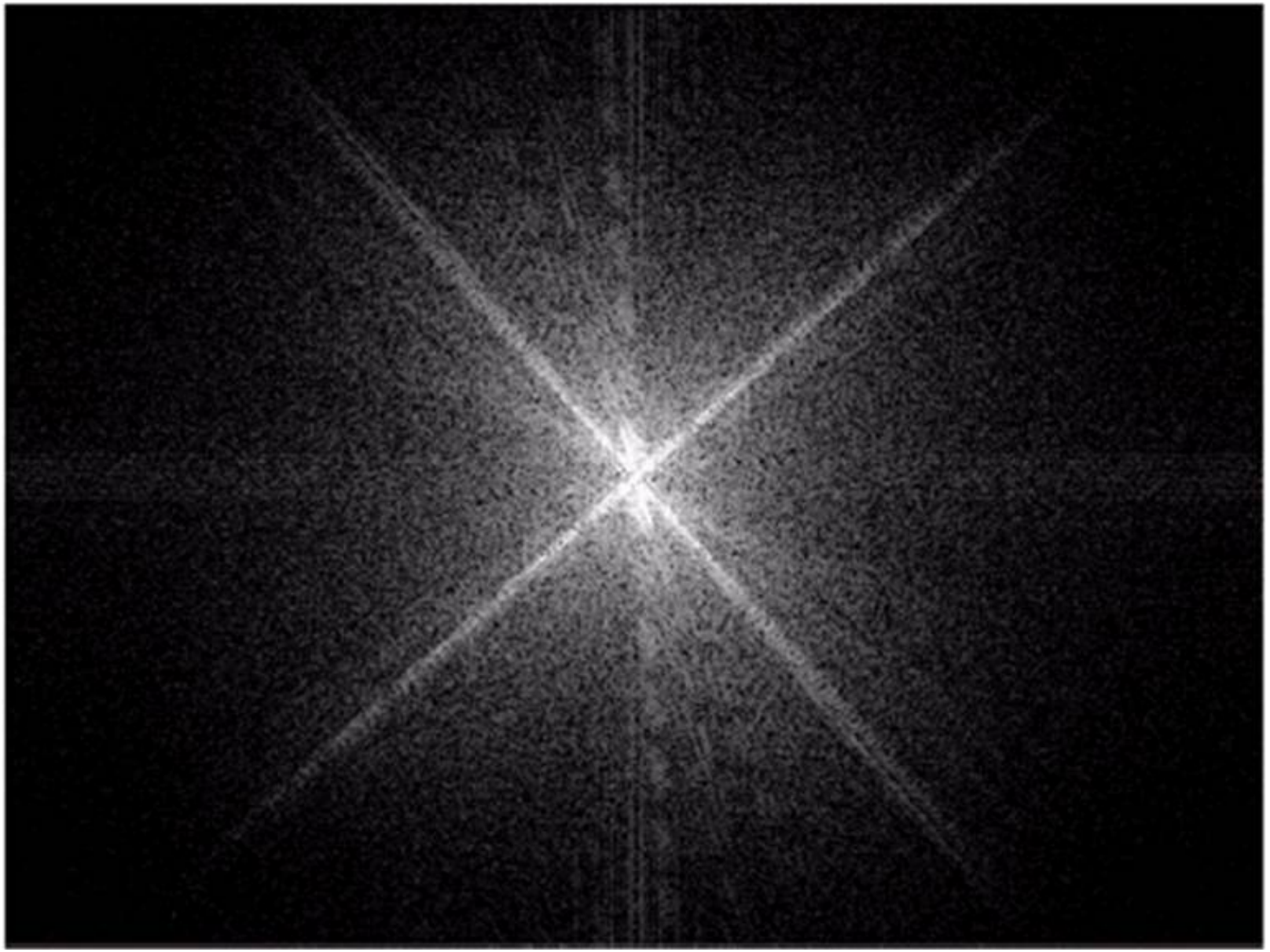
Fourier spectrum of the image

DFT & Images (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

DFT & Images (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

The Inverse DFT

It is really important to note that the Fourier transform is completely **reversible**

The inverse DFT is given by:

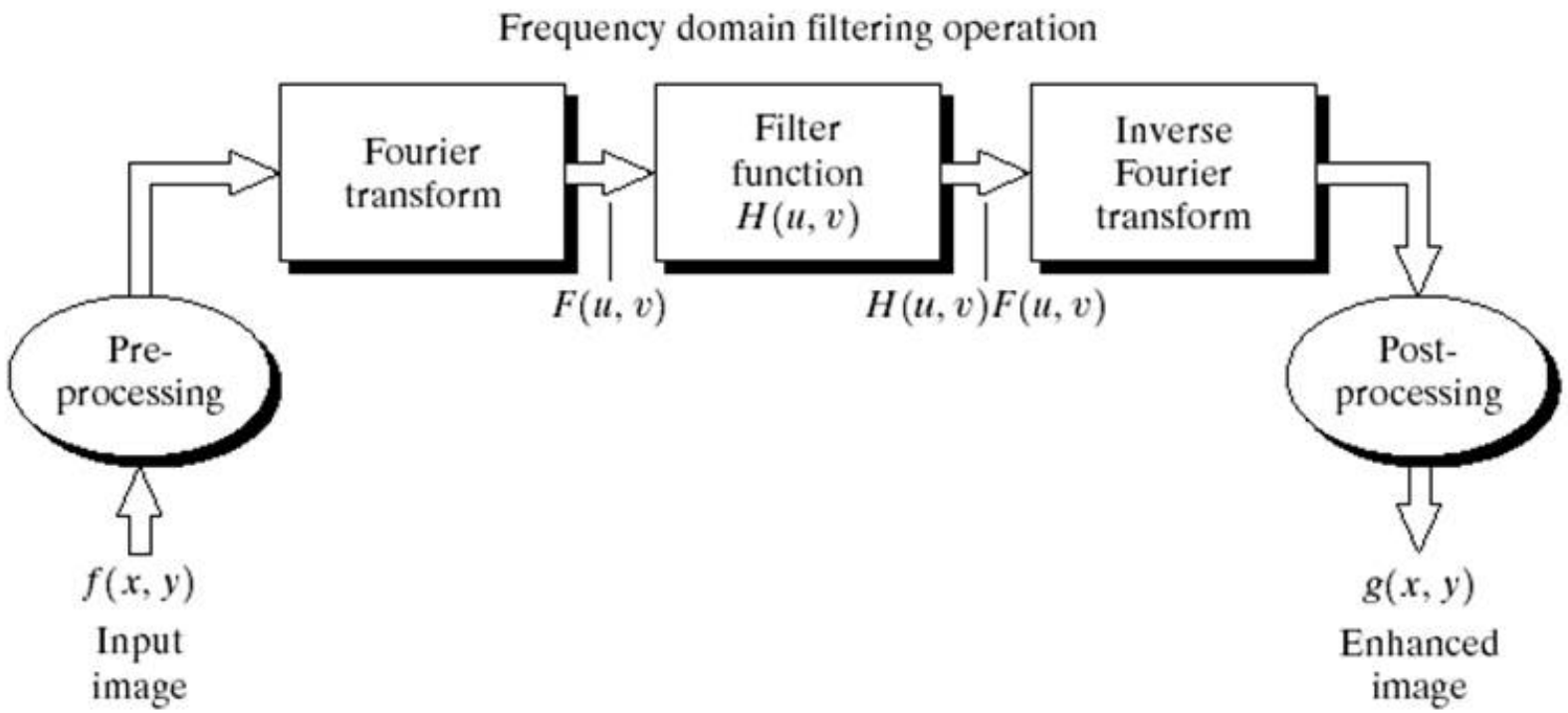
$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi(ux/M + vy/N)}$$

for x = 0, 1, 2…M-1 and y = 0, 1, 2…N-1

The DFT and Image Processing

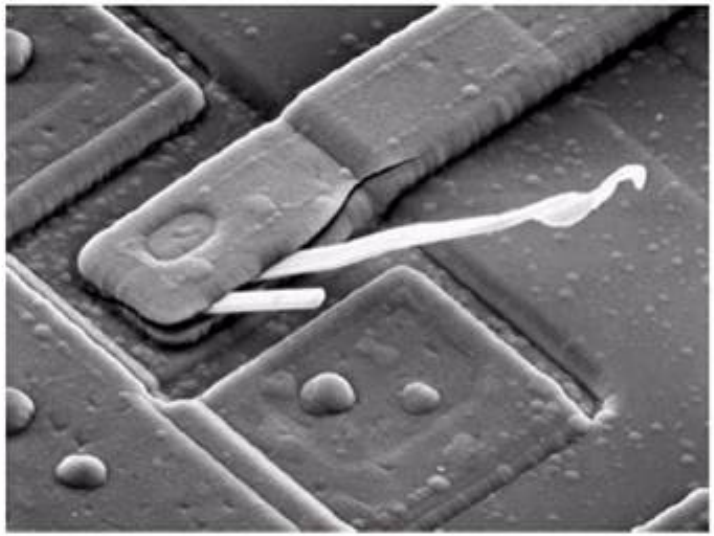
To filter an image in the frequency domain:

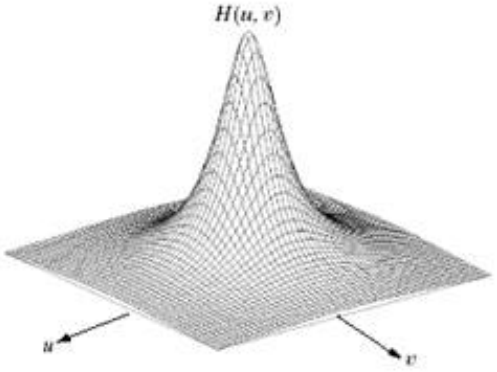
1. Compute $F(u,v)$ the DFT of the image
2. Multiply $F(u,v)$ by a filter function $H(u,v)$
3. Compute the inverse DFT of the result

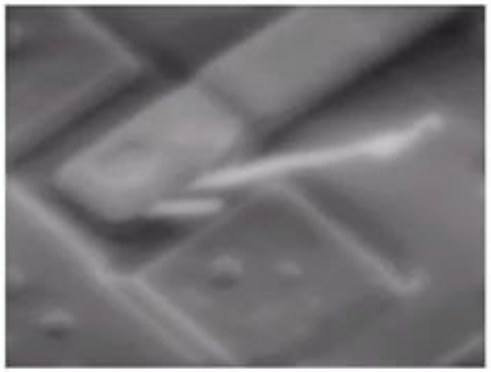


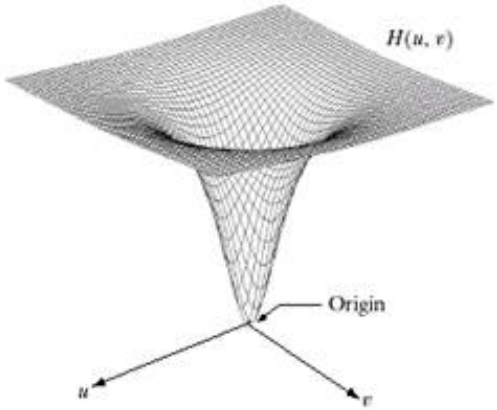
Some Basic Frequency Domain Filters

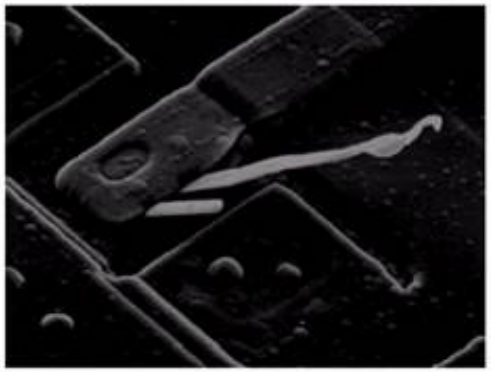
Low Pass Filter







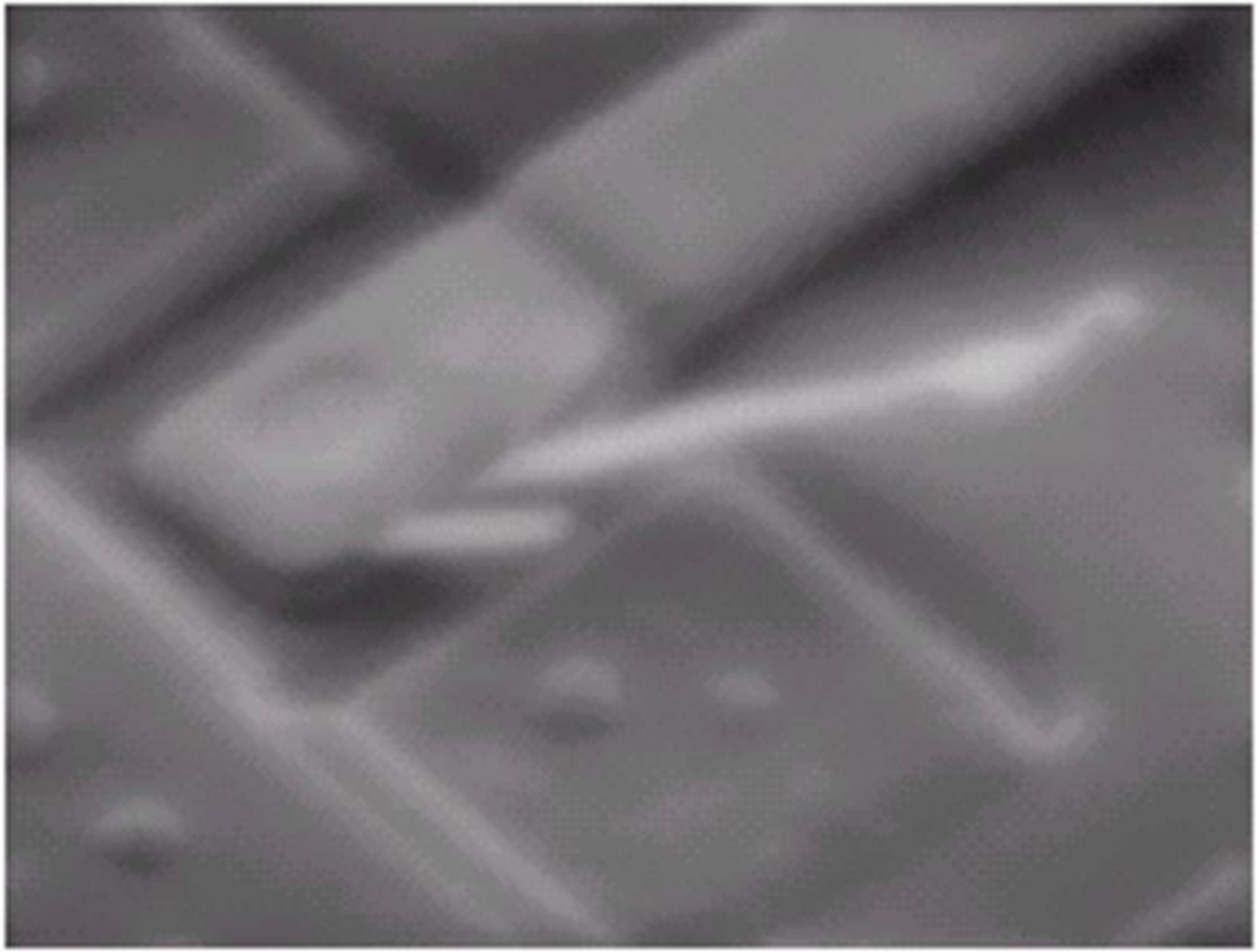




High Pass Filter

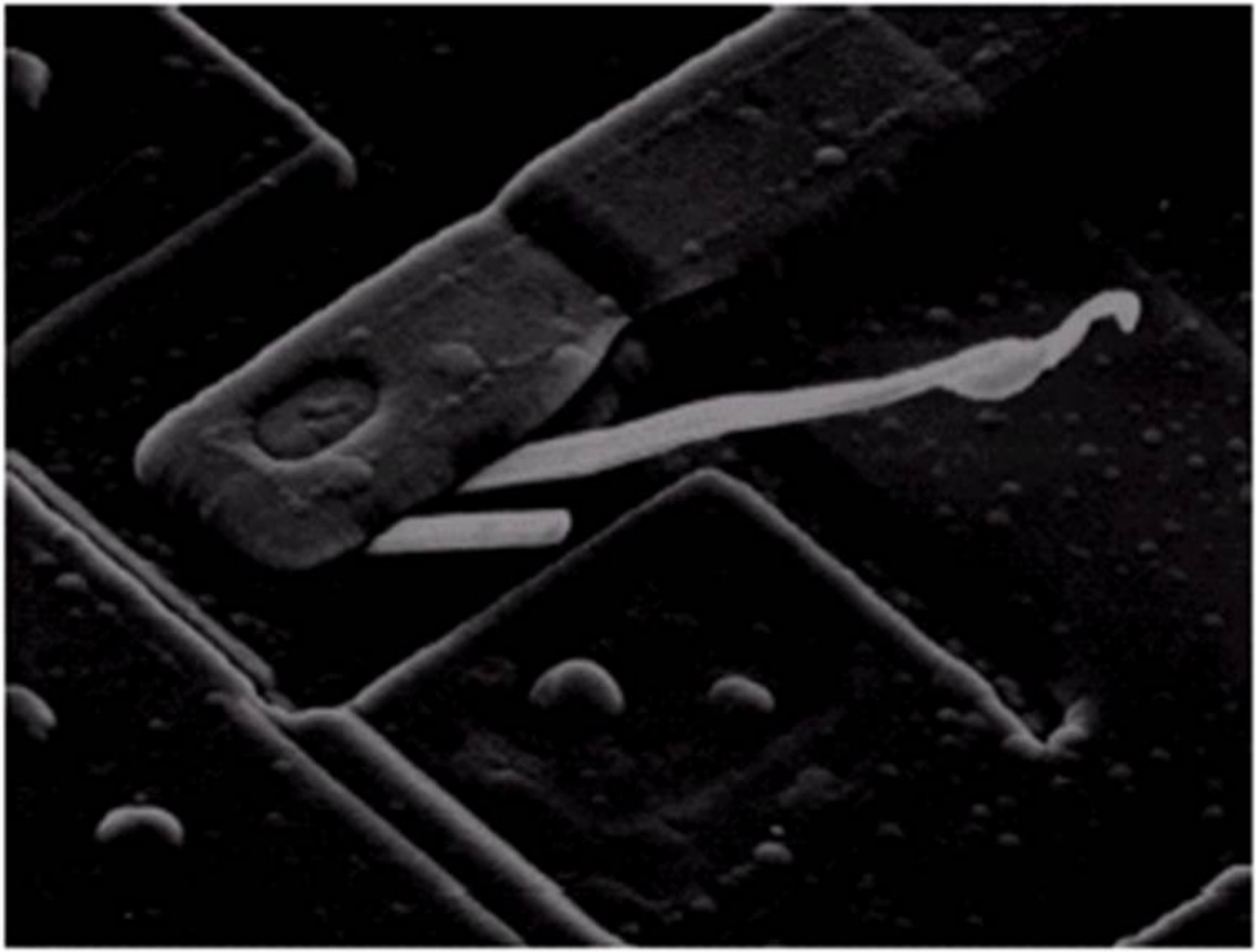
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Some Basic Frequency Domain Filters



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Some Basic Frequency Domain Filters



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Smoothing Frequency Domain Filters

Smoothing is achieved in the frequency domain by dropping out the high frequency components

The basic model for filtering is:

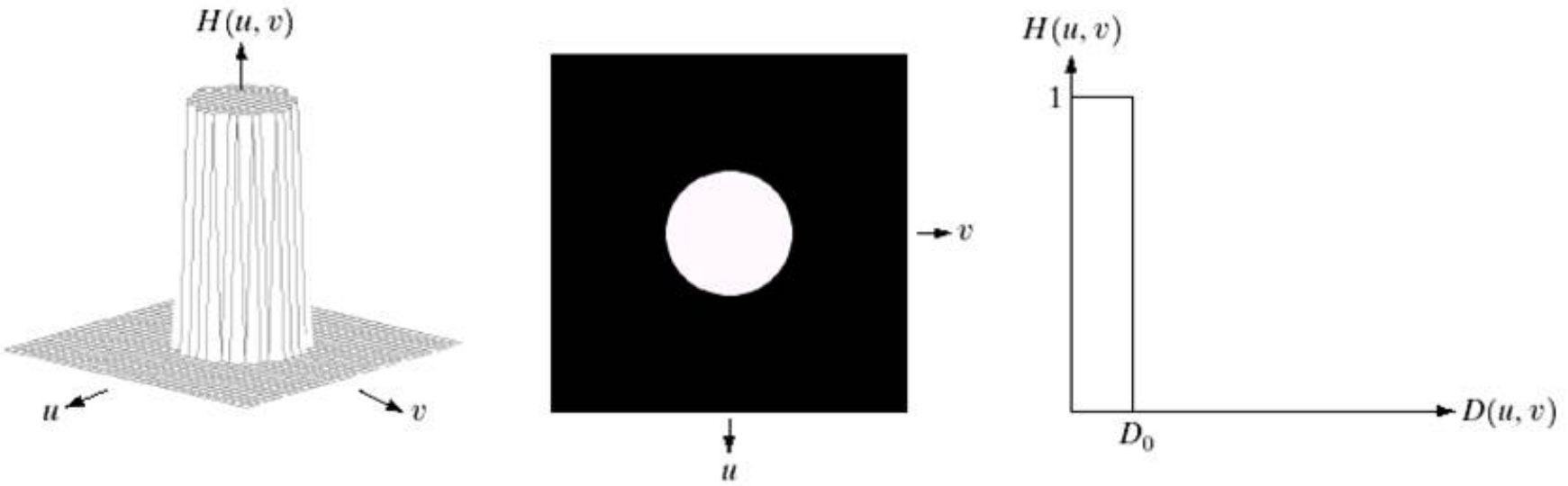
$$G(u,v) = H(u,v)F(u,v)$$

where $F(u,v)$ is the Fourier transform of the image being filtered and $H(u,v)$ is the filter transform function

Low pass filters – only pass the low frequencies, drop the high ones

Ideal Low Pass Filter

Simply cut off all high frequency components that are a specified distance D_0 from the origin of the transform



changing the distance changes the behaviour of the filter

Ideal Low Pass Filter (cont…)

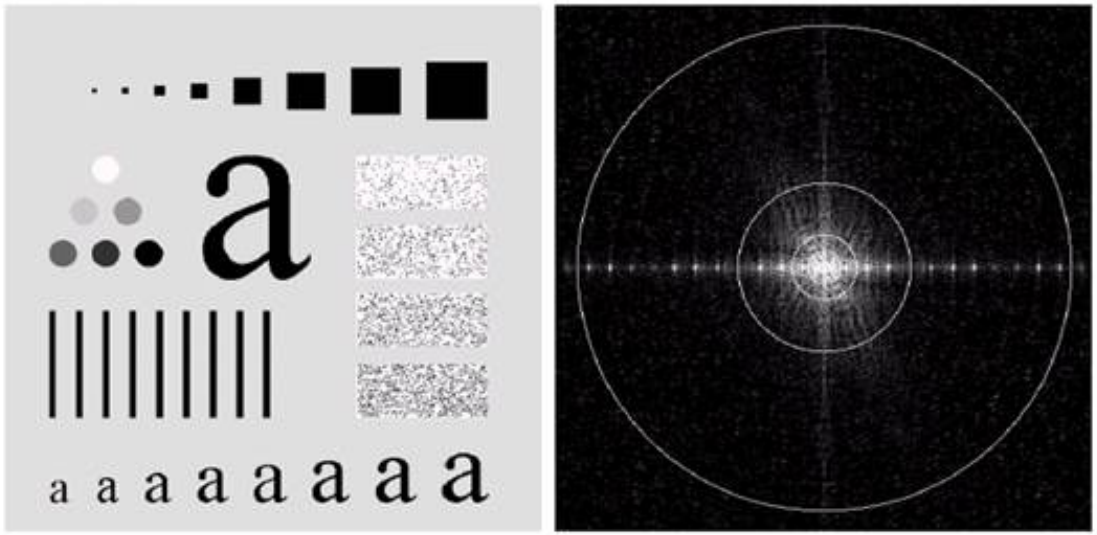
The transfer function for the ideal low pass filter can be given as:

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

where $D(u,v)$ is given as:

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

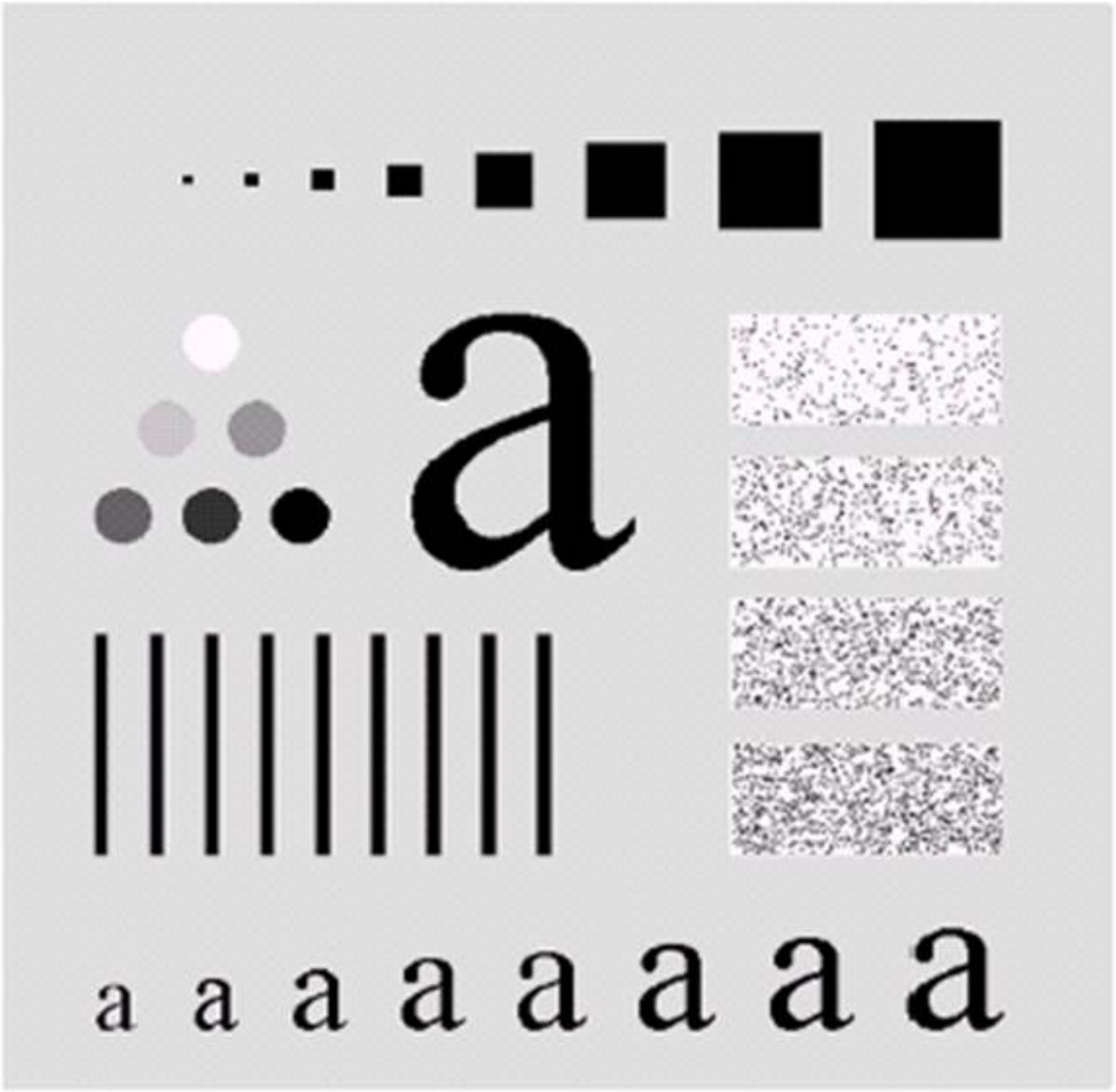
Ideal Low Pass Filter (cont…)



Above we show an image, it's Fourier spectrum and a series of ideal low pass filters of radius 5, 15, 30, 80 and 230 superimposed on top of it

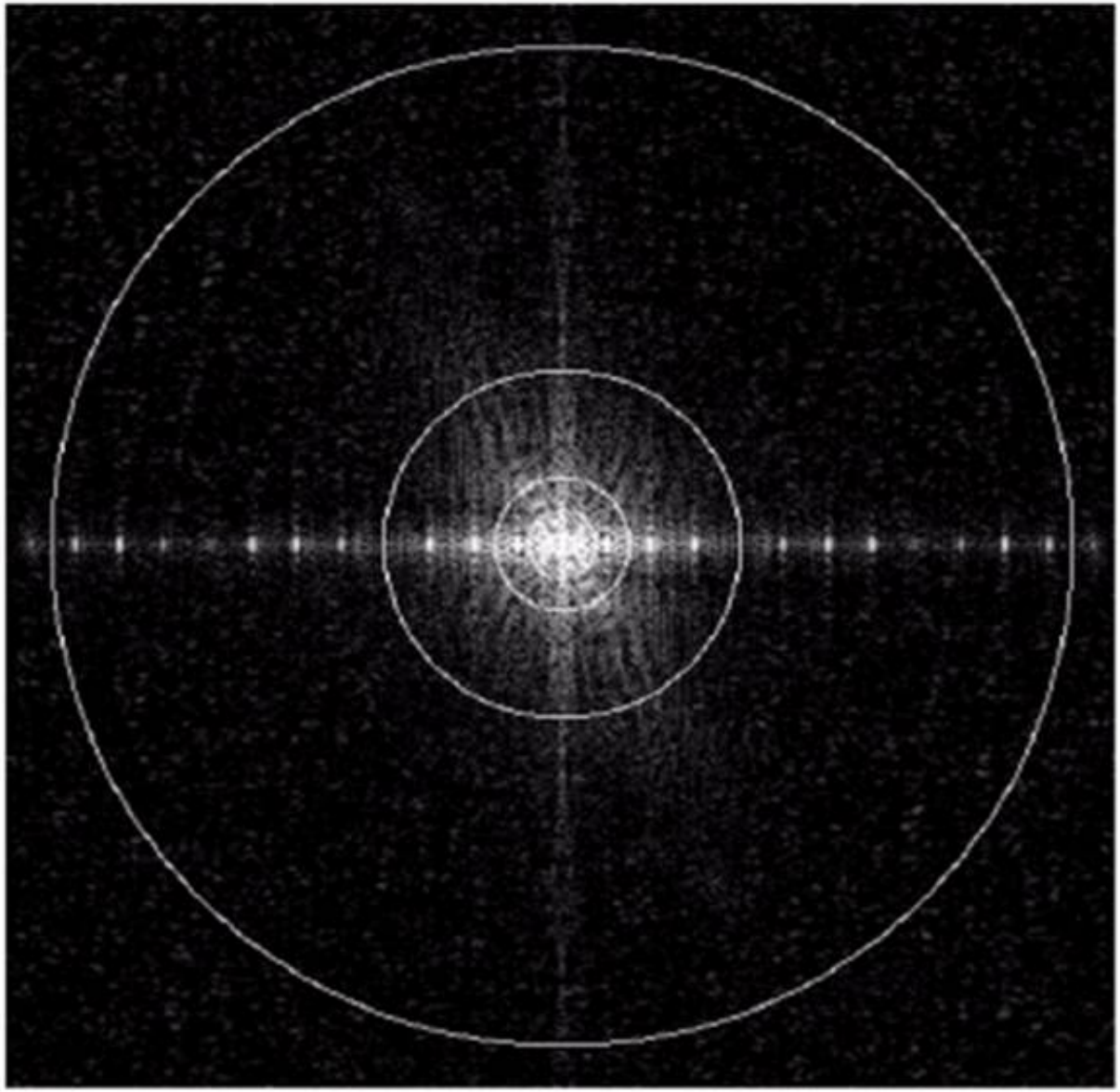
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Ideal Low Pass Filter (cont…)



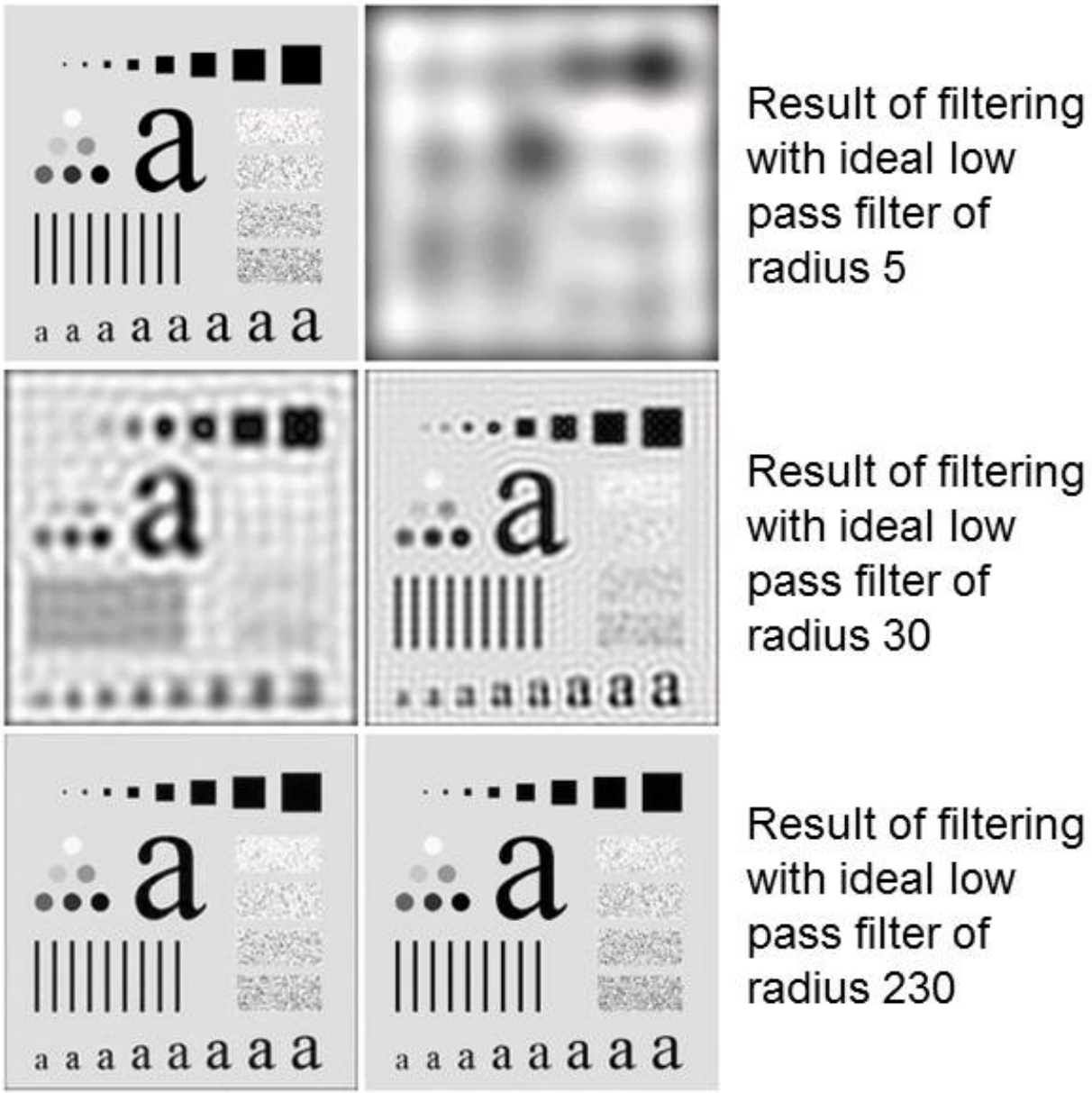
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Ideal Low Pass Filter (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Ideal Low Pass Filter (cont…)



Original image

Result of filtering with ideal low pass filter of radius 5

Result of filtering with ideal low pass filter of radius 15

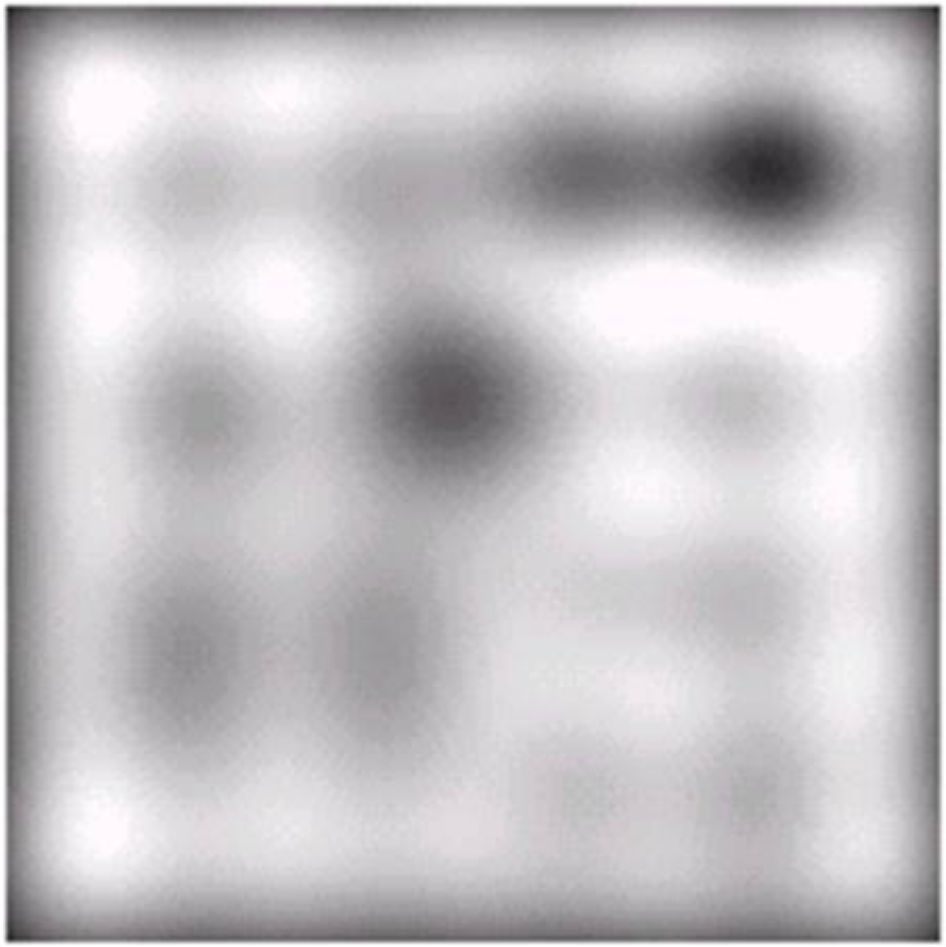
Result of filtering with ideal low pass filter of radius 30

Result of filtering with ideal low pass filter of radius 80

Result of filtering with ideal low pass filter of radius 230

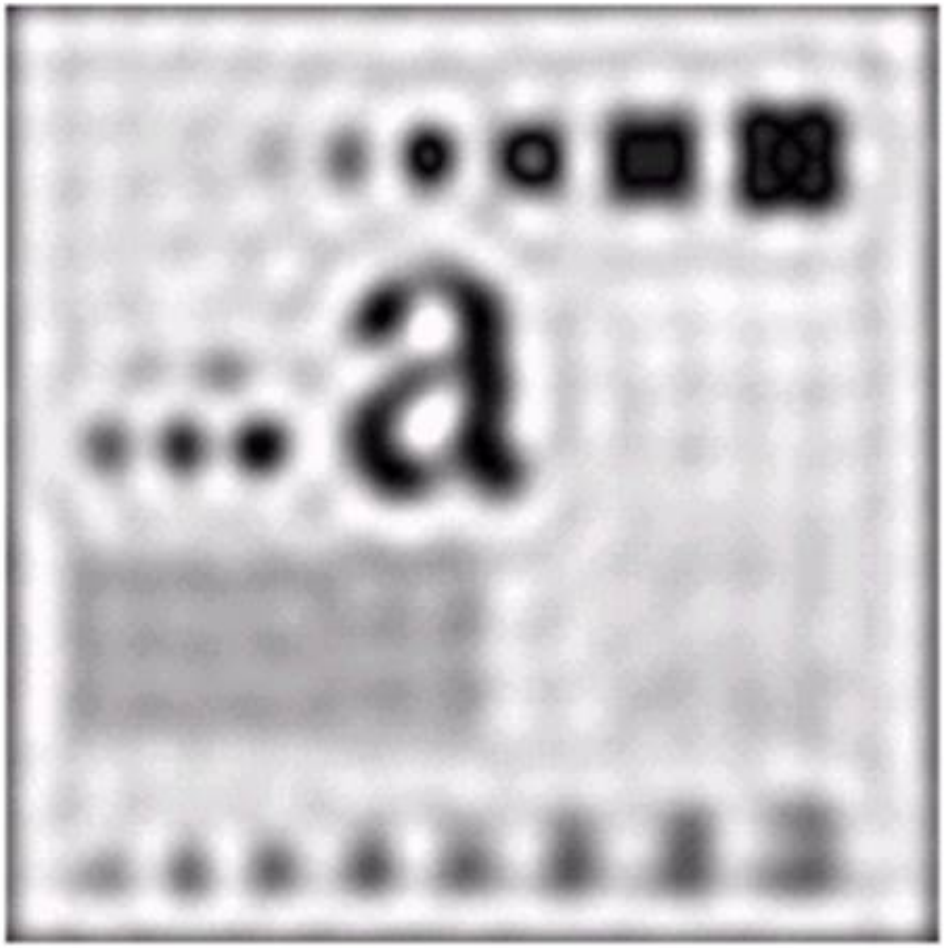
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Ideal Low Pass Filter (cont…)



Result of filtering with ideal low pass filter of radius 5

Ideal Low Pass Filter (cont…)

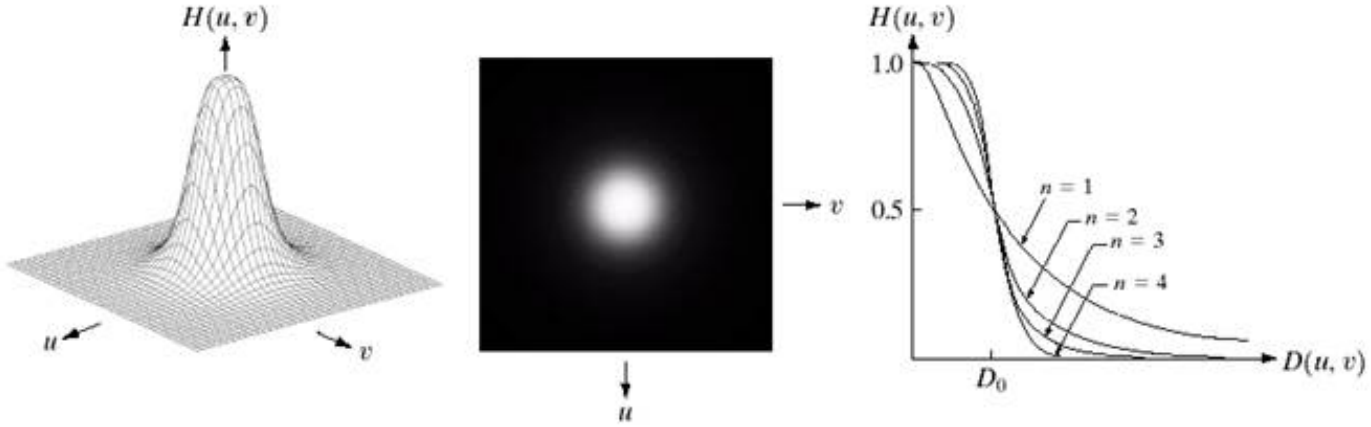


Result of filtering with ideal low pass filter of radius 15

Butterworth Lowpass Filters

The transfer function of a Butterworth lowpass filter of order n with cutoff frequency at distance D_0 from the origin is defined as:

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



Butterworth Lowpass Filter (cont…)

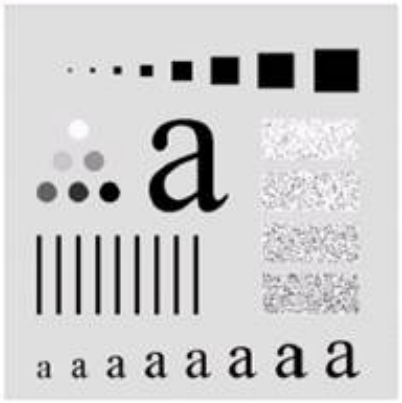
Original image

Result of filtering with Butterworth filter of order 2 and cutoff radius 5

Result of filtering with Butterworth filter of order 2 and cutoff radius 15

Result of filtering with Butterworth filter of order 2 and cutoff radius 30

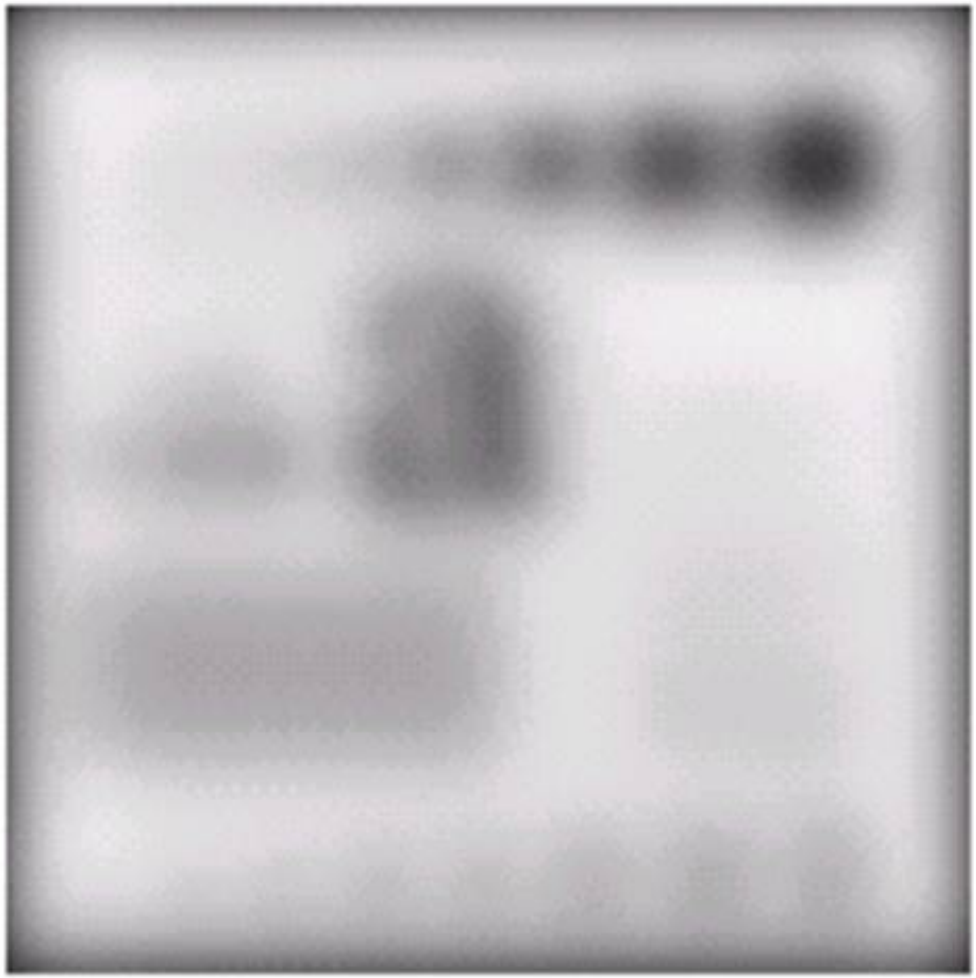
Result of filtering with Butterworth filter of order 2 and cutoff radius 80



Result of filtering with Butterworth filter of order 2 and cutoff radius 230

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

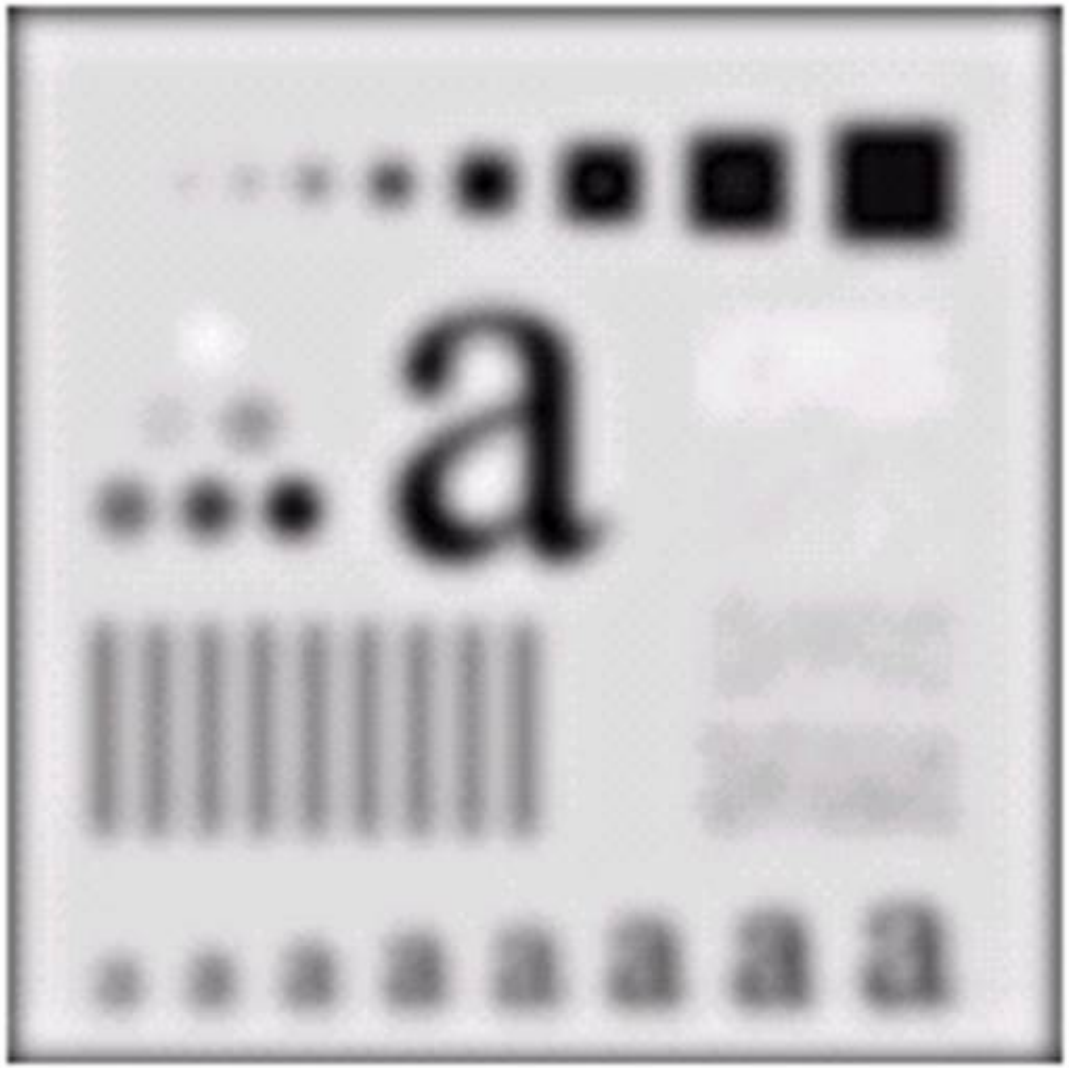
Butterworth Lowpass Filter (cont…)



Result of filtering with Butterworth filter of order 2 and cutoff radius 5

Butterworth Lowpass Filter (cont…)

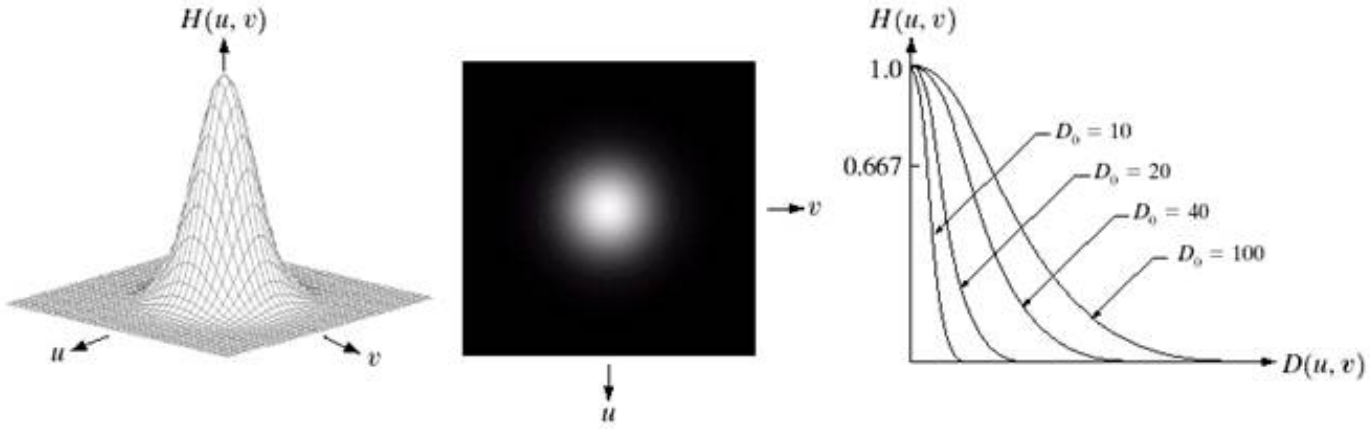
Result of filtering with Butterworth filter of order 2 and cutoff radius 15



Gaussian Lowpass Filters

The transfer function of a Gaussian lowpass filter is defined as:

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



Gaussian Lowpass Filters (cont…)

Original image

Result of filtering with Gaussian filter with cutoff radius 5

Result of filtering with Gaussian filter with cutoff radius 15

Result of filtering with Gaussian filter with cutoff radius 30

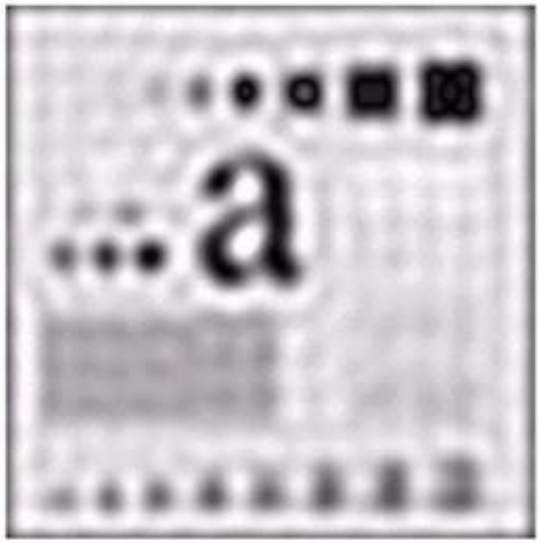
Result of filtering with Gaussian filter with cutoff radius 85

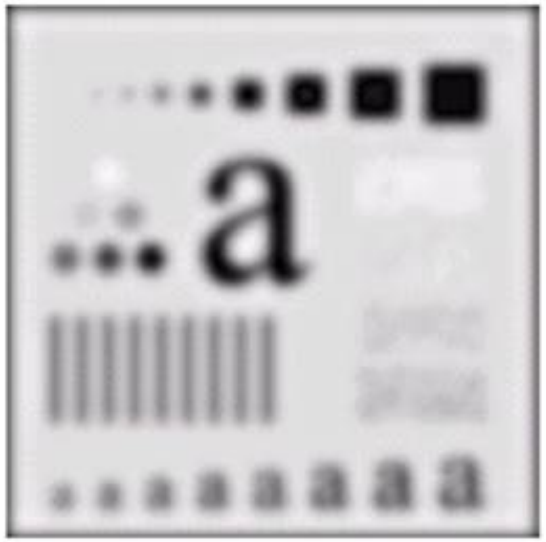
Result of filtering with Gaussian filter with cutoff radius 230

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Lowpass Filters Compared

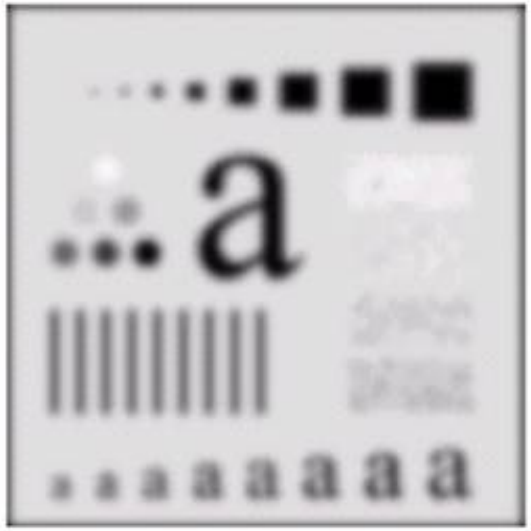
Result of filtering with ideal low pass filter of radius 15





Result of filtering with Butterworth filter of order 2 and cutoff radius 15

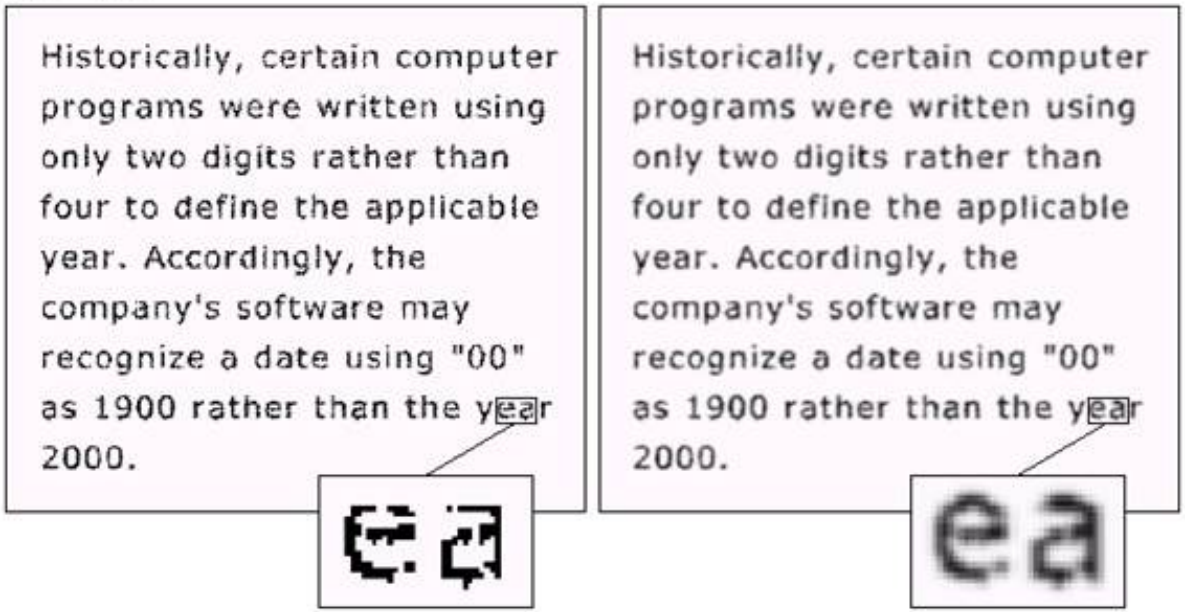
Result of filtering with Gaussian filter with cutoff radius 15



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

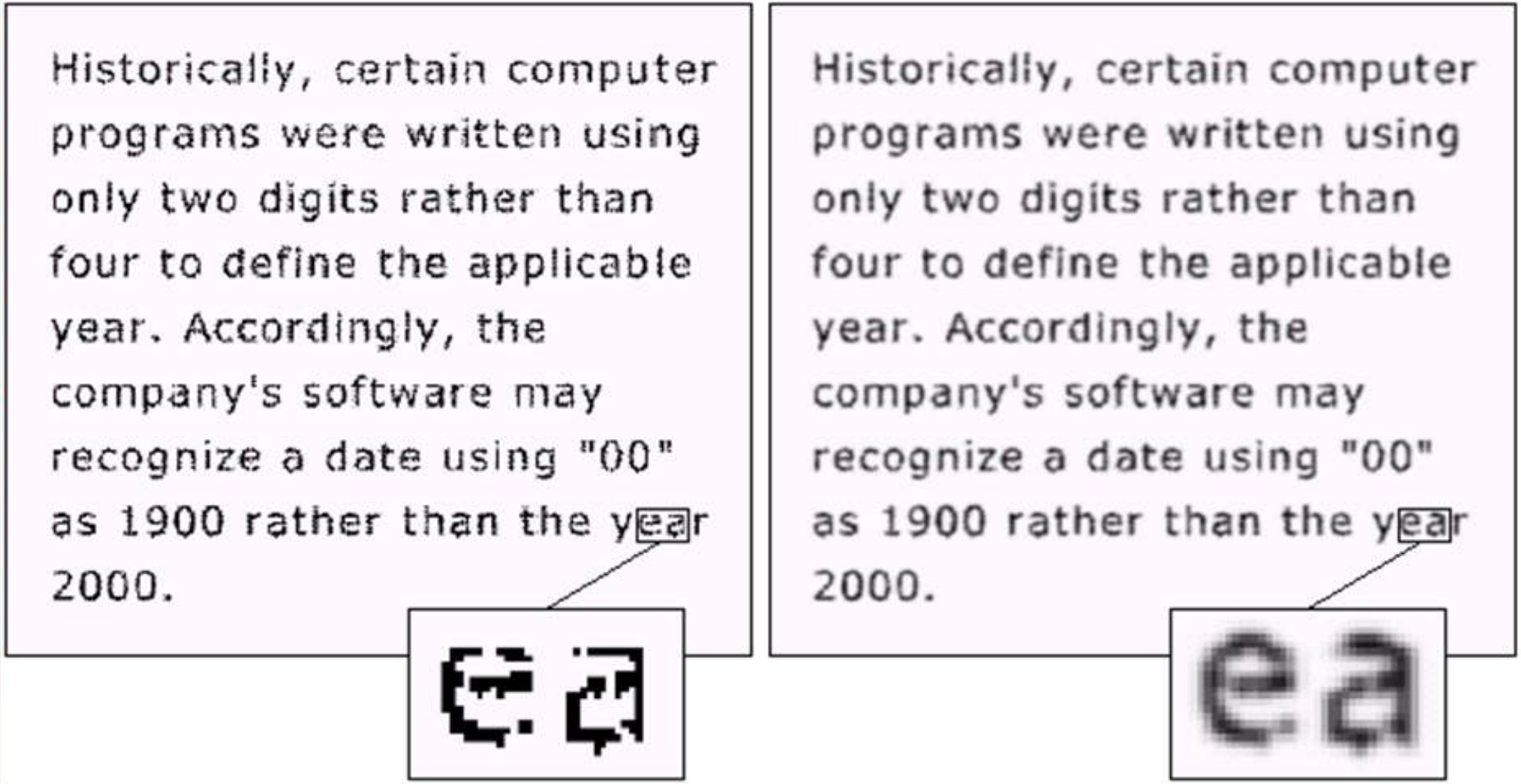
Lowpass Filtering Examples

A low pass Gaussian filter is used to connect broken text



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Lowpass Filtering Examples



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Lowpass Filtering Examples (cont…)

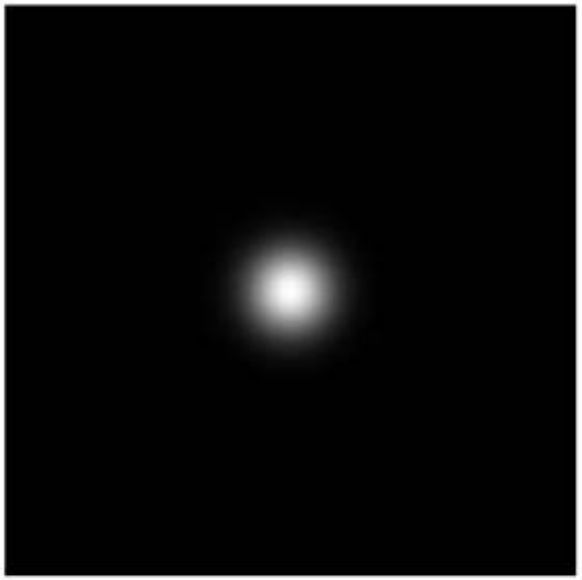
Different lowpass Gaussian filters used to remove blemishes in a photograph

Lowpass Filtering Examples (cont…)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

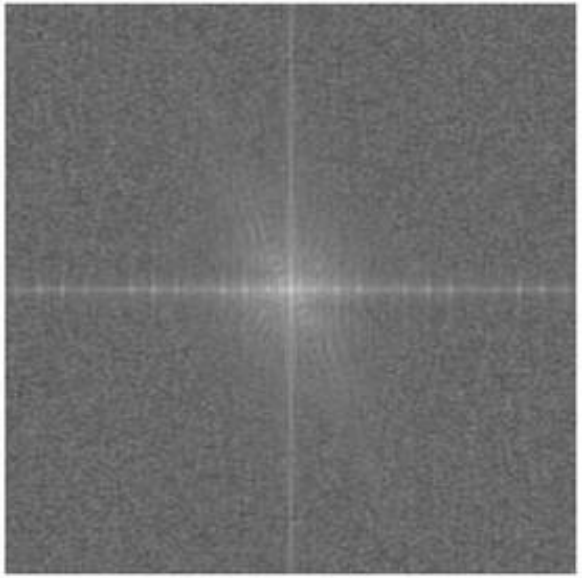
Lowpass Filtering Examples (cont…)

Original image



Gaussian lowpass filter

Spectrum of original image



Processed image

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Sharpening in the Frequency Domain

Edges and fine detail in images are associated with high frequency components

High pass filters – only pass the high frequencies, drop the low ones

High pass frequencies are precisely the reverse of low pass filters, so:

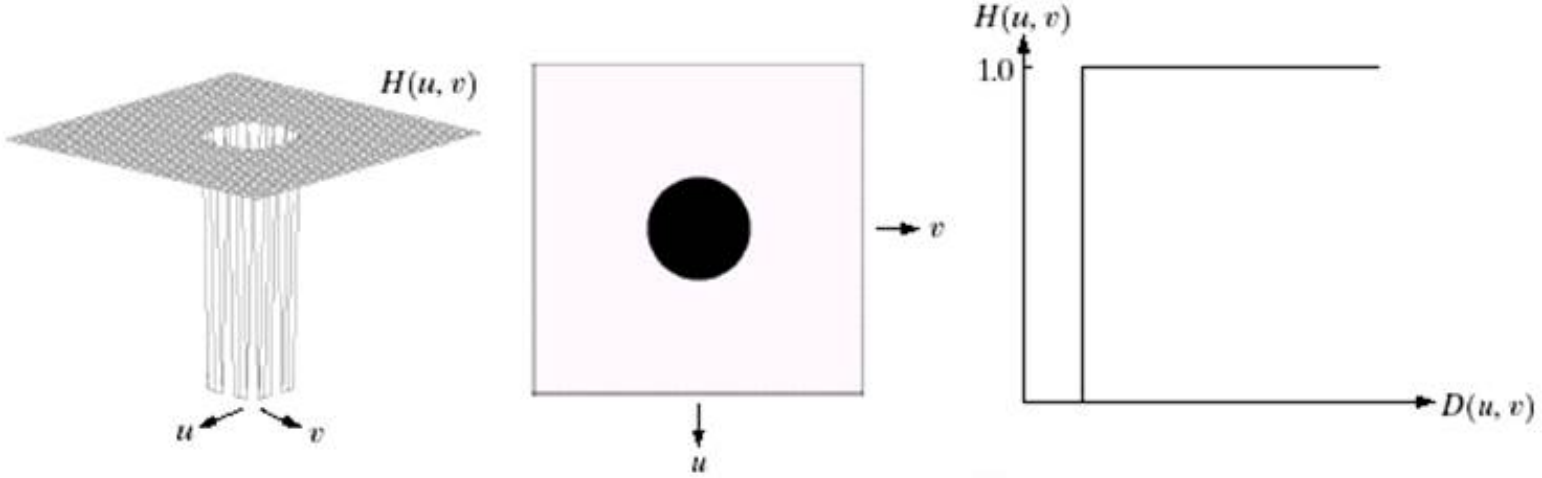
$$H_{hp}(u, v) = 1 - H_{lp}(u, v)$$

Ideal High Pass Filters

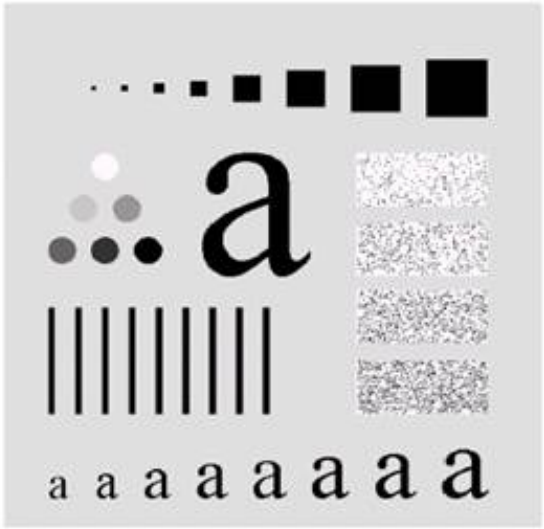
The ideal high pass filter is given as:

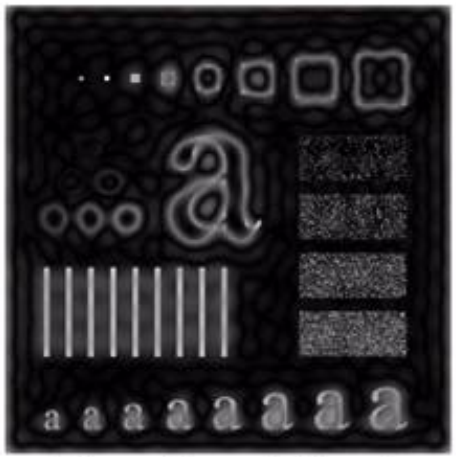
$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$$

where D_0 is the cut off distance as before

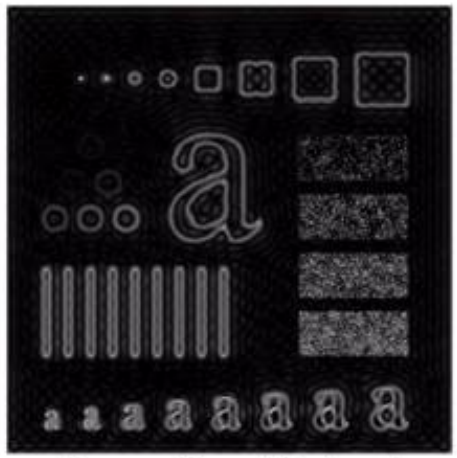


Ideal High Pass Filters (cont…)





Results of ideal high pass filtering with D_0 = 15



Results of ideal high pass filtering with D_0 = 30

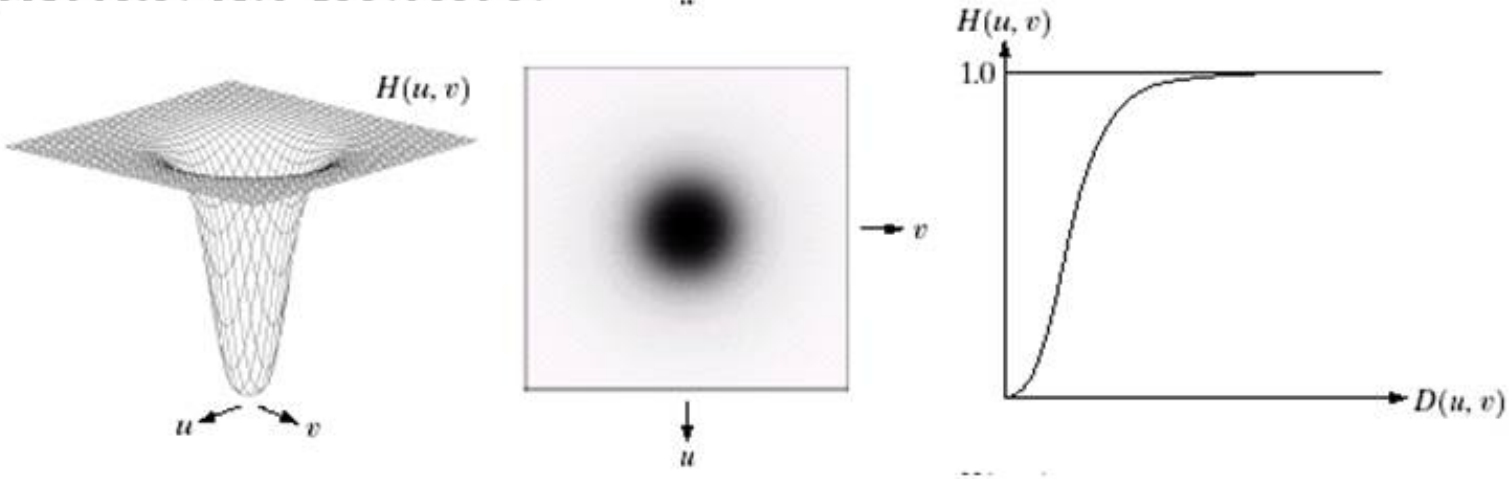
Results of ideal high pass filtering with D_0 = 80

Butterworth High Pass Filters

The Butterworth high pass filter is given as:

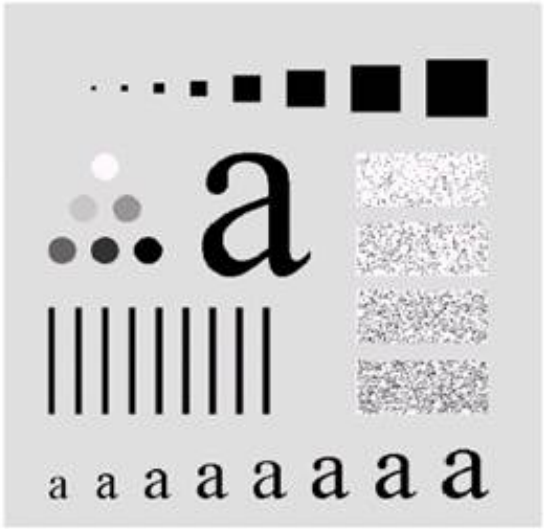
$$H(u,v) = \frac{1}{1+[D_0 / D(u,v)]^{2n}}$$

where n is the order and D_0 is the cut off distance as before

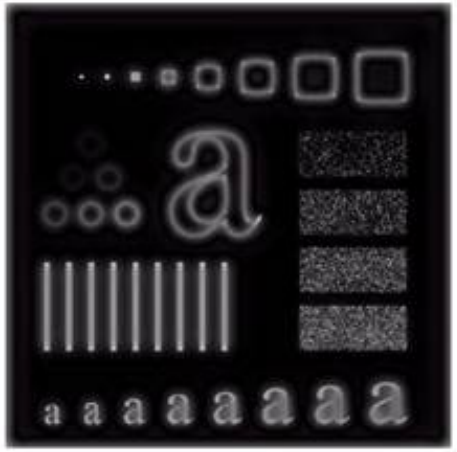


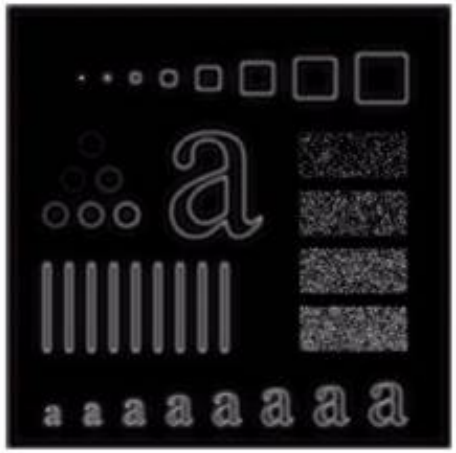
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Butterworth High Pass Filters (cont…)



Results of Butterworth high pass filtering of order 2 with D_0 = 15





Results of Butterworth high pass filtering of order 2 with D_0 = 80

Results of Butterworth high pass filtering of order 2 with D_0 = 30

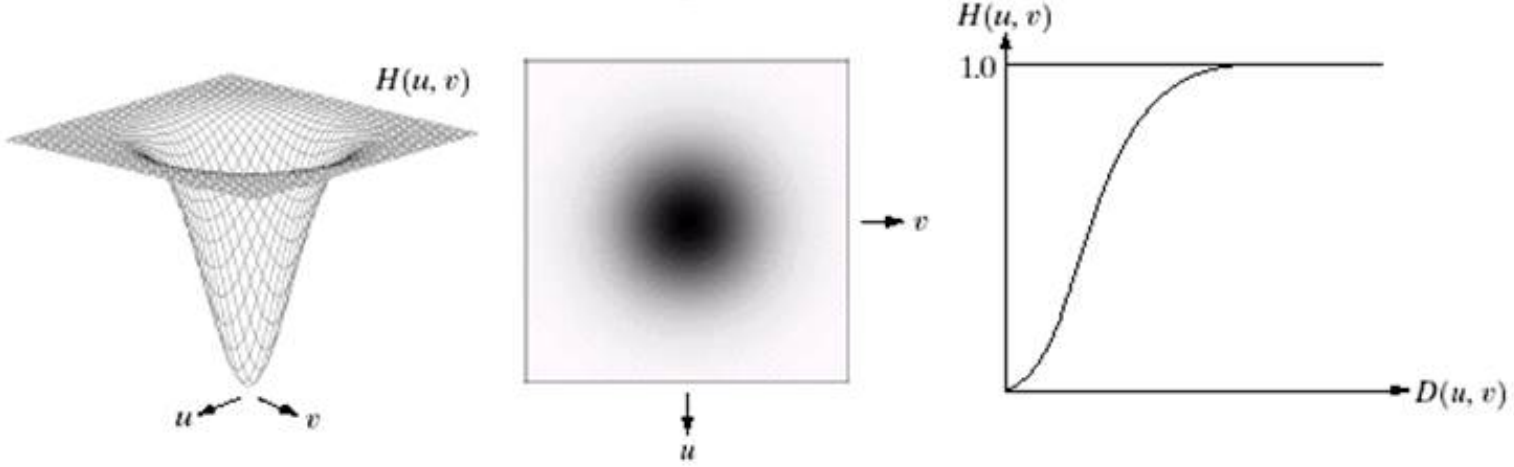
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Gaussian High Pass Filters

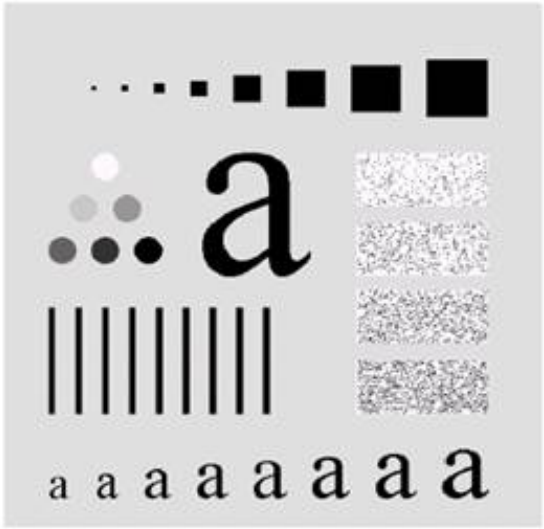
The Gaussian high pass filter is given as:

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

where D_0 is the cut off distance as before



Gaussian High Pass Filters (cont…)



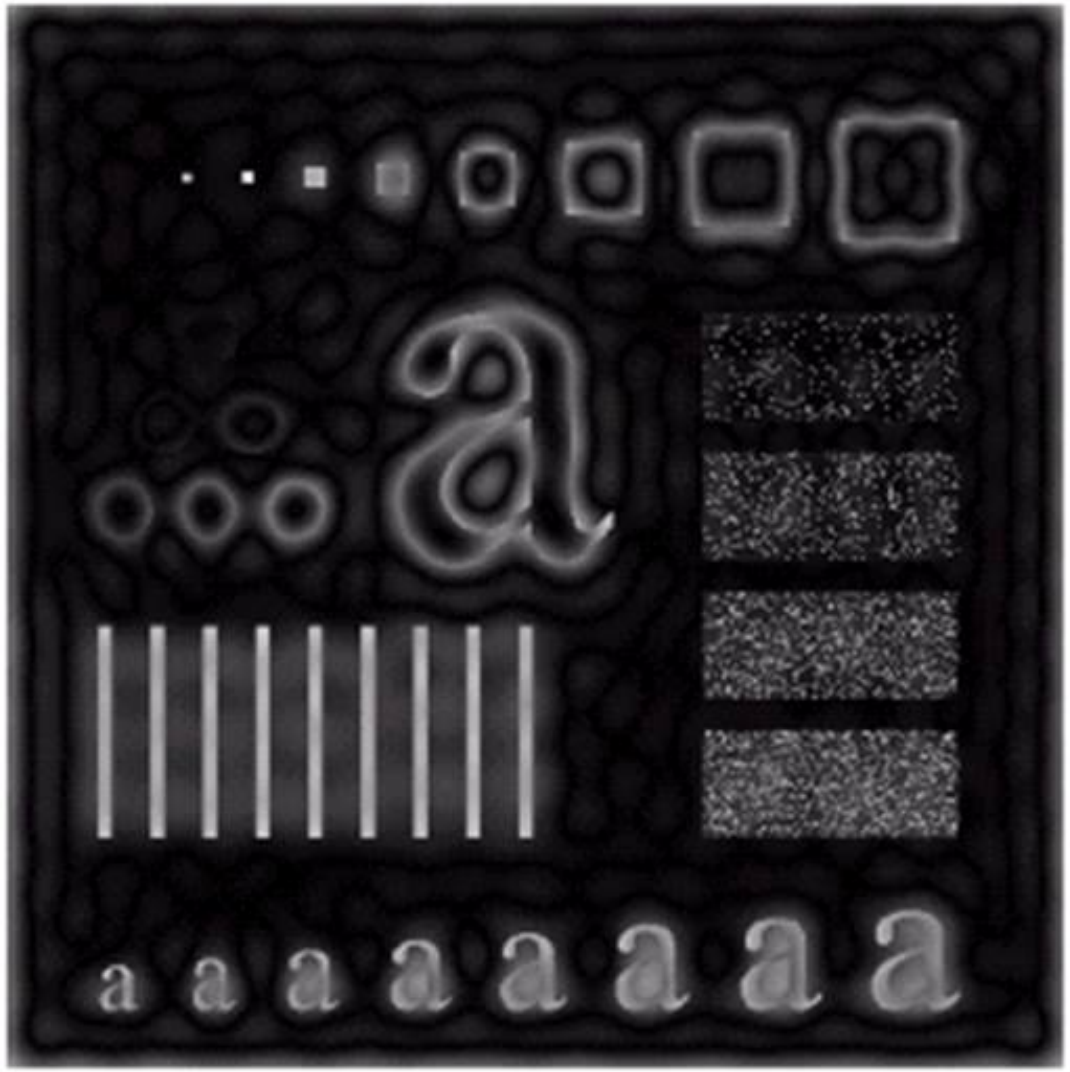
Results of Gaussian high pass filtering with D_0 = 15

Results of Gaussian high pass filtering with D_0 = 80

Results of Gaussian high pass filtering with D_0 = 30

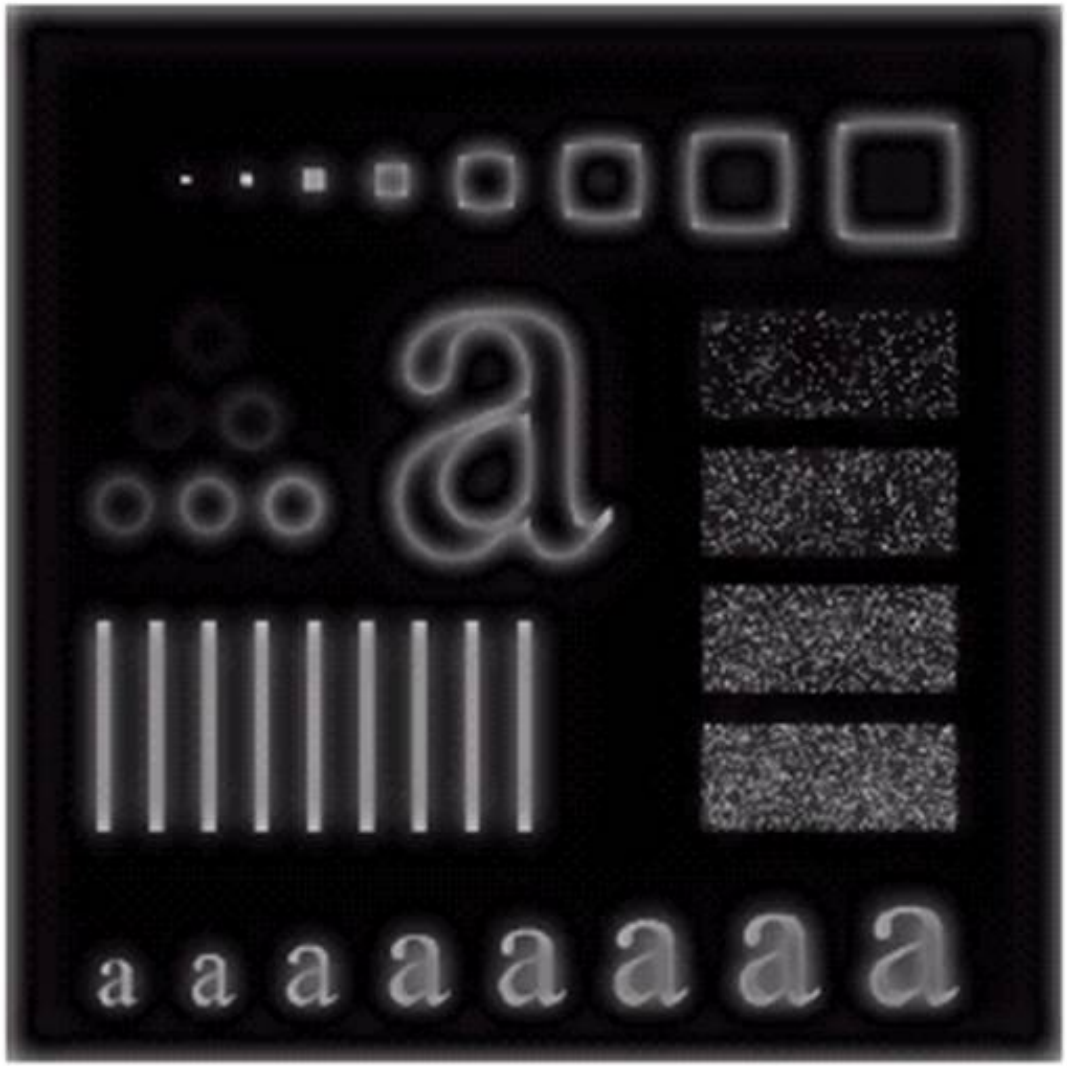
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Highpass Filter Comparison



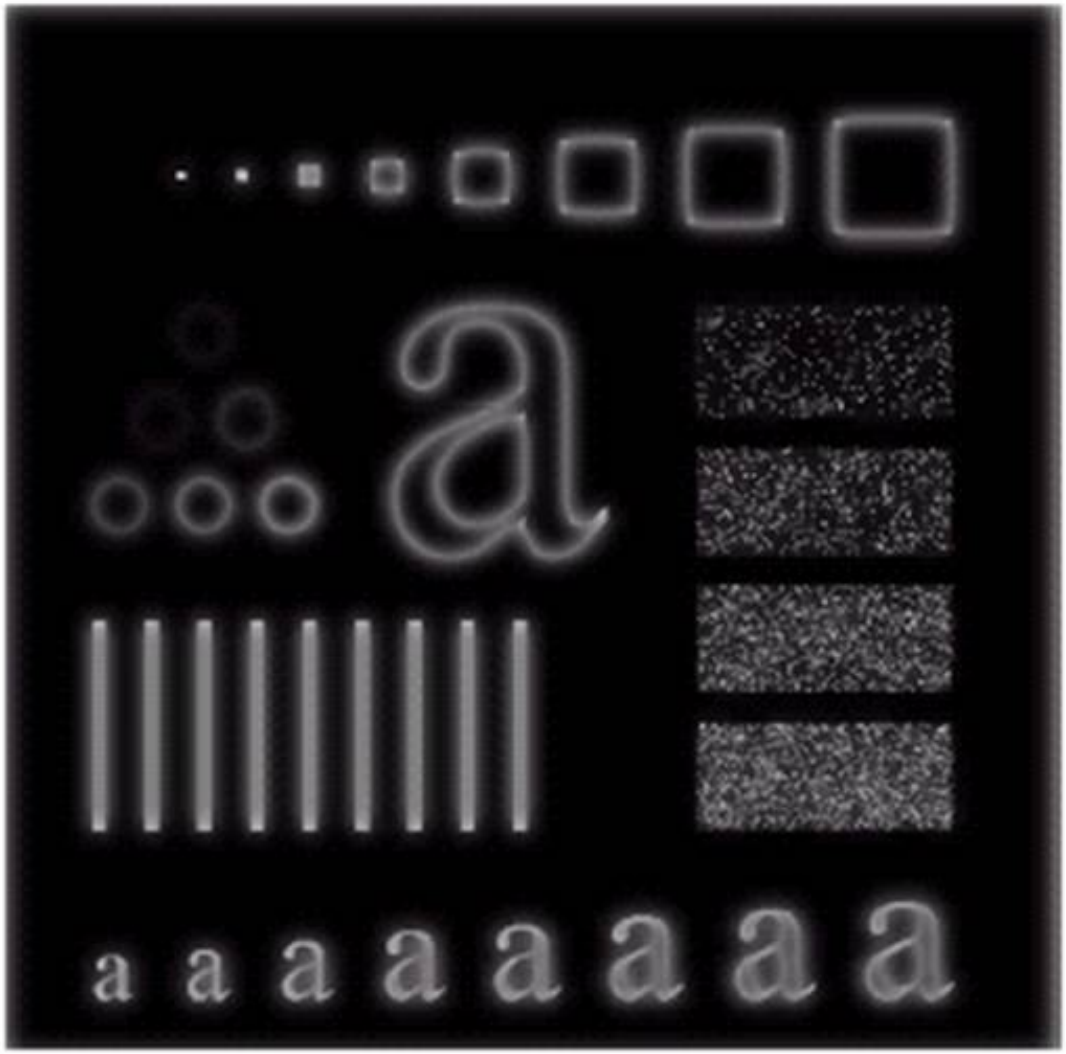
Results of ideal high pass filtering with D_0 = 15

Highpass Filter Comparison



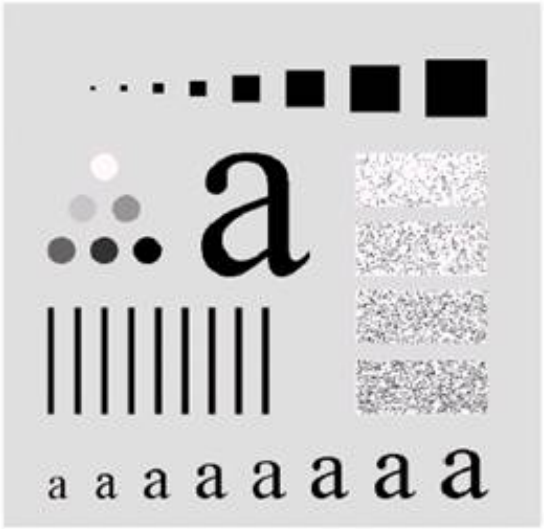
Results of Butterworth high pass filtering of order 2 with D_0 = 15

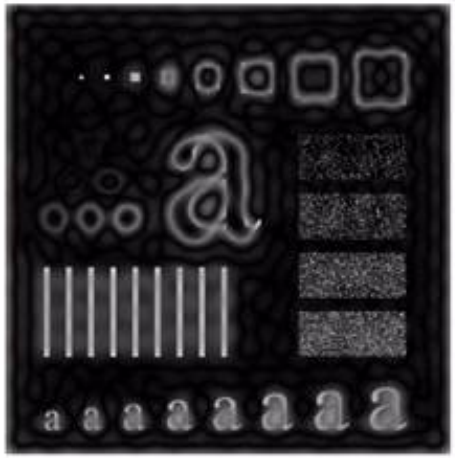
Highpass Filter Comparison



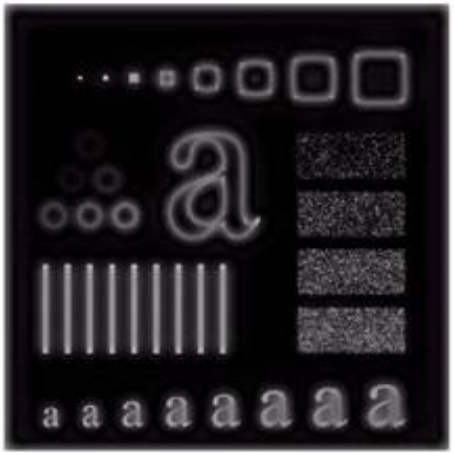
Results of Gaussian high pass filtering with D_0 = 15

Highpass Filter Comparison

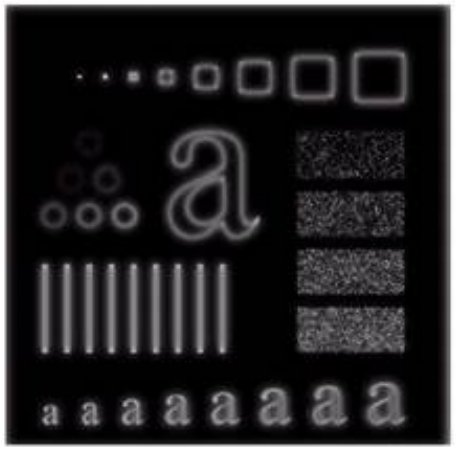




Results of ideal high pass filtering with D_0 = 15



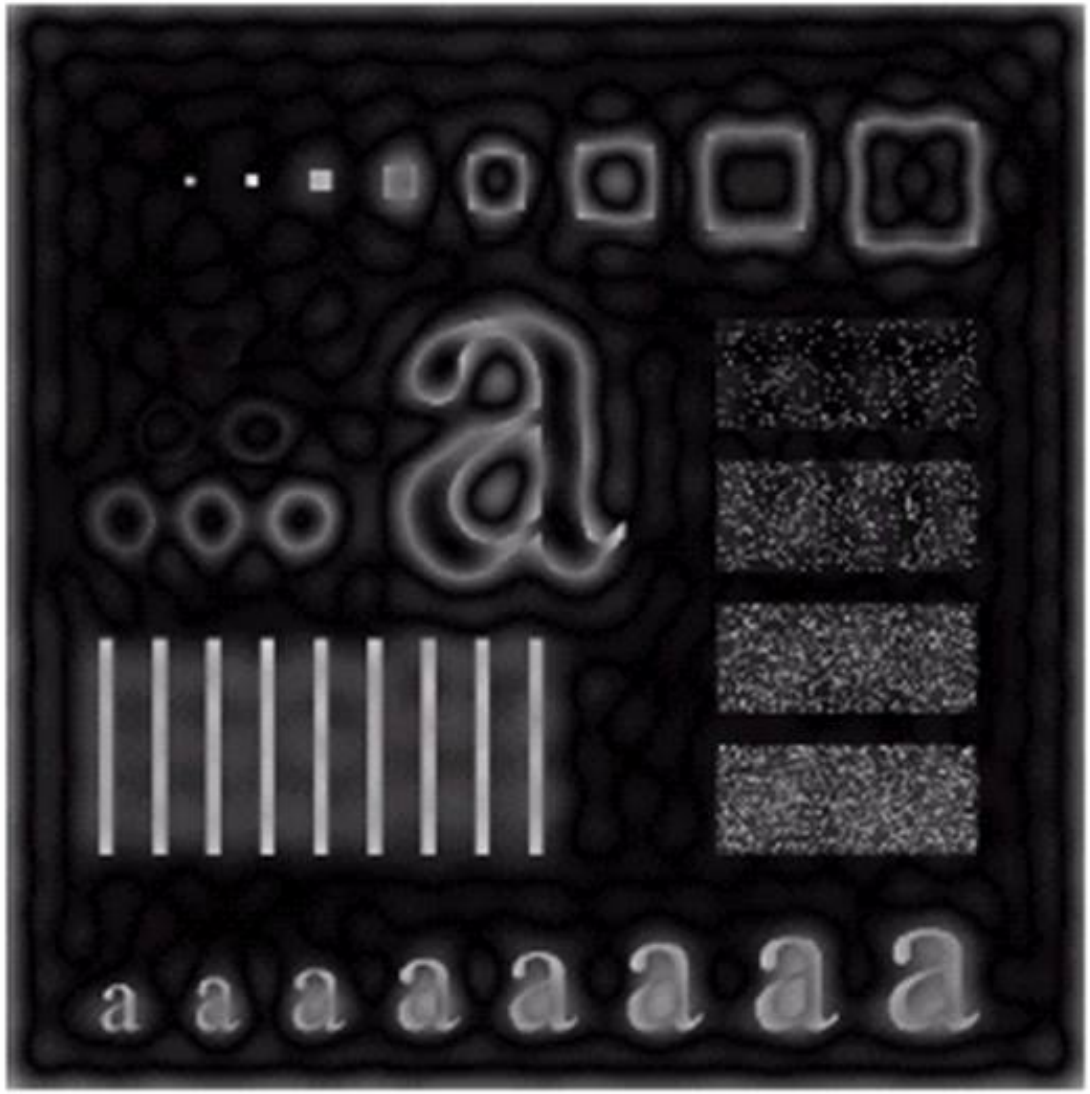
Results of Butterworth high pass filtering of order 2 with D_0 = 15



Results of Gaussian high pass filtering with D_0 = 15

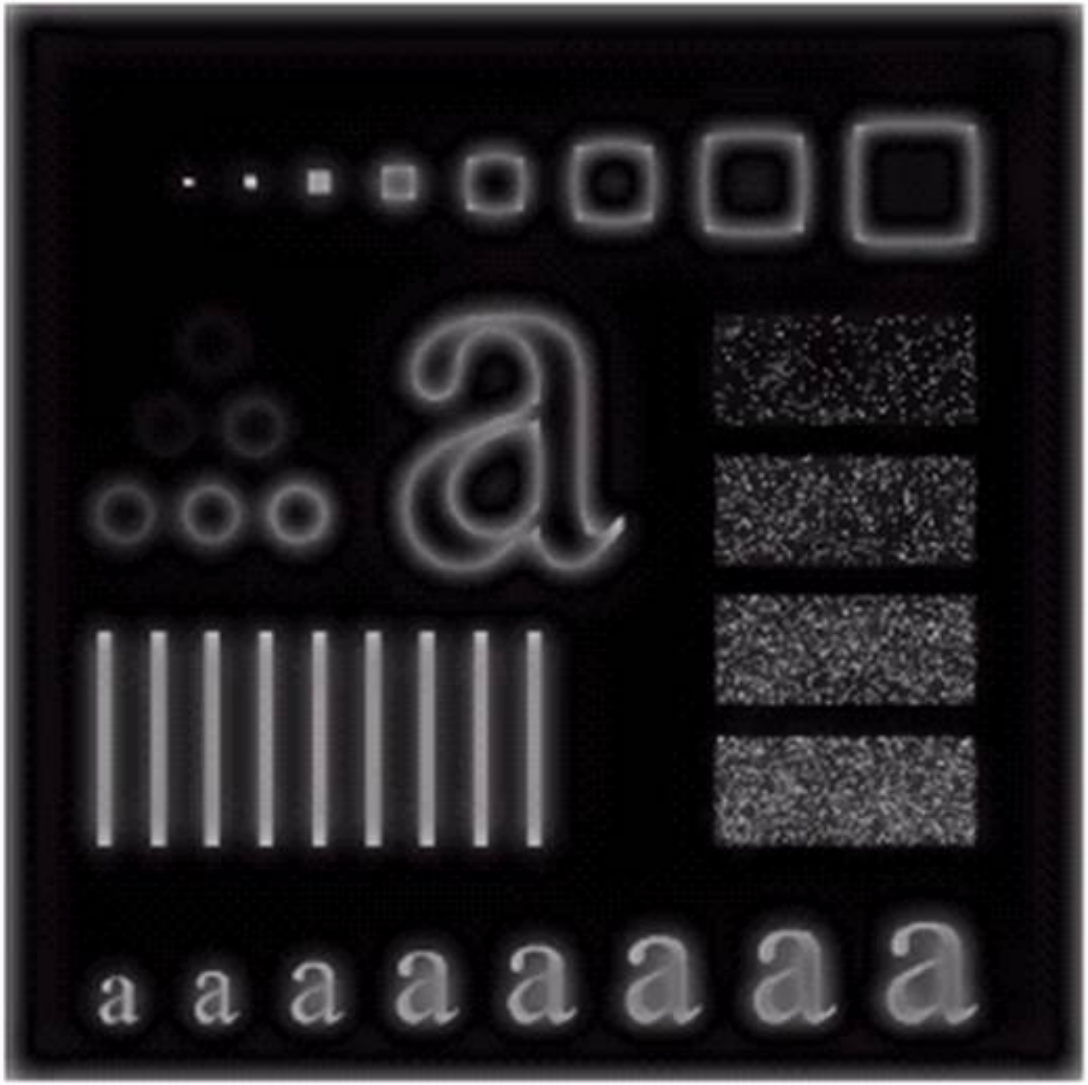
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Highpass Filter Comparison



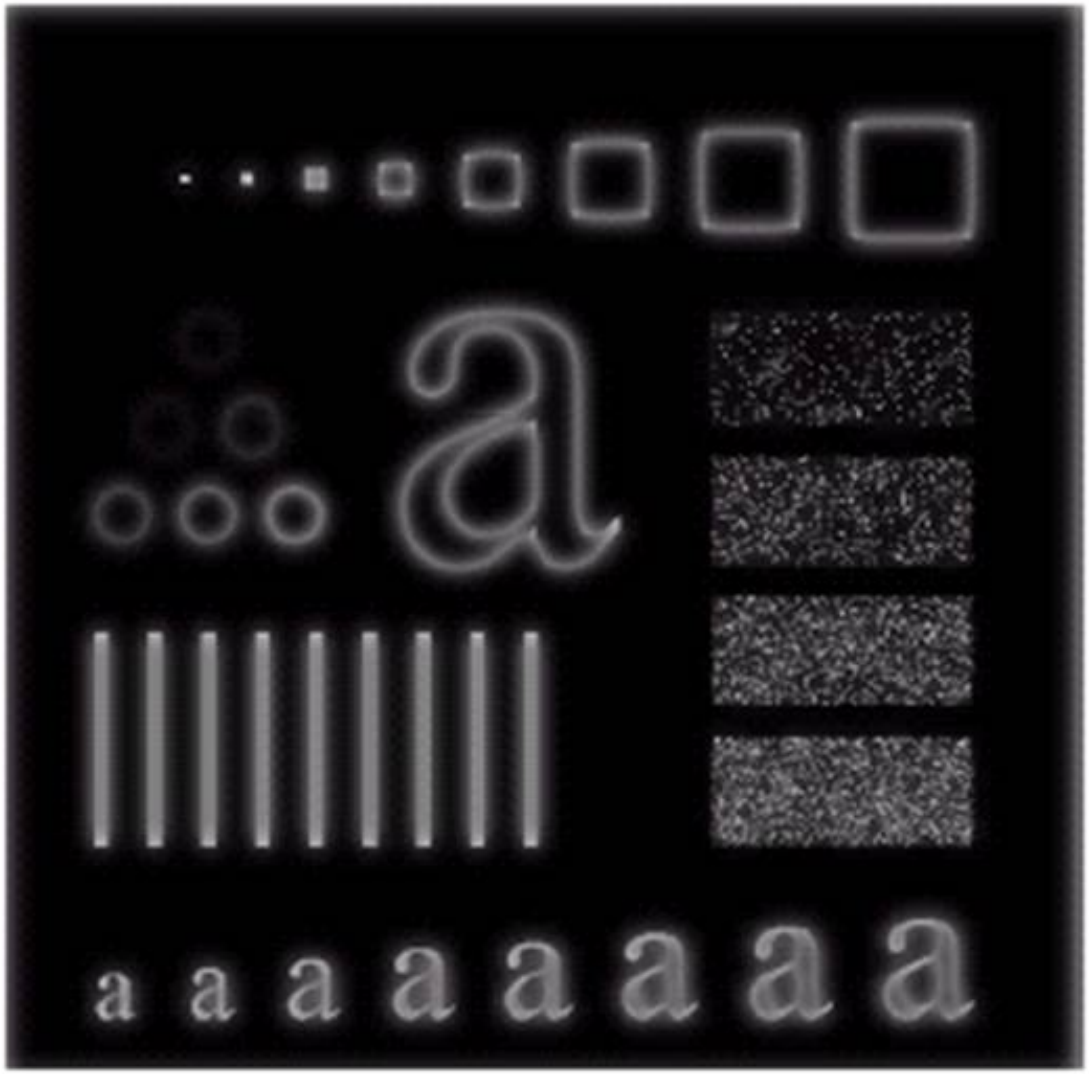
Results of ideal high pass filtering with D_0 = 15

Highpass Filter Comparison



Results of Butterworth high pass filtering of order 2 with D_0 = 15

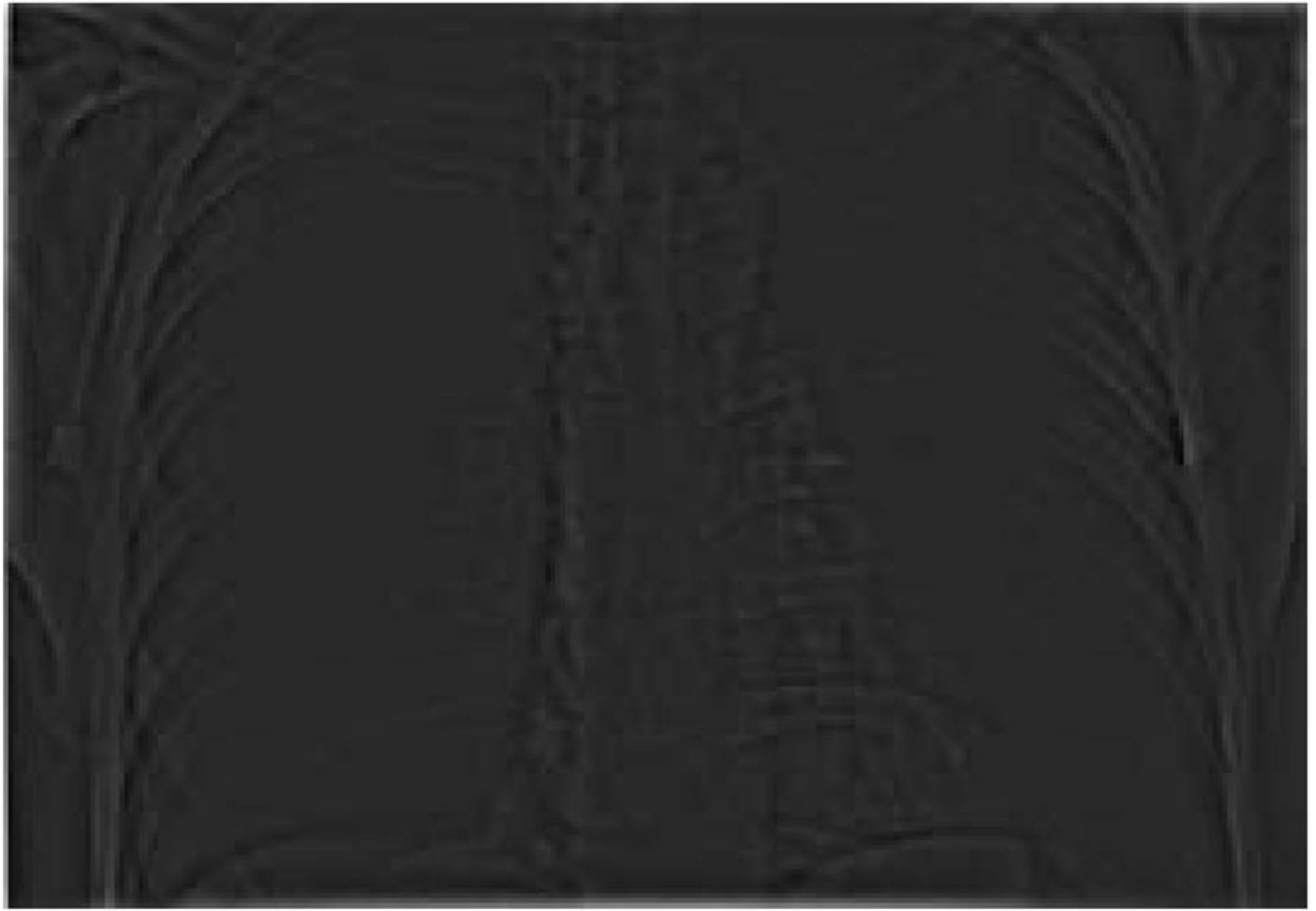
Highpass Filter Comparison

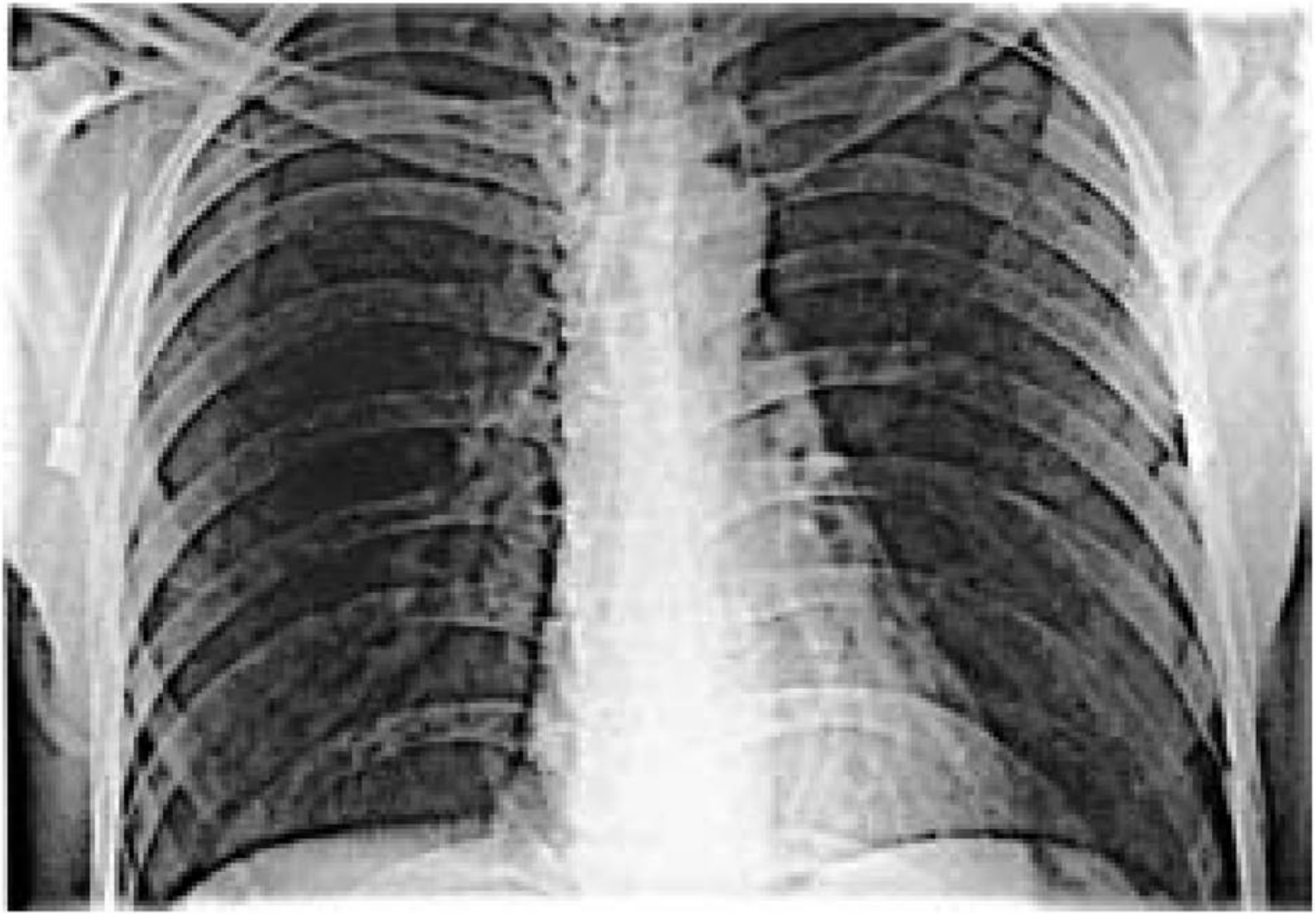


Results of Gaussian high pass filtering with D_0 = 15

Highpass Filtering Example

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

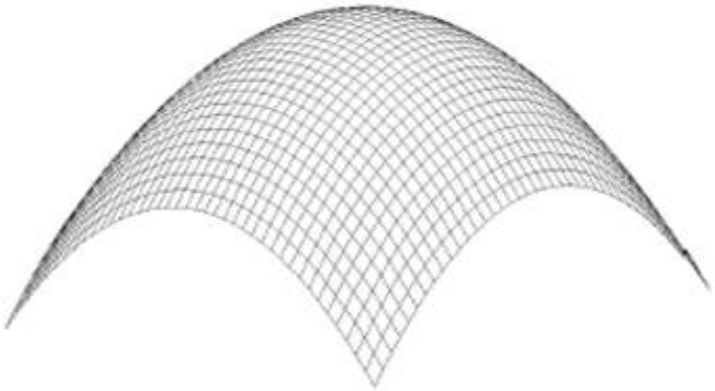




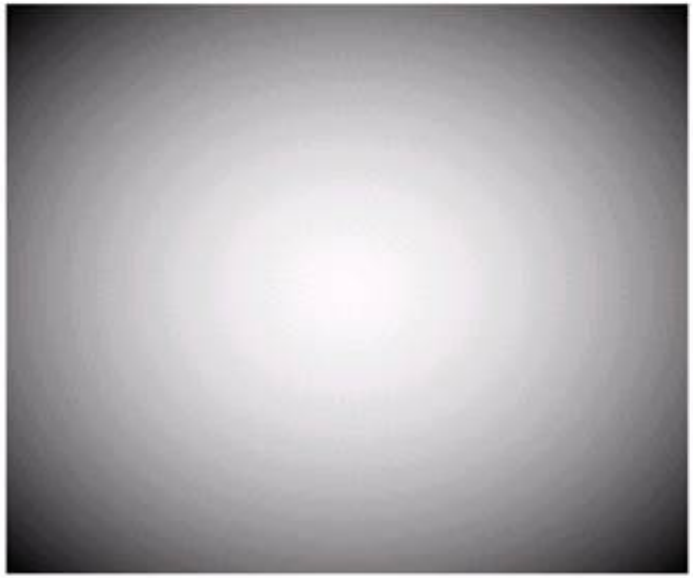
Laplacian In The Frequency Domain

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

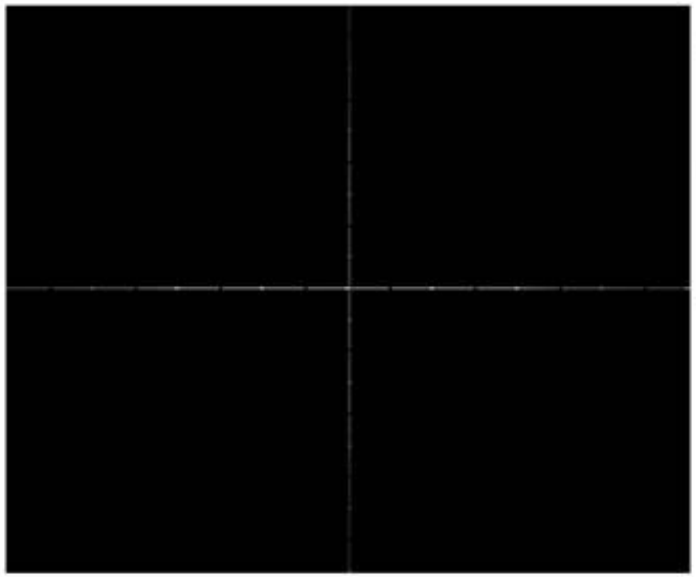
Laplacian in the frequency domain

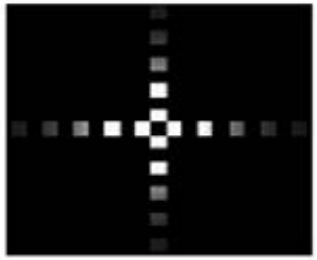


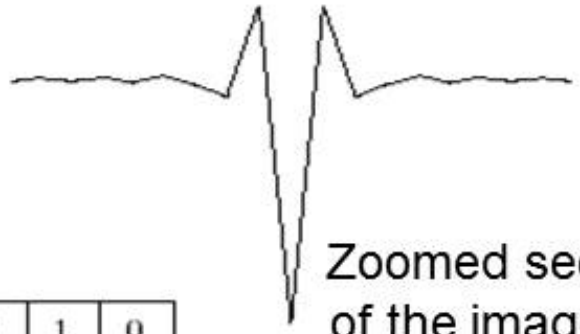
2-D image of Laplacian in the frequency domain



Inverse DFT of Laplacian in the frequency domain



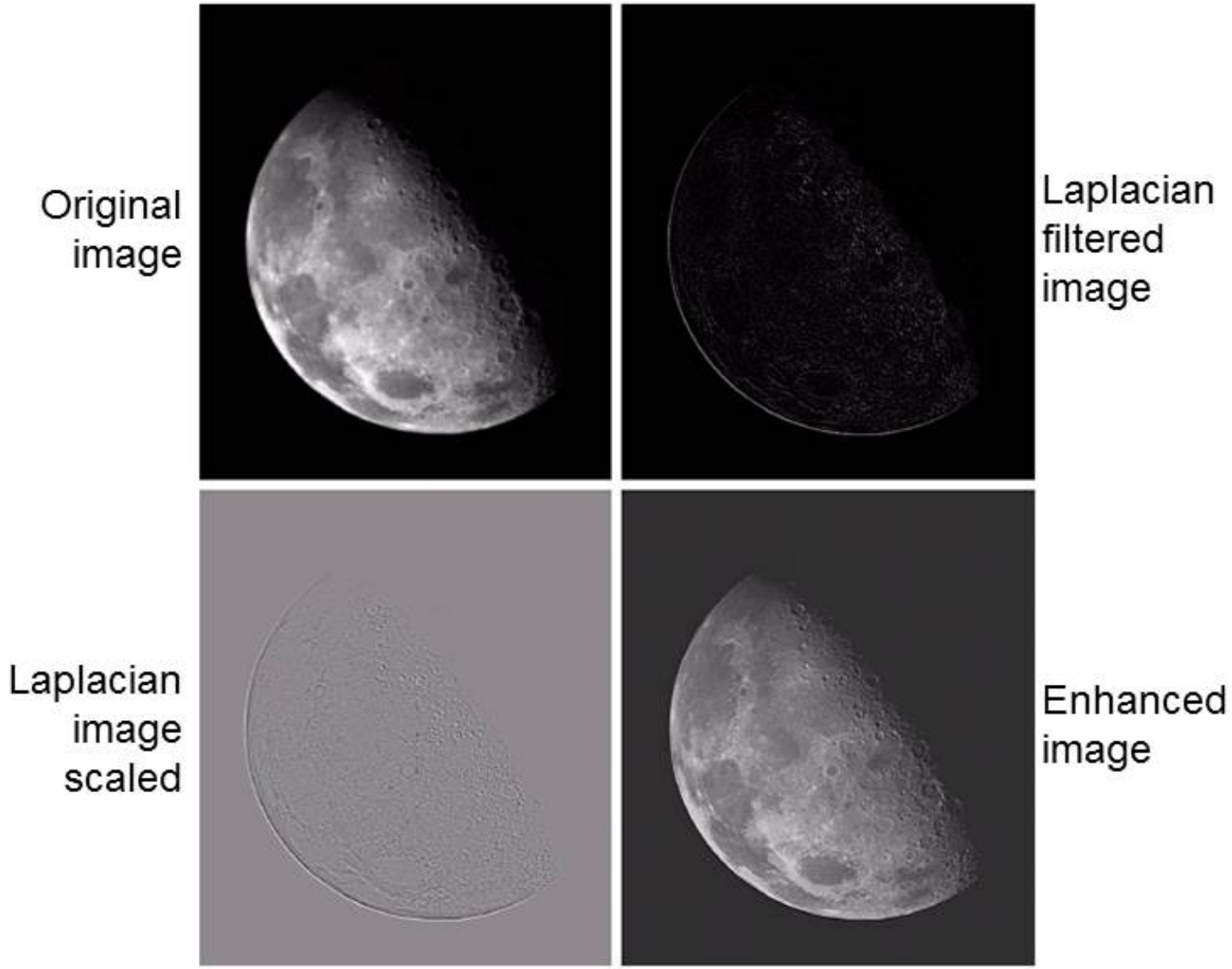




Zoomed section of the image on the left compared to spatial filter

0	1	0
1	−4	1
0	1	0

Frequency Domain Laplacian Example



Fast Fourier Transform

The reason that Fourier based techniques have become so popular is the development of the *Fast Fourier Transform (FFT)* algorithm

Allows the Fourier transform to be carried out in a reasonable amount of time

Reduces the amount of time required to perform a Fourier transform by a factor of 100 – 600 times!

Frequency Domain Filtering & Spatial Domain Filtering

Similar jobs can be done in the spatial and frequency domains

Filtering in the spatial domain can be easier to understand

Filtering in the frequency domain can be much faster – especially for large images

Summary

In this lecture we examined image enhancement in the frequency domain

- The Fourier series & the Fourier transform
- Image Processing in the frequency domain
  - Image smoothing
  - Image sharpening
- Fast Fourier Transform

Next time we will begin to examine image restoration using the spatial and frequency based techniques we have been looking at

Imteresting Application Of Frequency Domain Filtering

Imteresting Application Of Frequency Domain Filtering

Contents

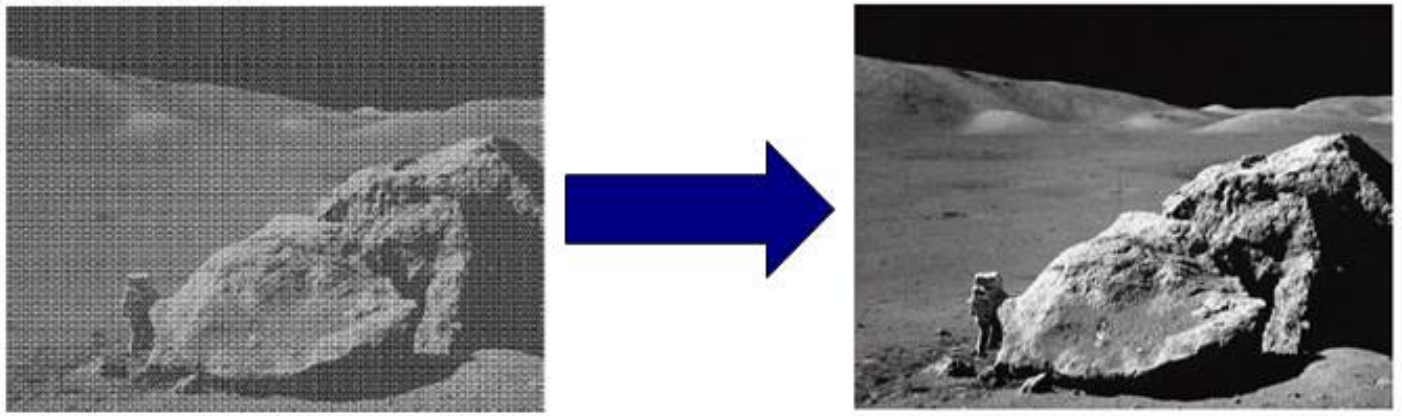
In this lecture we will look at image restoration techniques used for noise removal

- What is image restoration?
- Noise and images
- Noise models
- Noise removal using spatial domain filtering
- Periodic noise
- Noise removal using frequency domain filtering

What is Image Restoration?

Image restoration attempts to restore images that have been degraded

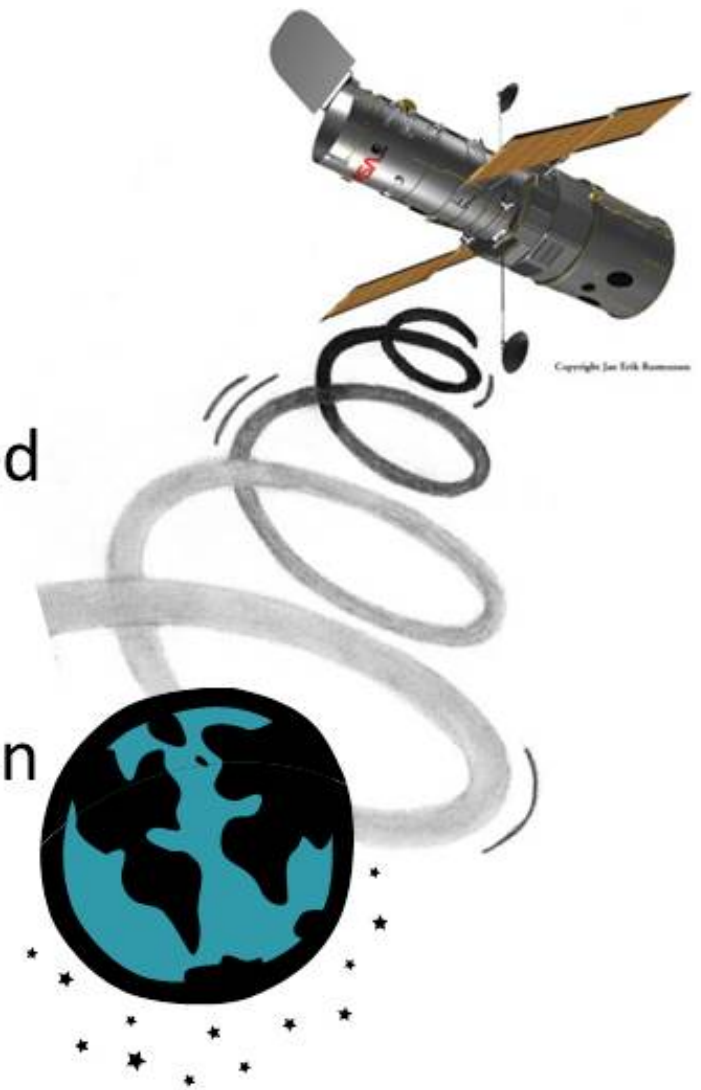
- Identify the degradation process and attempt to reverse it
- Similar to image enhancement, but more objective



Noise and Images

The sources of noise in digital images arise during image acquisition (digitization) and transmission

- Imaging sensors can be affected by ambient conditions
- Interference can be added to an image during transmission



Noise Model

We can consider a noisy image to be modelled as follows:

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

where $f(x, y)$ is the original image pixel, $\eta(x, y)$ is the noise term and $g(x, y)$ is the resulting noisy pixel

If we can estimate the model the noise in an image is based on this will help us to figure out how to restore the image

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	51	52	52	56	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
148	154	157	160	163	167	170
151	155	159	162	165	169	172

x

y

Image f (x, y)

Noisy Image

x

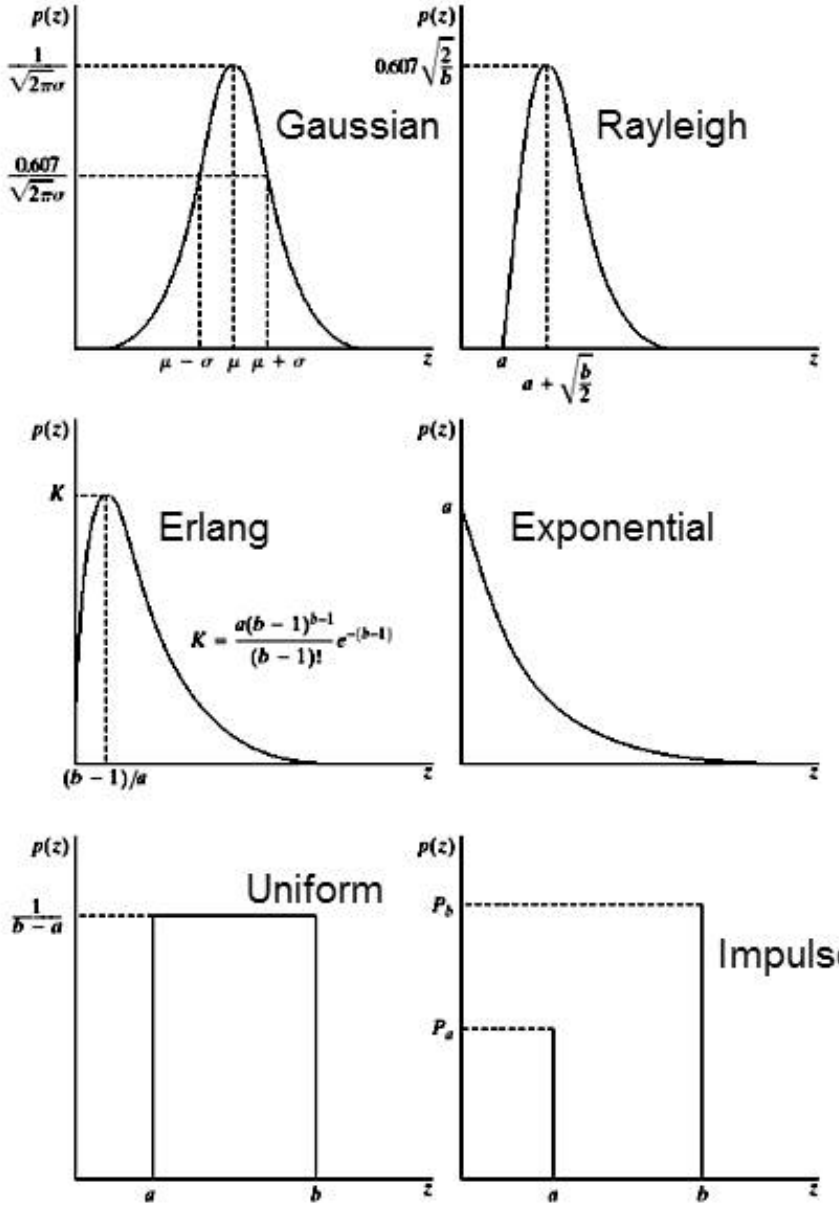
y

Image f (x, y)

Noise Models

There are many different models for the image noise term $\eta(x, y)$:

- Gaussian
  - Most common model
- Rayleigh
- Erlang
- Exponential
- Uniform
- Impulse
  - *Salt and pepper* noise



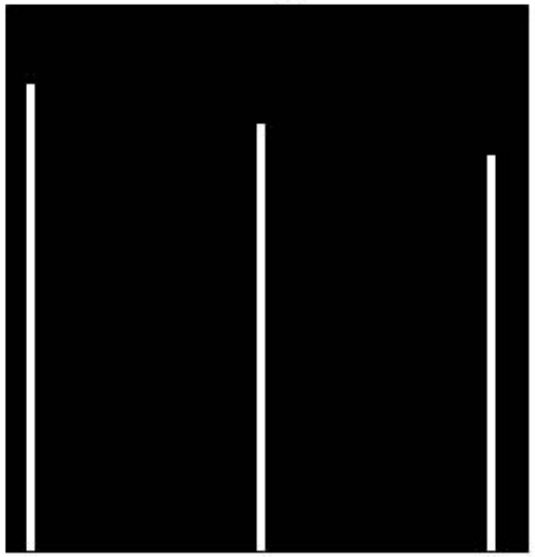
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Noise Example

The test pattern to the right is ideal for demonstrating the addition of noise

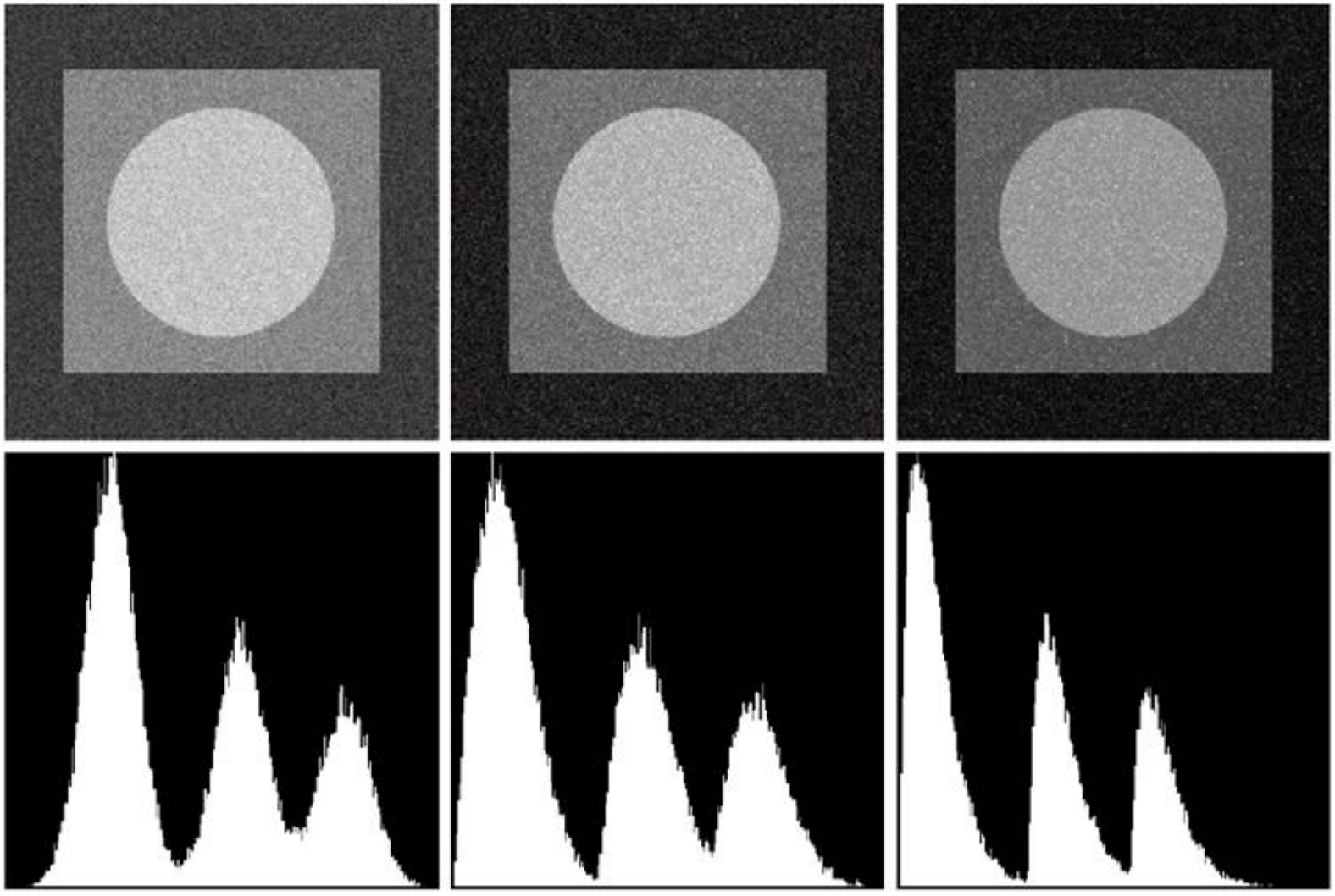
The following slides will show the result of adding noise based on various models to this image

Image



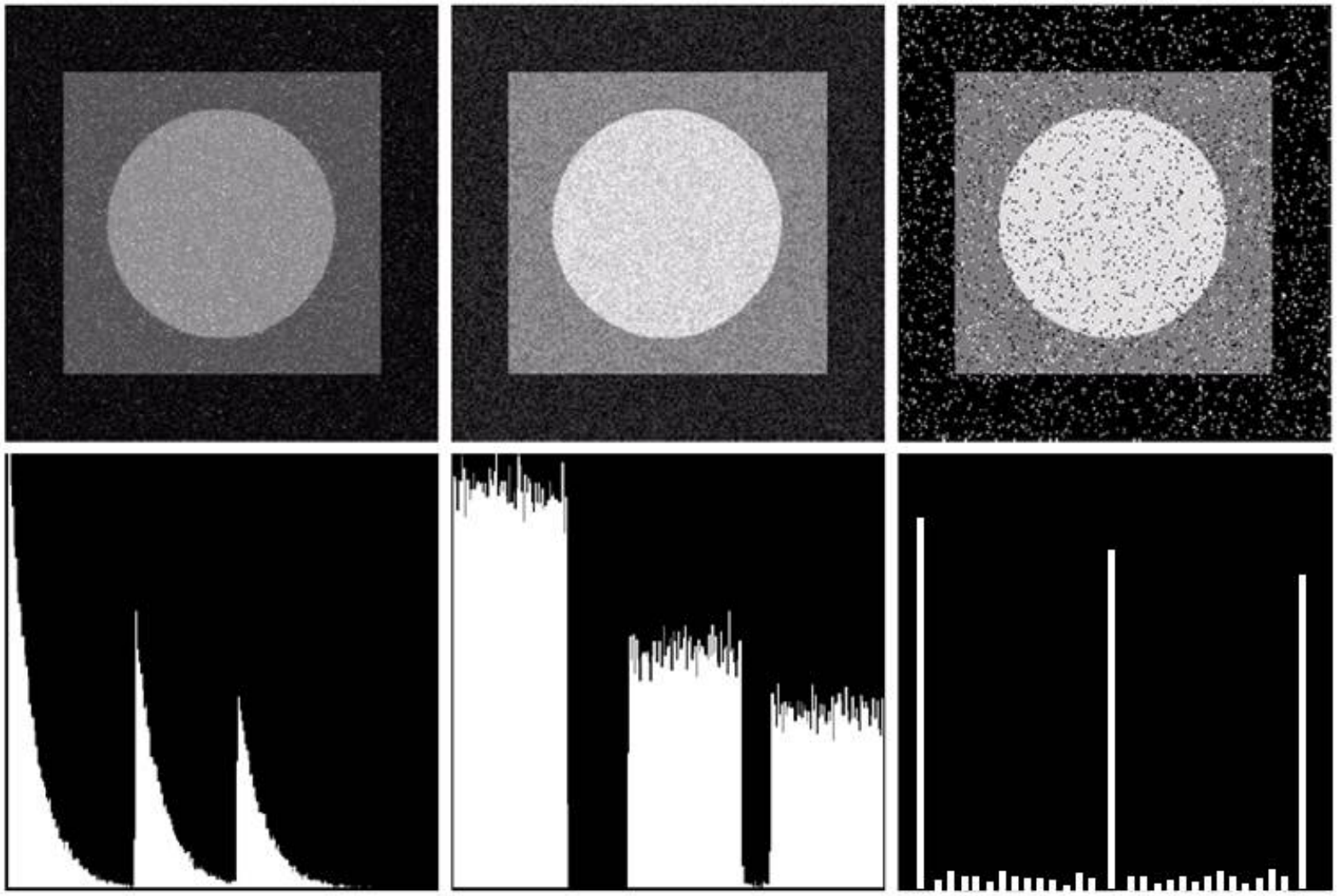
Histogram

Noise Example (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Noise Example (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Filtering to Remove Noise

We can use spatial filters of different kinds to remove different kinds of noise

The *arithmetic mean* filter is a very simple one and is calculated as follows:

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t)\in S_{xy}} g(s,t)$$

This is implemented as the simple smoothing filter

Blurs the image to remove noise

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Noise Removal Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
148	154	157	160	163	167	170
151	155	159	162	165	169	172

x

y

Image f (x, y)

Filtered Image

x

y

Image f (x, y)

Other Means

There are different kinds of mean filters all of which exhibit slightly different behaviour:

- Geometric Mean
- Harmonic Mean
- Contraharmonic Mean

Oth  Means (cont...)

There are other variants on the mean which can give different performance

Geometric Mean:

$$\hat{f}(x,y) = \left[\prod_{(s,t)\in S_{xy}} g(s,t) \right]^{\frac{1}{mn}}$$

Achieves similar smoothing to the arithmetic mean, but tends to lose less image detail

Noise Removal Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
148	154	157	160	163	167	170
151	155	159	162	165	169	172

x

y

Image f (x, y)

Filtered Image

x

y

Image f (x, y)

Oth  Means (cont...)

Harmonic Mean:

$$\hat{f}(x,y) = \frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

Works well for salt noise, but fails for pepper noise

Also does well for other kinds of noise such as Gaussian noise

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

x

y

Image f (x, y)

Filtered Image

x

y

Image f (x, y)

Oth  Means (cont...)

Contraharmonic Mean:

$$\hat{f}(x,y) = \frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

Q is the *order* of the filter and adjusting its value changes the filter's behaviour

Positive values of Q eliminate pepper noise

Negative values of Q eliminate salt noise

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

x

y

Image f (x, y)

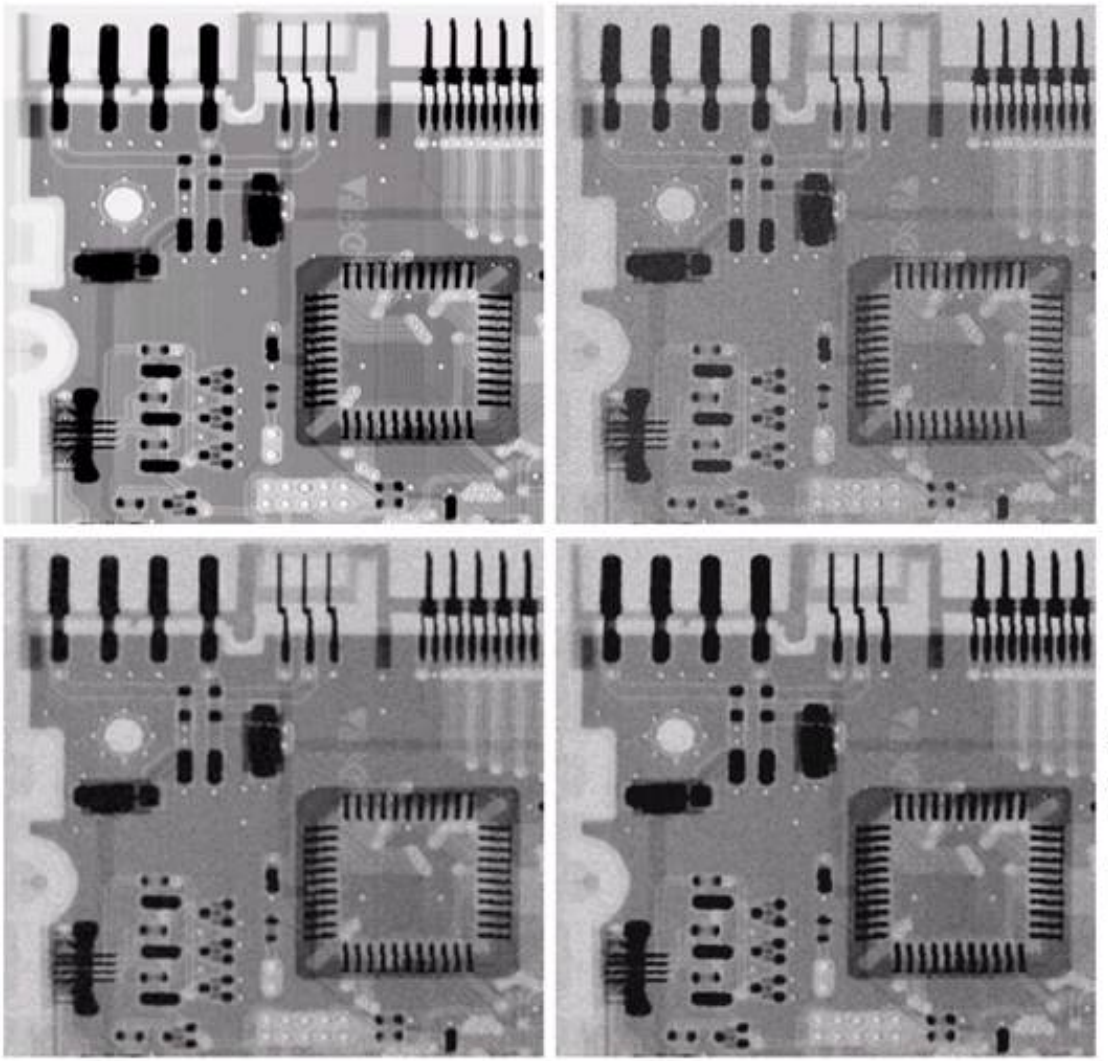
Filtered Image

x

y

Image f (x, y)

Noise Removal Examples



Original Image

Image Corrupted By Gaussian Noise

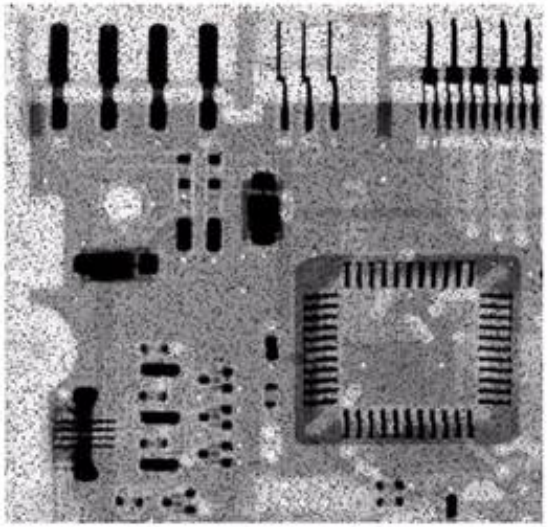
After A 3*3 Arithmetic Mean Filter

After A 3*3 Geometric Mean Filter

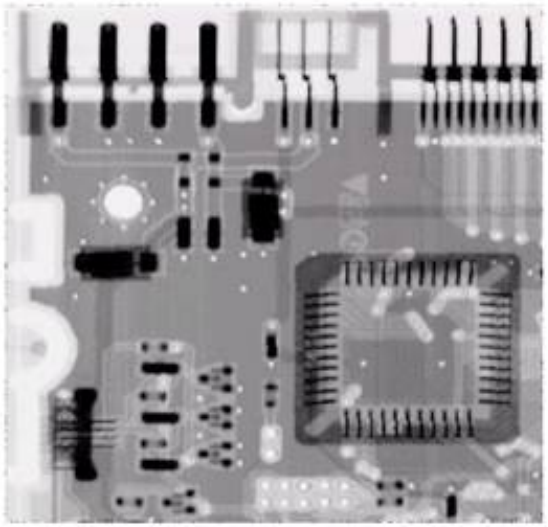
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Noise Removal Examples (cont…)

Image Corrupted By Pepper Noise



Result of Filtering Above With 3*3 Contraharmonic Q=1.5



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Noise Removal Examples (cont…)

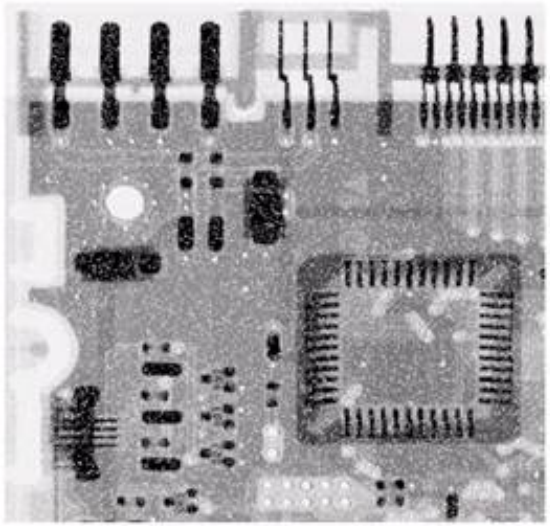
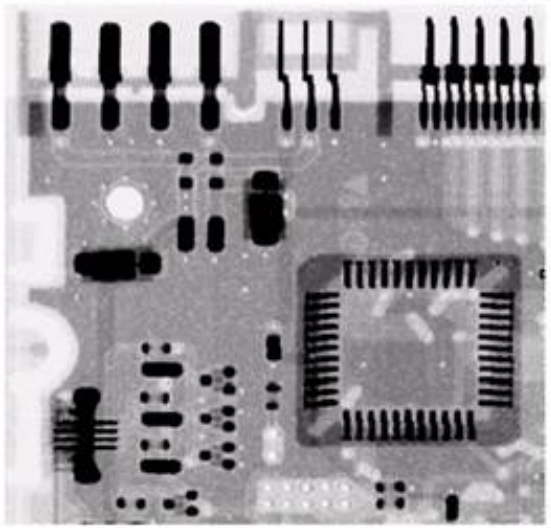


Image Corrupted By Salt Noise

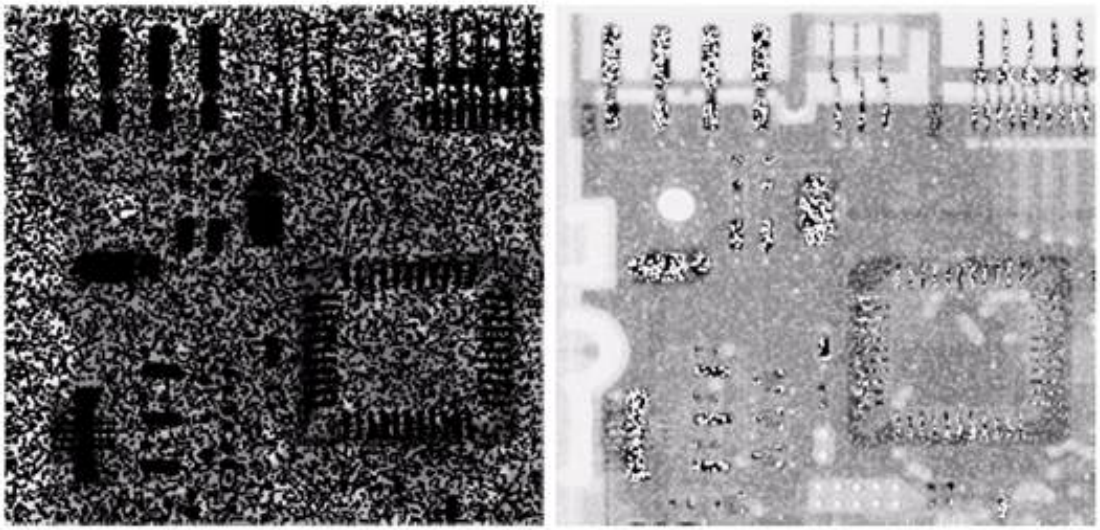


Result of Filtering Above With 3*3 Contraharmonic Q=-1.5

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Contraharmonic Filter: Here Be Dragons

Choosing the wrong value for Q when using the contraharmonic filter can have drastic results



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Order Statistics Filters

Spatial filters that are based on ordering the pixel values that make up the neighbourhood operated on by the filter

Useful spatial filters include

- Median filter
- Max and min filter
- Midpoint filter
- Alpha trimmed mean filter

Median Filter

Median Filter:

$$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

Excellent at noise removal, without the smoothing effects that can occur with other smoothing filters

Particularly good when salt and pepper noise is present

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

x

y

Image f (x, y)

Filtered Image

x

y

Image f (x, y)

Max and Min Filter

Max Filter:

$$\hat{f}(x,y) = \max_{(s,t)\in S_{xy}} \{g(s,t)\}$$

Min Filter:

$$\hat{f}(x,y) = \min_{(s,t)\in S_{xy}} \{g(s,t)\}$$

Max filter is good for pepper noise and min is good for salt noise

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

x

y

Image f (x, y)

Filtered Image

x

y

Image f (x, y)

Midpoint Filter

Midpoint Filter:

$$\hat{f}(x,y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}}\{g(s,t)\} + \min_{(s,t)\in S_{xy}}\{g(s,t)\}\right]$$

Good for random Gaussian and uniform noise

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

x

y

Image f (x, y)

Filtered Image

x

y

Image f (x, y)

Alpha-Trimmed Mean Filter

Alpha-Trimmed Mean Filter:

$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t)$$

We can delete the $d/2$ lowest and $d/2$ highest grey levels

So $g_r(s, t)$ represents the remaining $mn - d$ pixels

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

x

y

Image f (x, y)

Filtered Image

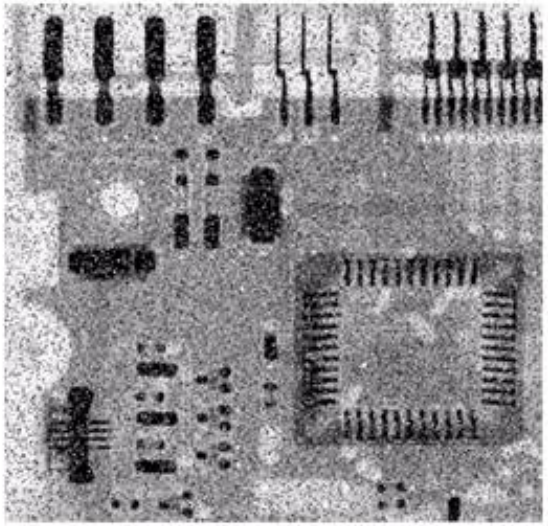
x

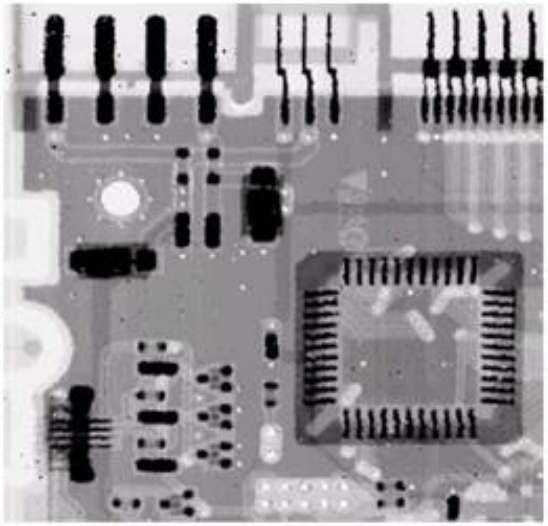
y

Image f (x, y)

Noise Removal Examples

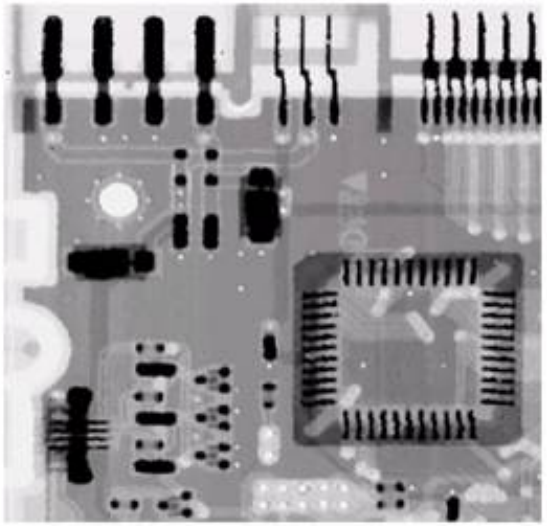
Image Corrupted By Salt And Pepper Noise

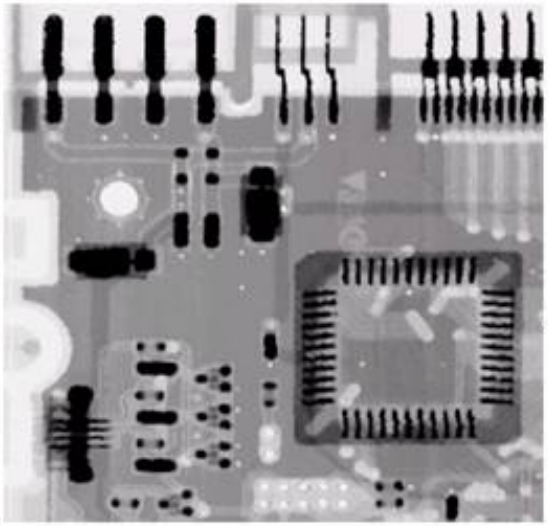




Result of 1 Pass With A 3*3 Median Filter

Result of 2 Passes With A 3*3 Median Filter



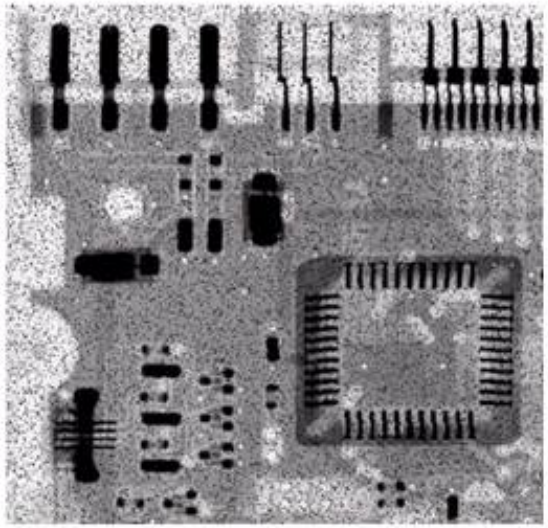


Result of 3 Passes With A 3*3 Median Filter

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Noise Removal Examples (cont…)

Image Corrupted By Pepper Noise



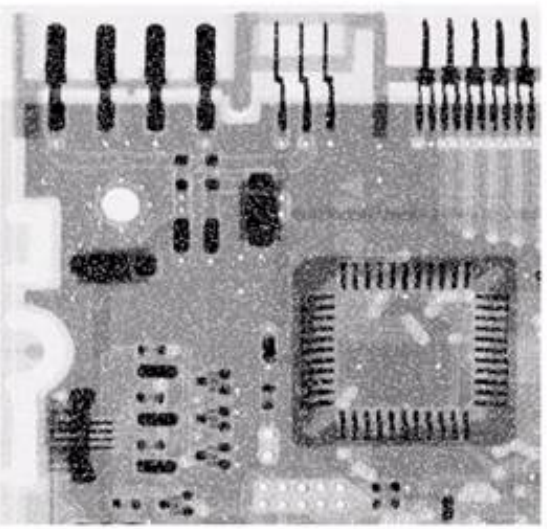
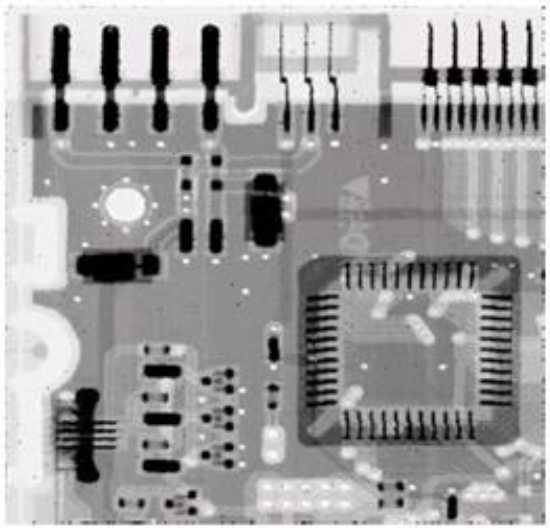
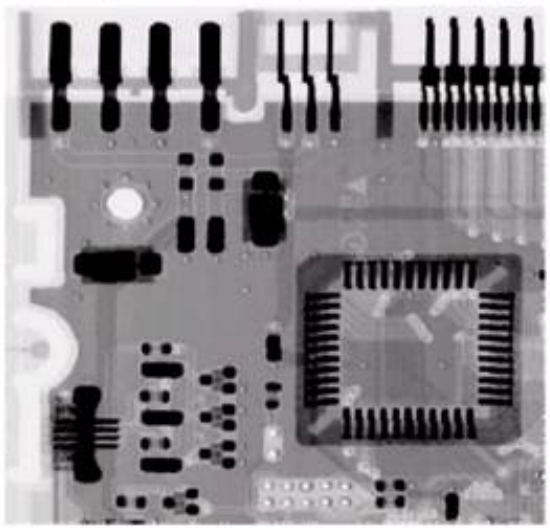


Image Corrupted By Salt Noise

Result Of Filtering Above With A 3*3 Max Filter



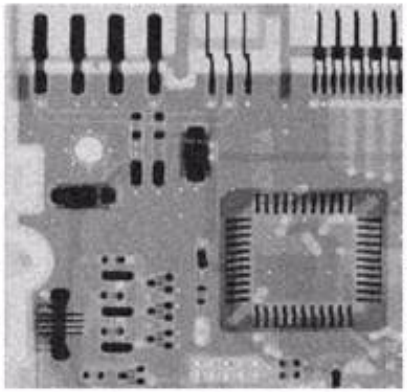


Result Of Filtering Above With A 3*3 Min Filter

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Noise Removal Examples (cont…)

Image Corrupted By Uniform Noise



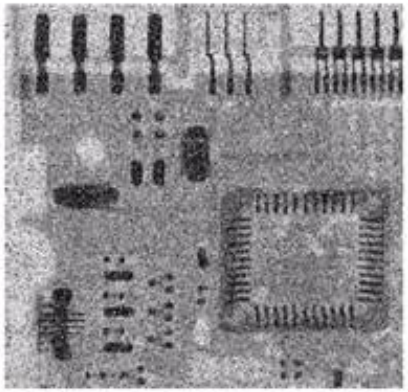
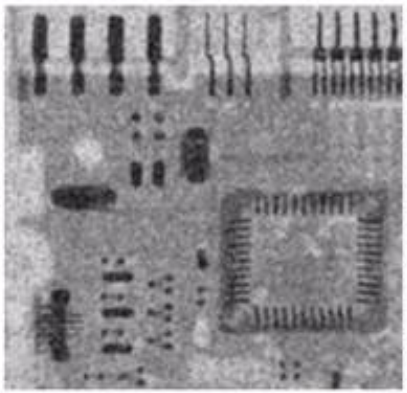
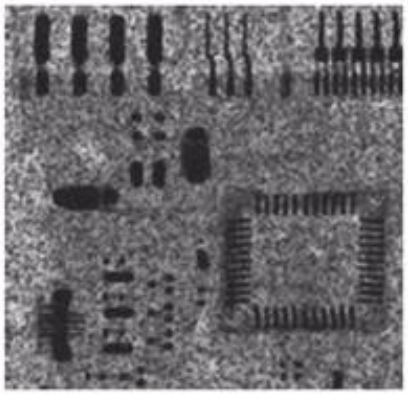


Image Further Corrupted By Salt and Pepper Noise

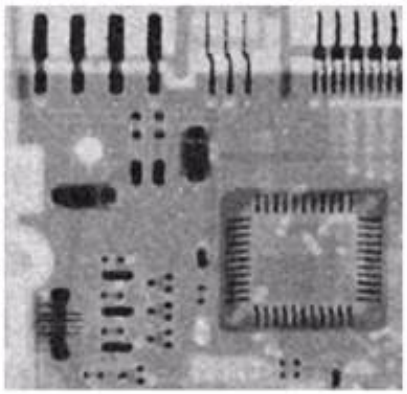
Filtered By 5*5 Arithmetic Mean Filter

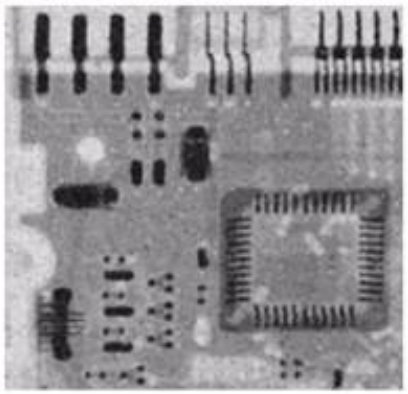




Filtered By 5*5 Geometric Mean Filter

Filtered By 5*5 Median Filter





Filtered By 5*5 Alpha-Trimmed Mean Filter

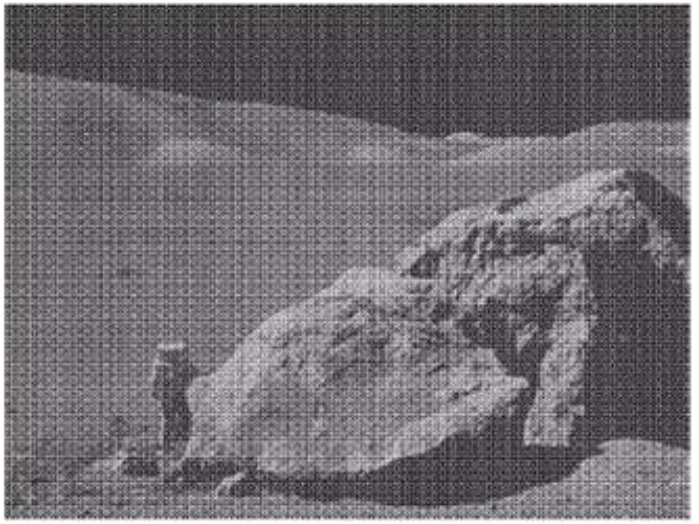
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

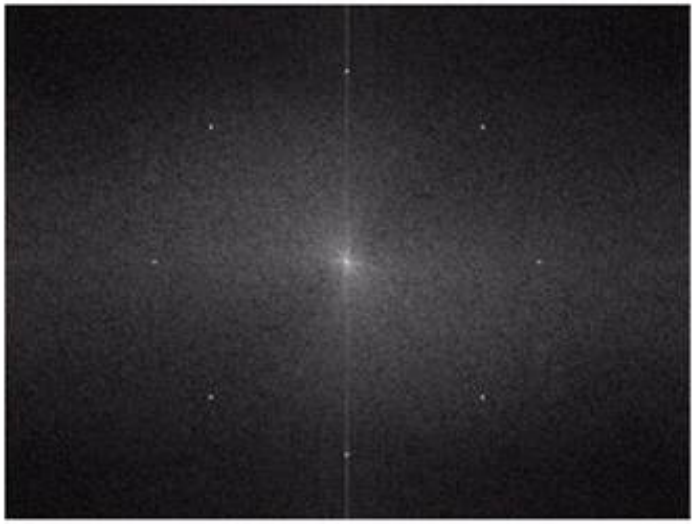
Periodic Noise

Typically arises due to electrical or electromagnetic interference

Gives rise to regular noise patterns in an image

Frequency domain techniques in the Fourier domain are most effective at removing periodic noise





Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Band Reject Filters

Removing periodic noise form an image involves removing a particular range of frequencies from that image

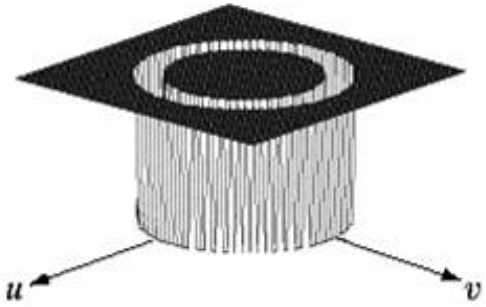
Band reject filters can be used for this purpose

An ideal band reject filter is given as follows:

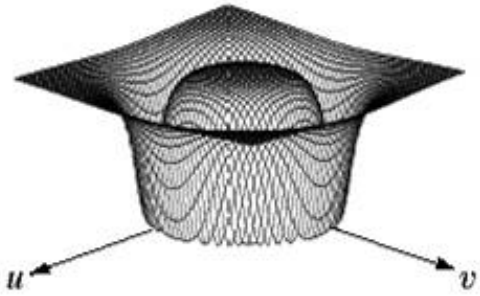
$$H(u,v)=\begin{cases}1 & if\ D(u,v)<D_0-\dfrac{W}{2}\\ 0 & if\ D_0-\dfrac{W}{2}\le D(u,v)\le D_0+\dfrac{W}{2}\\ 1 & if\ D(u,v)>D_0+\dfrac{W}{2}\end{cases}$$

Band Reject Filters (cont…)

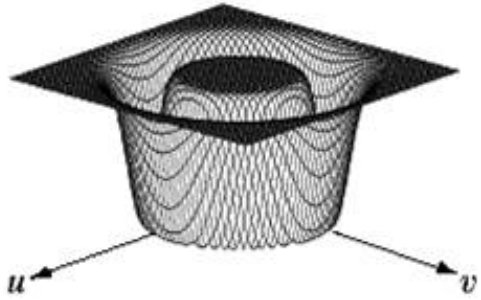
The ideal band reject filter is shown below, along with Butterworth and Gaussian versions of the filter



Ideal Band Reject Filter



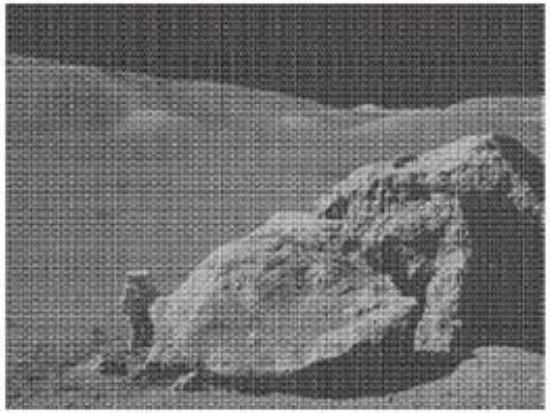
Butterworth Band Reject Filter (of order 1)



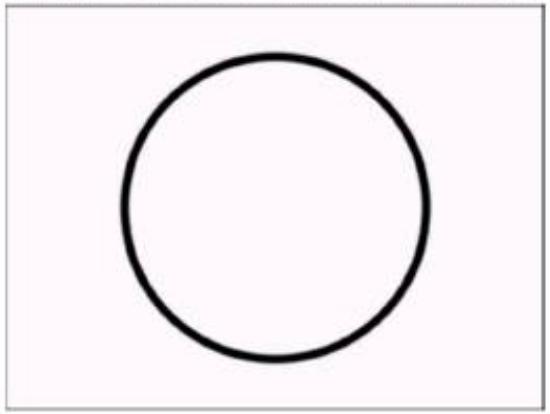
Gaussian Band Reject Filter

Band Reject Filter Example

Image corrupted by sinusoidal noise



Fourier spectrum of corrupted image



Butterworth band reject filter

Filtered image

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Summary

In this lecture we will look at image restoration for noise removal

Restoration is slightly more objective than enhancement

Spatial domain techniques are particularly useful for removing random noise

Frequency domain techniques are particularly useful for removing periodic noise

Adaptive Filters

The filters discussed so far are applied to an entire image without any regard for how image characteristics vary from one point to another

The behaviour of **adaptive filters** changes depending on the characteristics of the image inside the filter region

We will take a look at the **adaptive median filter**

Adaptive Median Filtering

The median filter performs relatively well on impulse noise as long as the spatial density of the impulse noise is not large

The adaptive median filter can handle much more spatially dense impulse noise, and also performs some smoothing for non-impulse noise

The key insight in the adaptive median filter is that the filter size changes depending on the characteristics of the image

Adaptive Median Filtering (cont…)

Remember that filtering looks at each original pixel image in turn and generates a new filtered pixel

First examine the following notation:

- z_{min} = minimum grey level in S_{xy}
- z_{max} = maximum grey level in S_{xy}
- z_{med} = median of grey levels in S_{xy}
- z_{xy} = grey level at coordinates (x, y)
- S_{max} =maximum allowed size of S_{xy}

Adaptive Median Filtering (cont…)

Level A: $A1 = z_{med} - z_{min}$

$A2 = z_{med} - z_{max}$

If $A1 > 0$ and $A2 < 0$, Go to level B

Else increase the window size

If window size $\leq$ repeat S_{max} level A

Else output z_{med}

Level B: $B1 = z_{xy} - z_{min}$

$B2 = z_{xy} - z_{max}$

If $B1 > 0$ and $B2 < 0$, output z_{xy}

Else output z_{med}

Adaptive Median Filtering (cont…)

The key to understanding the algorithm is to remember that the adaptive median filter has three purposes:

- Remove impulse noise
- Provide smoothing of other noise
- Reduce distortion

Adaptive Filtering Example

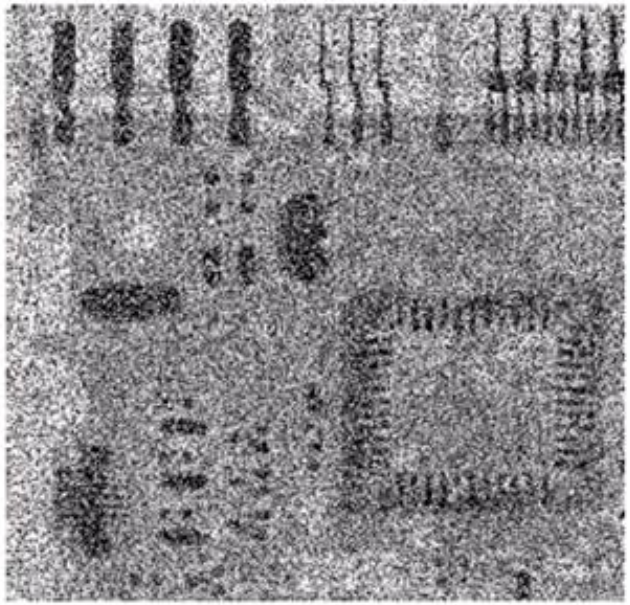
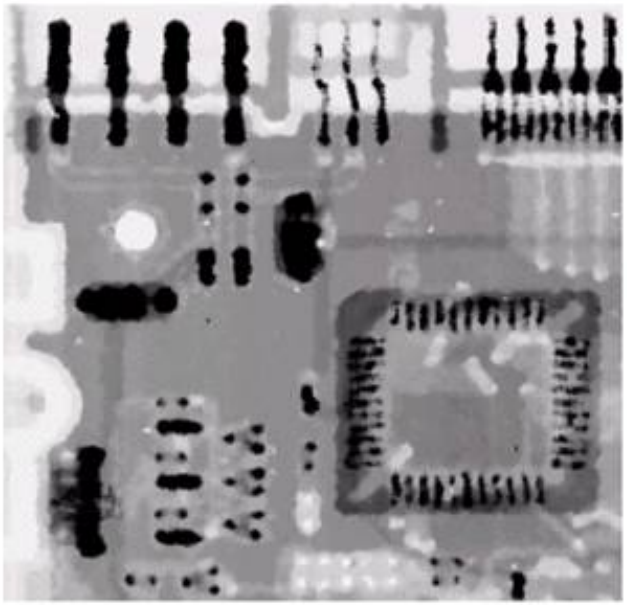
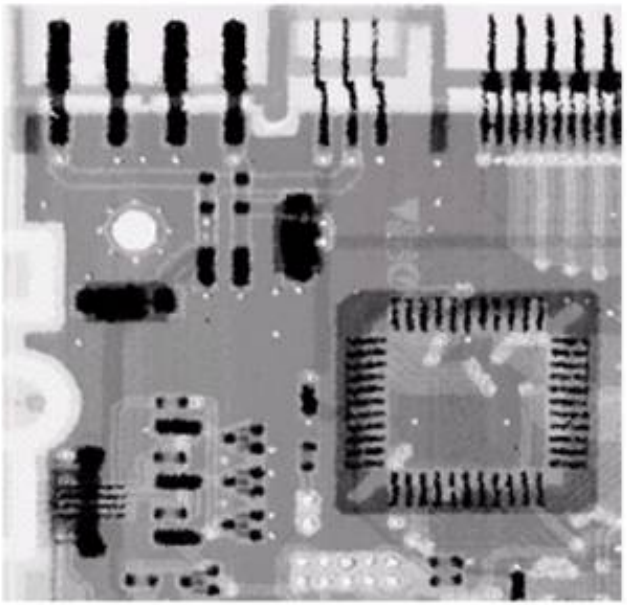


Image corrupted by salt and pepper noise with probabilities $P_a = P_b = 0.25$



Result of filtering with a 7 * 7 median filter



Result of adaptive median filtering with i = 7

Digital Image Processing

Colour Image Processing

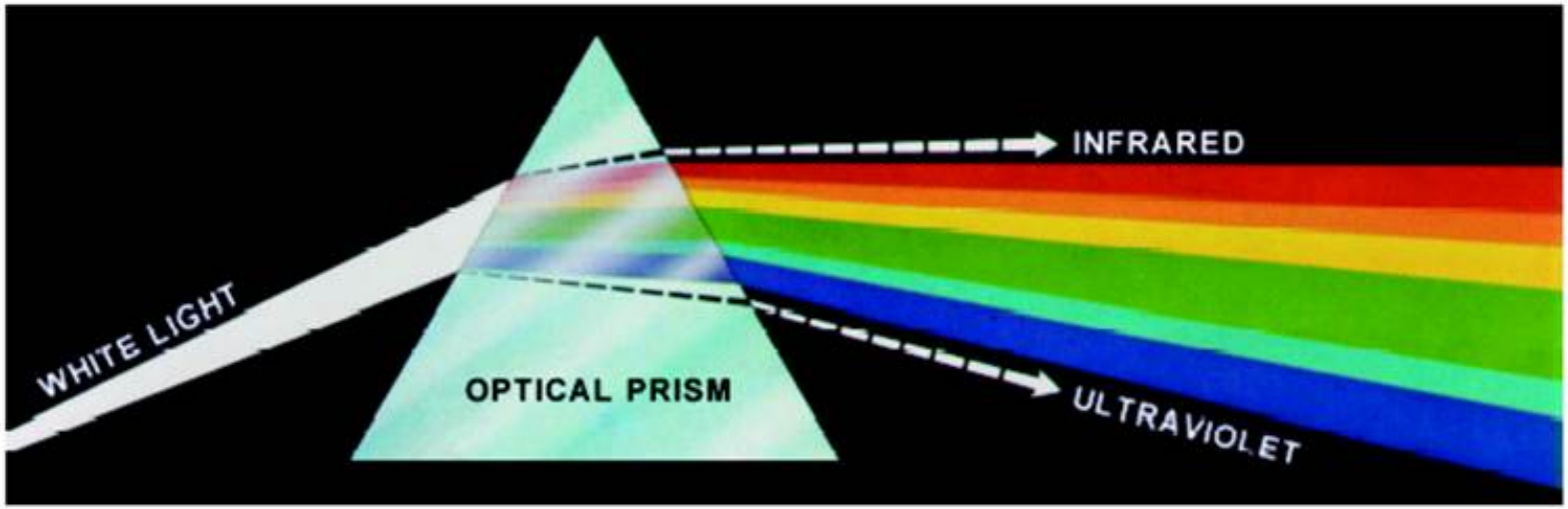
Introduction

Today we'll look at colour image processing, covering:

- Colour fundamentals
- Colour models

Colour Fundamentals

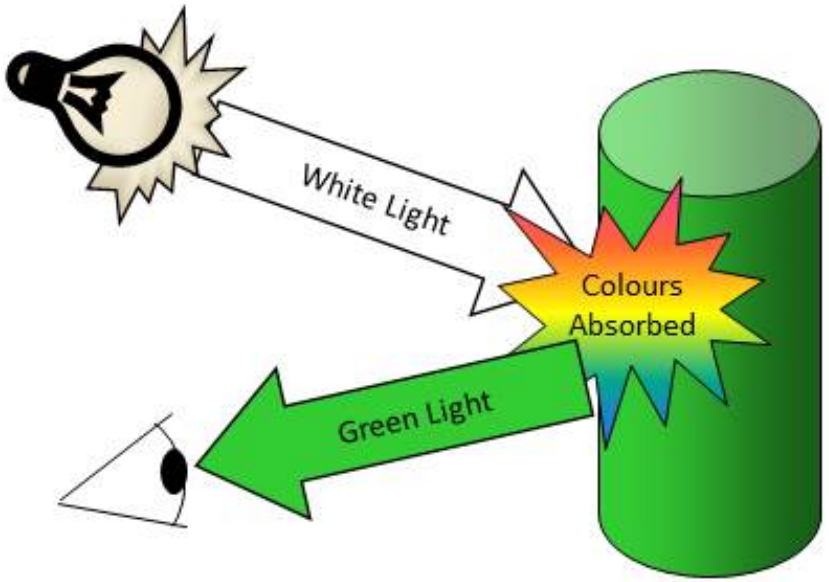
In 1666 Sir Isaac Newton discovered that when a beam of sunlight passes through a glass prism, the emerging beam is split into a spectrum of colours



Colour Fundamentals (cont…)

The colours that humans and most animals perceive in an object are determined by the nature of the light reflected from the object

For example, green objects reflect light with wave lengths primarily in the range of 500 – 570 nm while absorbing most of the energy at other wavelengths

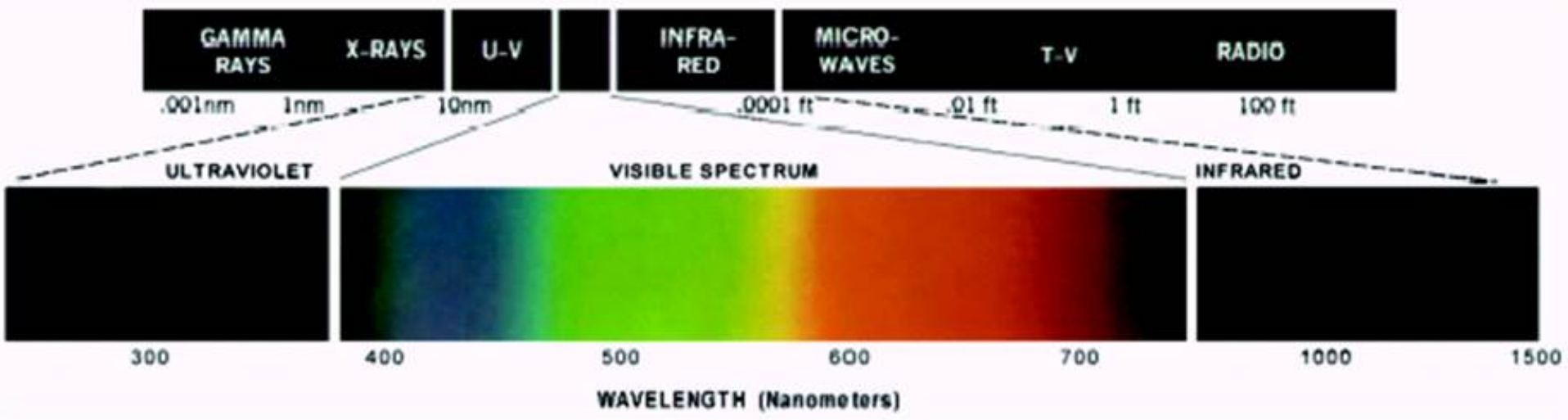


Colour Fundamentals (cont…)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Chromatic light spans the electromagnetic spectrum from approximately 400 to 700 nm

As we mentioned before human colour vision is achieved through 6 to 7 million cones in each eye



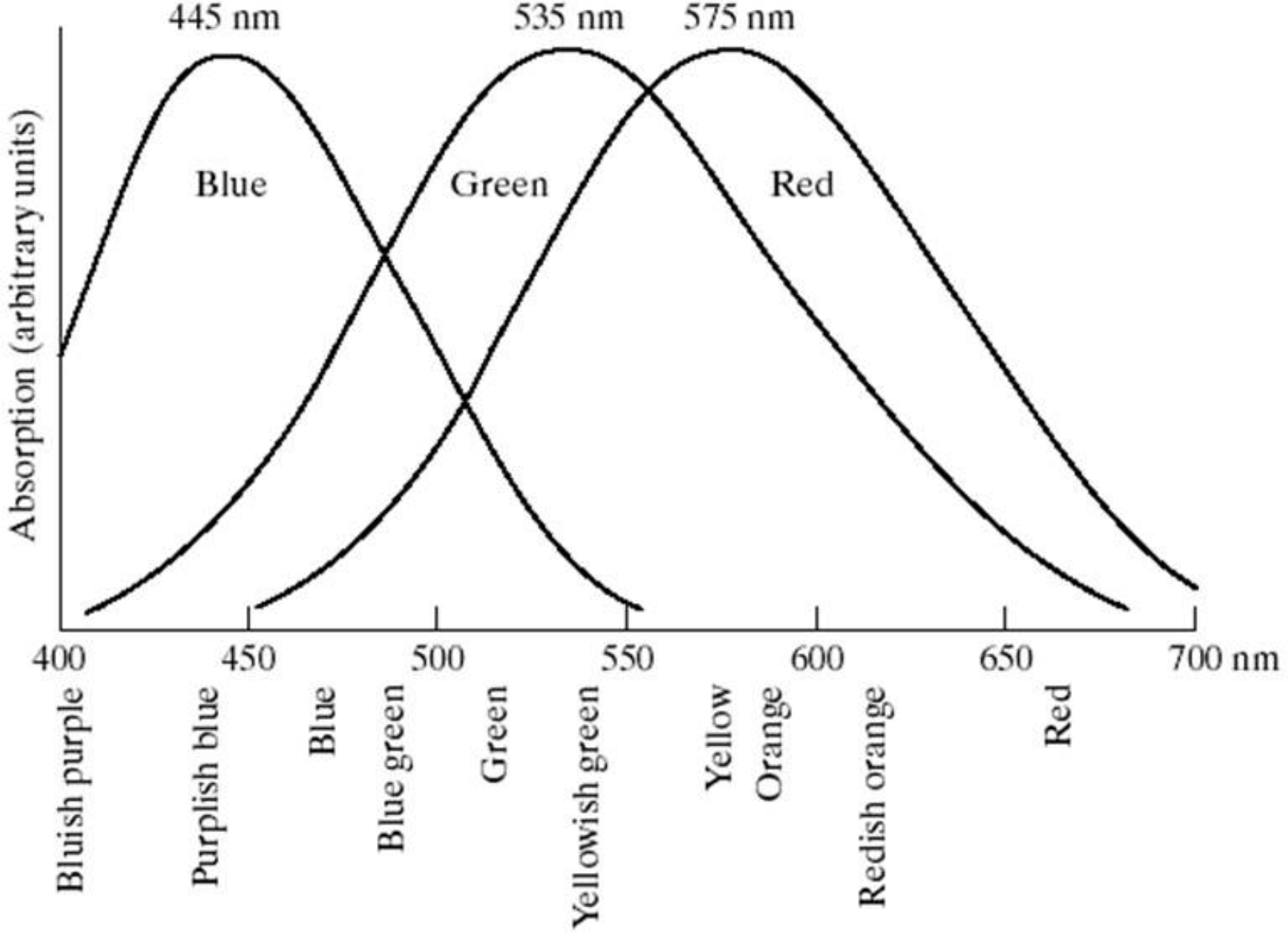
Colour Fundamentals (cont…)

Approximately 66% of these cones are sensitive to red light, 33% to green light and 6% to blue light

Absorption curves for the different cones have been determined experimentally

Strangely these do not match the CIE standards for red (700nm), green (546.1nm) and blue (435.8nm) light as the standards were developed before the experiments!

Colour Fundamentals (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Colour Fundamentals (cont…)

3 basic qualities are used to describe the quality of a chromatic light source:

- **Radiance:** the total amount of energy that flows from the light source (measured in watts)
- **Luminance:** the amount of energy an observer *perceives* from the light source (measured in lumens)
  - Note we can have high radiance, but low luminance
- **Brightness:** a subjective (practically unmeasurable) notion that embodies the intensity of light

We'll return to these later on

CIE Chromacity Diagram

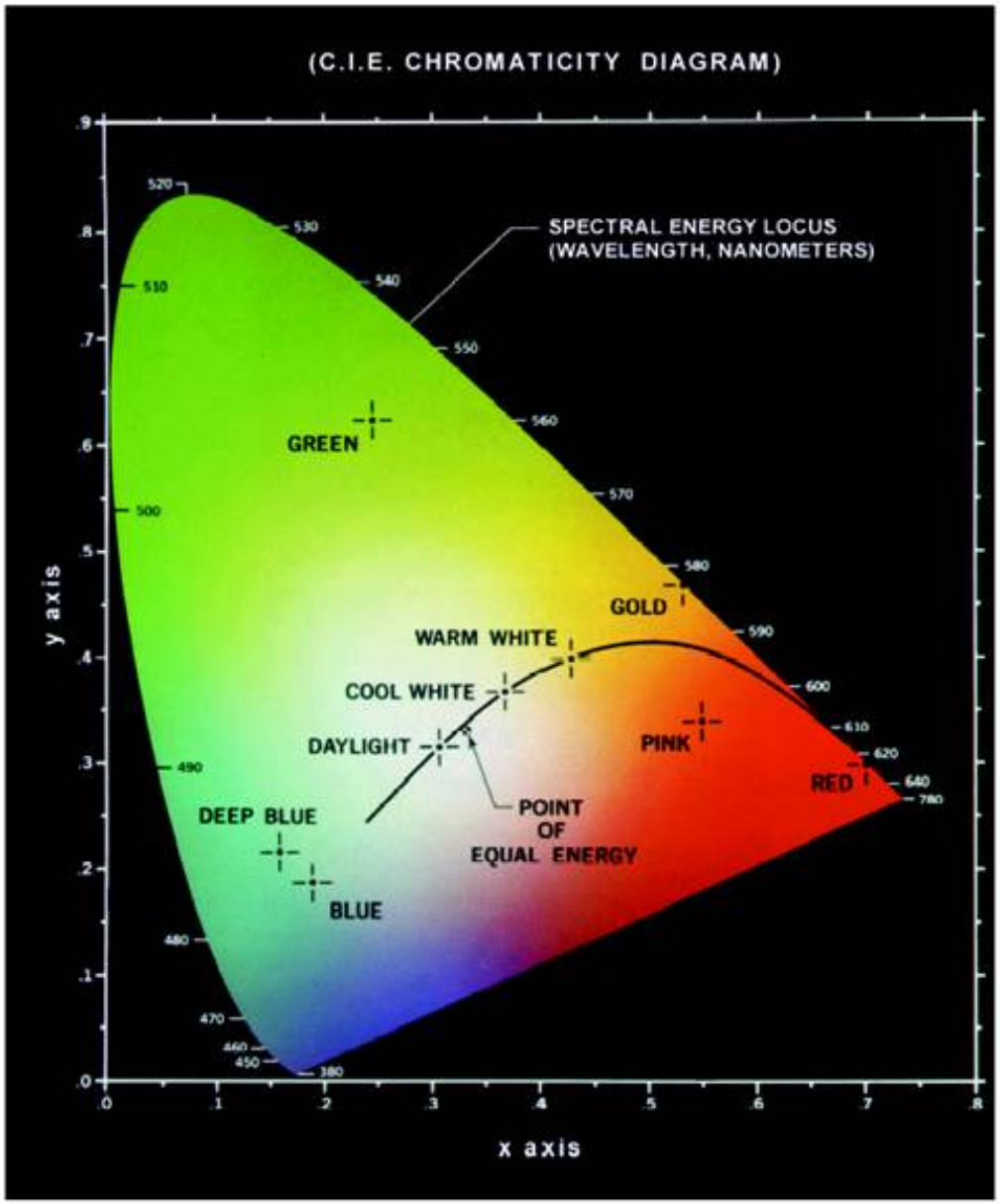
Specifying colours systematically can be achieved using the CIE **chromacity diagram**

On this diagram the x-axis represents the proportion of red and the y-axis represents the proportion of red used

The proportion of blue used in a colour is calculated as:

$$z = 1 - (x + y)$$

CIE Chromacity Diagram (cont…)



Green: 62% green, 25% red and 13% blue

Red: 32% green, 67% red and 1% blue

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

CIE Chromacity Diagram (cont…)

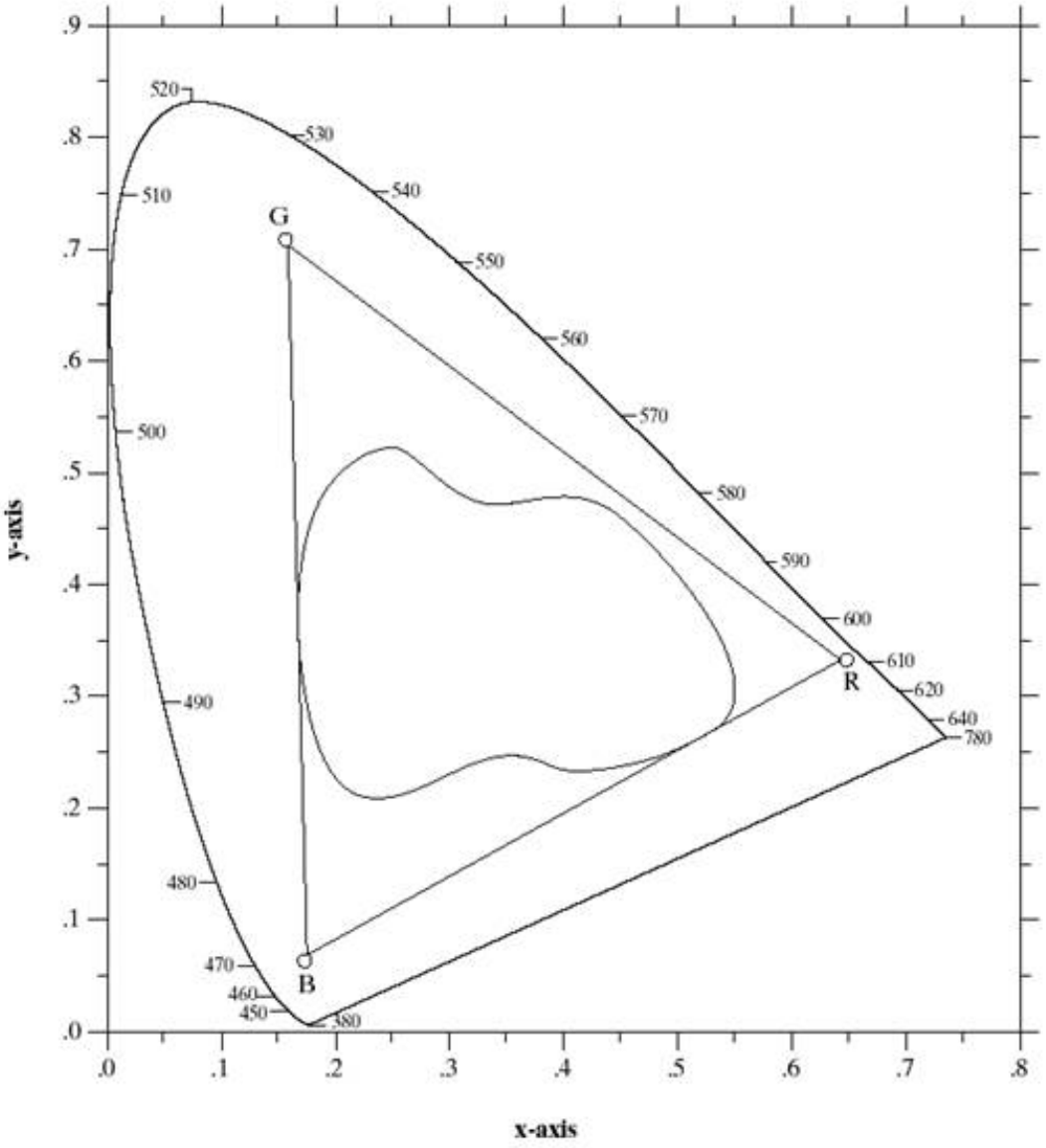
Any colour located on the boundary of the chromacity chart is fully saturated

The point of equal energy has equal amounts of each colour and is the CIE standard for pure white

Any straight line joining two points in the diagram defines all of the different colours that can be obtained by combining these two colours additively

This can be easily extended to three points

CIE Chromacity Diagram (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

This means the entire colour range cannot be displayed based on any three colours

The triangle shows the typical colour gamut produced by RGB monitors

The strange shape is the gamut achieved by high quality colour printers

Colour Models

From the previous discussion it should be obvious that there are different ways to model colour

We will consider two very popular models used in colour image processing:

- RGB (**R**ed **G**reen **B**lue)
- HIS (**H**ue **S**aturation **I**ntensity)

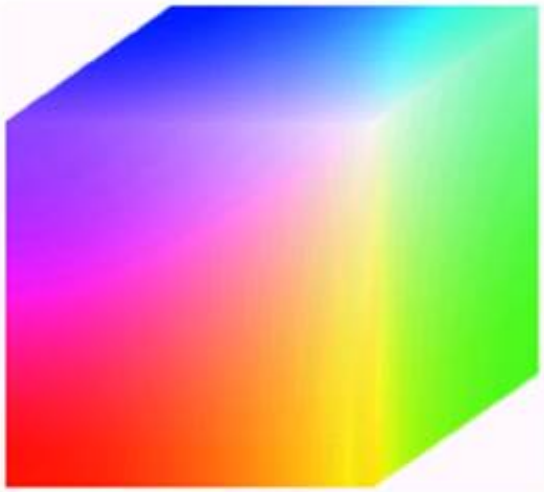
RGB

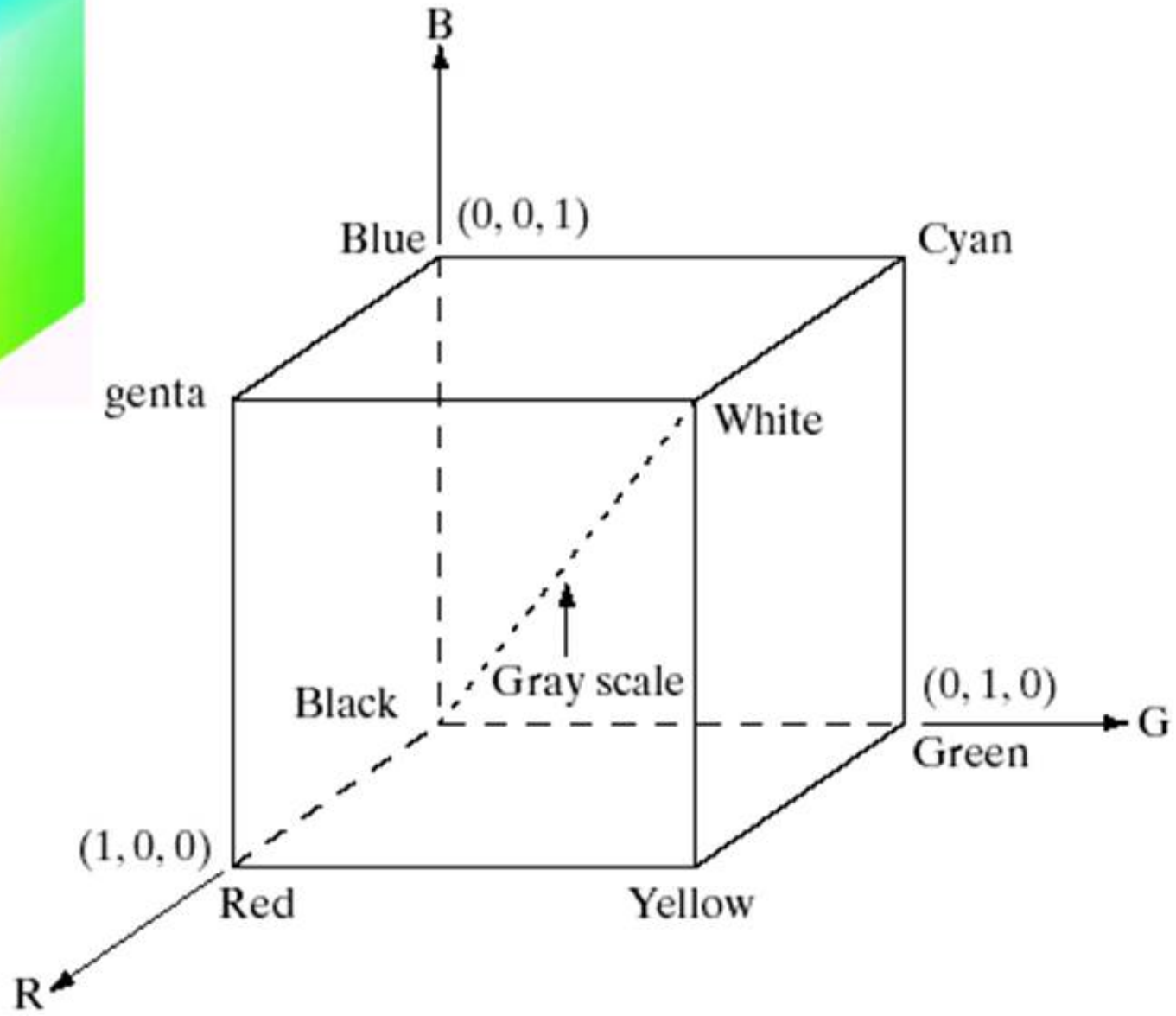
In the RGB model each colour appears in its primary spectral components of red, green and blue

The model is based on a Cartesian coordinate system

- RGB values are at 3 corners
- Cyan magenta and yellow are at three other corners
- Black is at the origin
- White is the corner furthest from the origin
- Different colours are points on or inside the cube represented by RGB vectors

RGB (cont...)





Images taken from Gonzalez & Woods, Digital Image Processing (2002)

RGB (cont...)

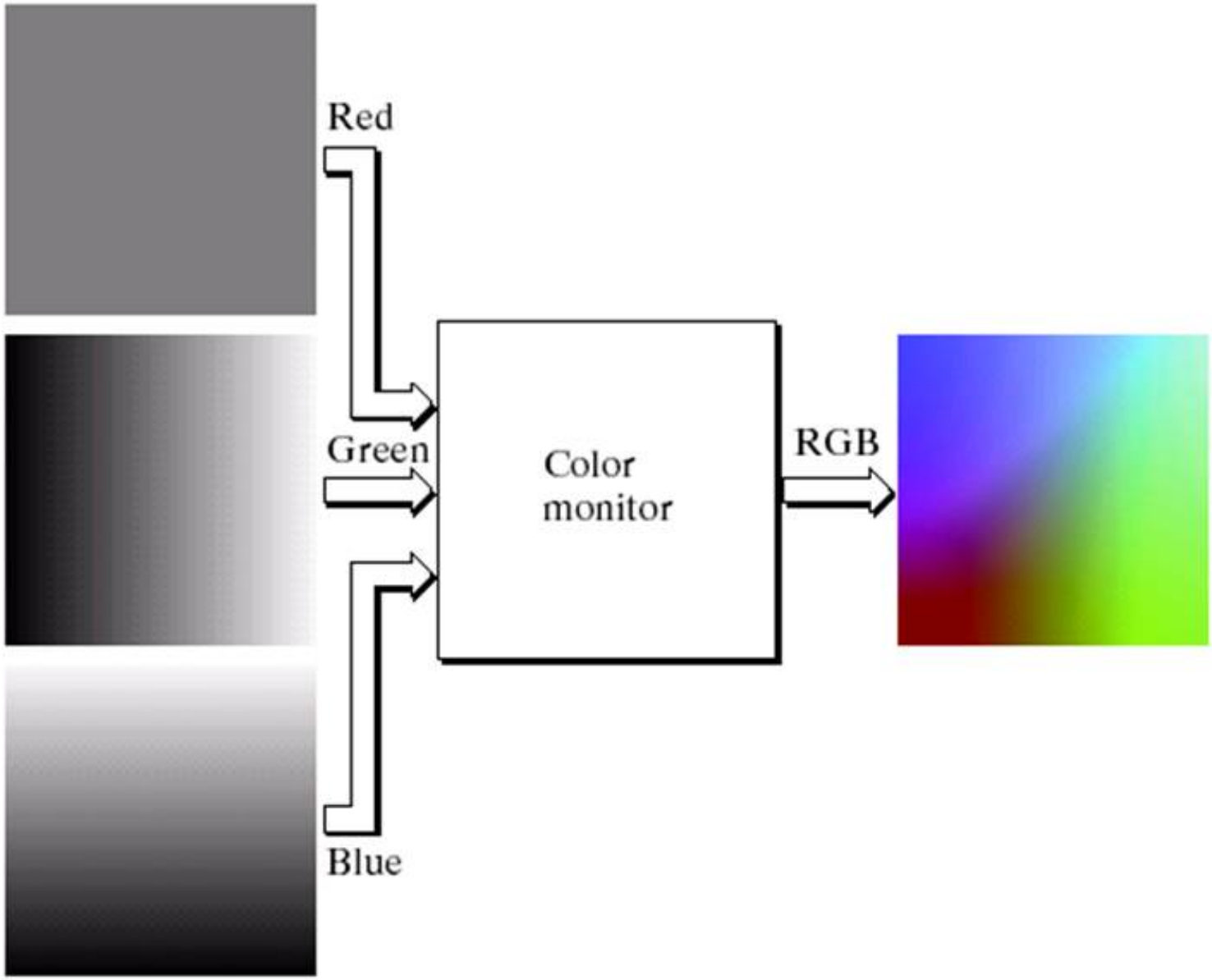
Images represented in the RGB colour model consist of three component images – one for each primary colour

When fed into a monitor these images are combined to create a composite colour image

The number of bits used to represent each pixel is referred to as the colour depth

A 24-bit image is often referred to as a full-colour image as it allows $(2^8)^3$ = 16,777,216 colours

RGB (cont...)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

The HSI Colour Model

RGB is useful for hardware implementations and is serendipitously related to the way in which the human visual system works

However, RGB is not a particularly intuitive way in which to describe colours

Rather when people describe colours they tend to use **hue**, **saturation** and **brightness**

RGB is great for colour generation, but HSI is great for colour description

The HSI Colour Model (cont…)

The HSI model uses three measures to describe colours:

- **Hue:** A colour attribute that describes a pure colour (pure yellow, orange or red)
- **Saturation:** Gives a measure of how much a pure colour is diluted with white light
- **Intensity:** Brightness is nearly impossible to measure because it is so subjective. Instead we use intensity. Intensity is the same achromatic notion that we have seen in grey level images

HSI, Intensity & RGB

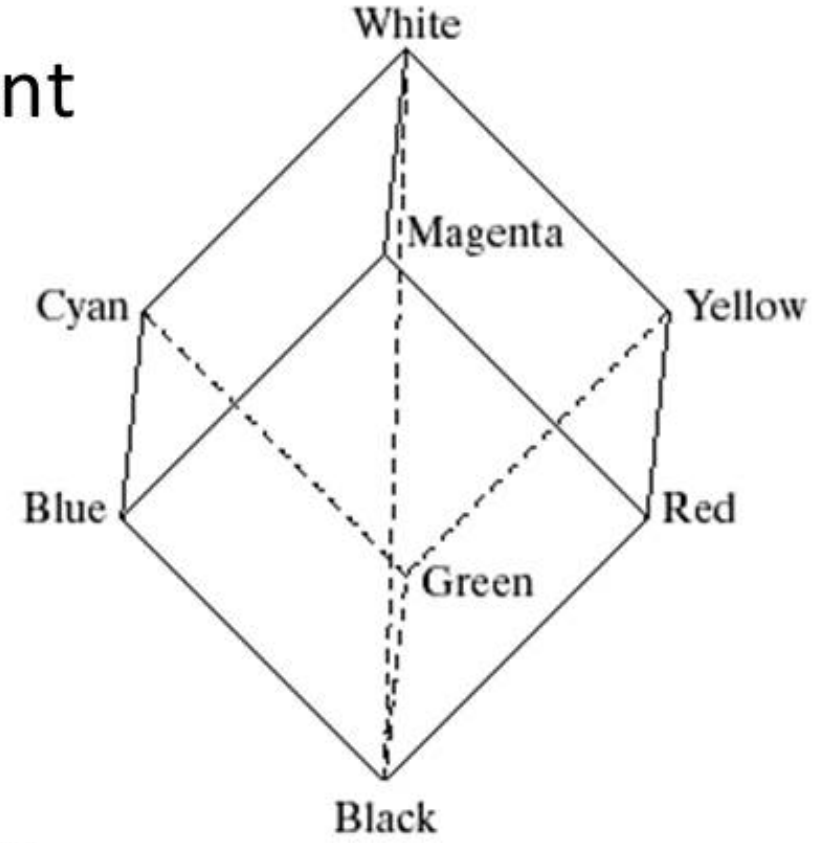
Intensity can be extracted from RGB images – which is not surprising if we stop to think about it

Remember the diagonal on the RGB colour cube that we saw previously ran from black to white

Now consider if we stand this cube on the black vertex and position the white vertex directly above it

HSI, Intensity & RGB (cont…)

Now the intensity component of any colour can be determined by passing a plane *perpendicular* to the intenisty axis and containing the colour point



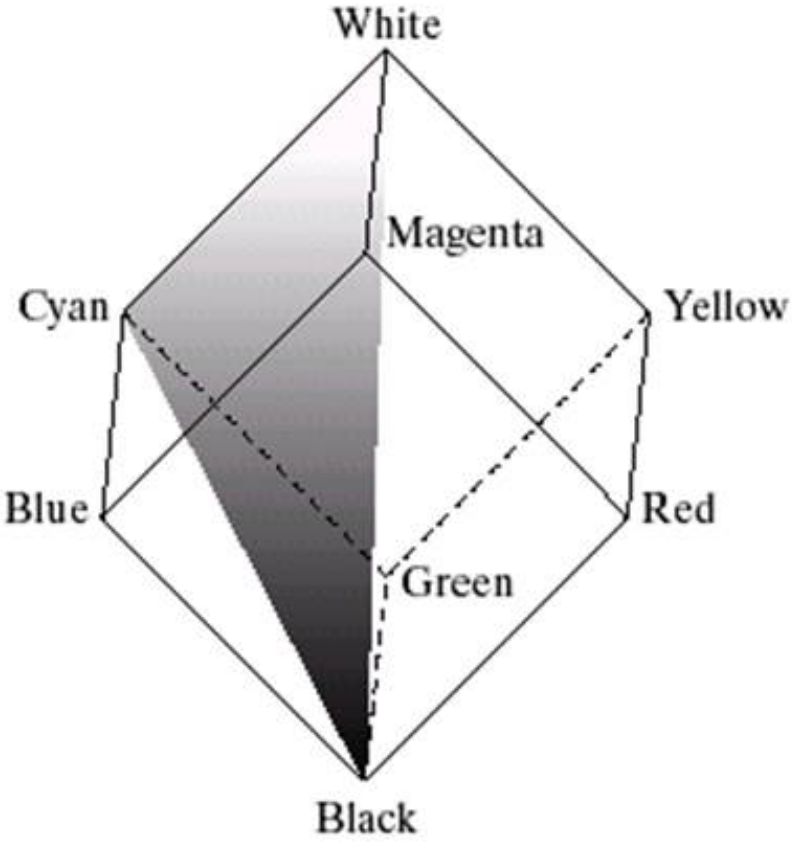
The intersection of the plane with the intensity axis gives us the intensity component of the colour

HSI, Hue & RGB

In a similar way we can extract the hue from the RGB colour cube

Consider a plane defined by the three points cyan, black and white

All points contained in this plane must have the same hue (cyan) as black and white cannot contribute hue information to a colour

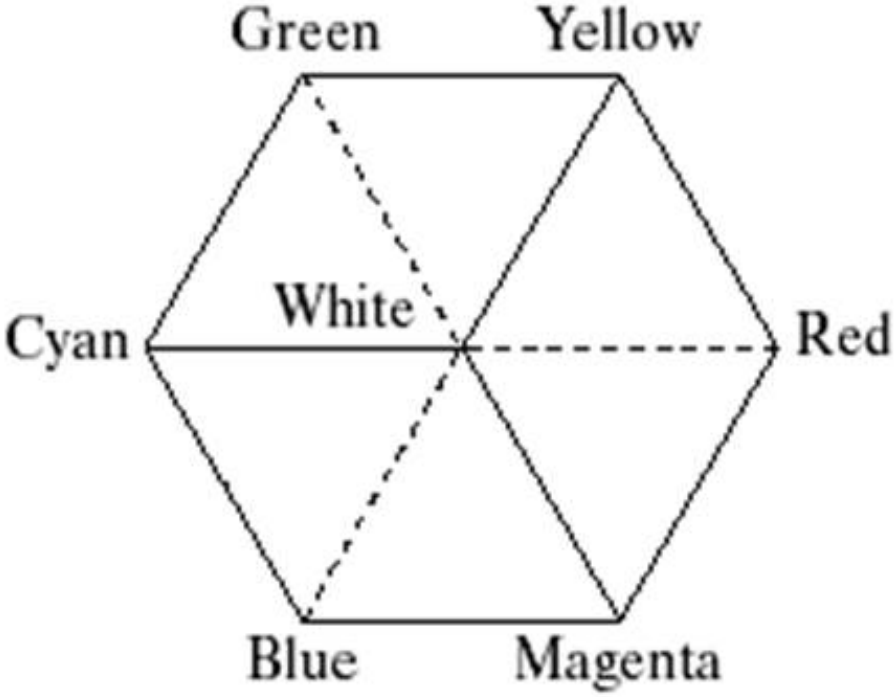


The HSI Colour Model

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Consider if we look straight down at the RGB cube as it was arranged previously

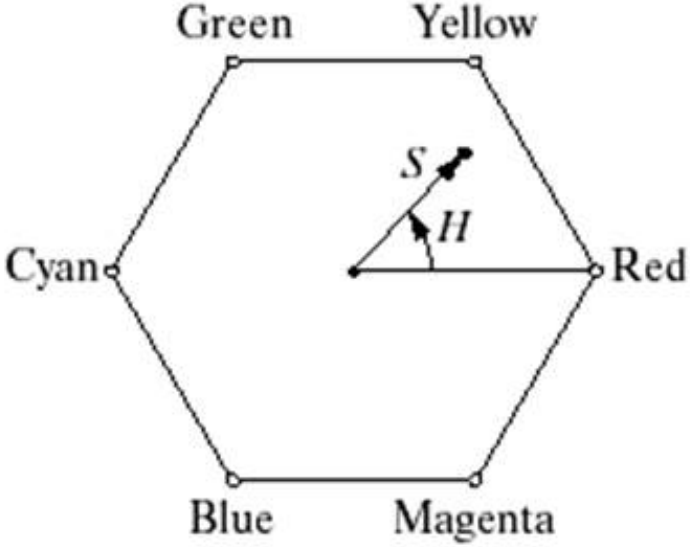
We would see a hexagonal shape with each primary colour separated by 120° and secondary colours at 60° from the primaries



So the HSI model is composed of a vertical intensity axis and the locus of colour points that lie on planes perpendicular to that axis

The HSI Colour Model (cont…)

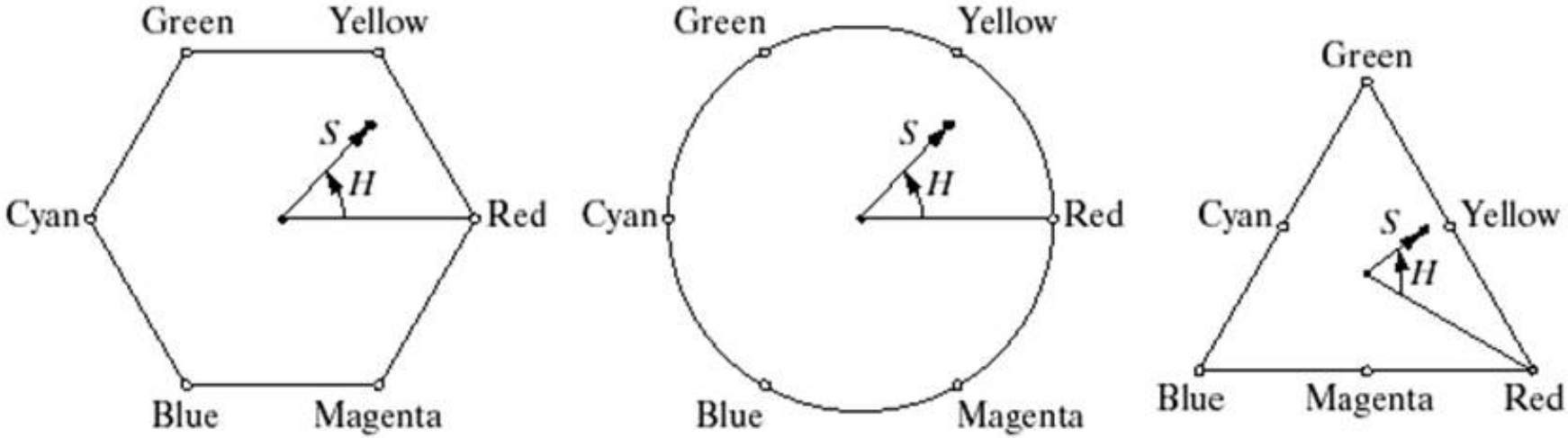
To the right we see a hexagonal shape and an arbitrary colour point



- The hue is determined by an angle from a reference point, usually red
- The saturation is the distance from the origin to the point
- The intensity is determined by how far up the vertical intenisty axis this hexagonal plane sits (not apparent from this diagram

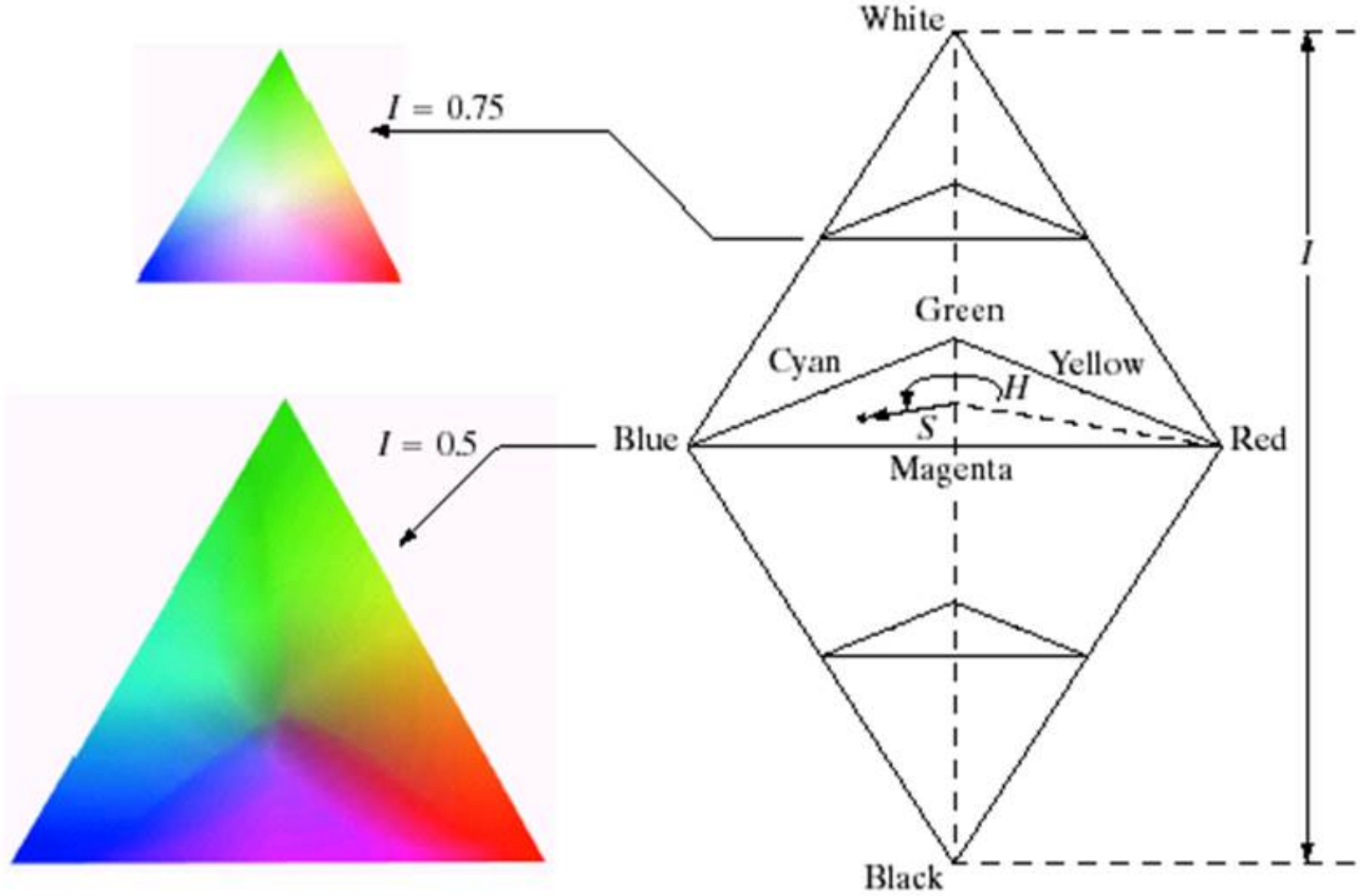
The HSI Colour Model (cont…)

Because the only important things are the angle and the length of the saturation vector this plane is also often represented as a circle or a triangle

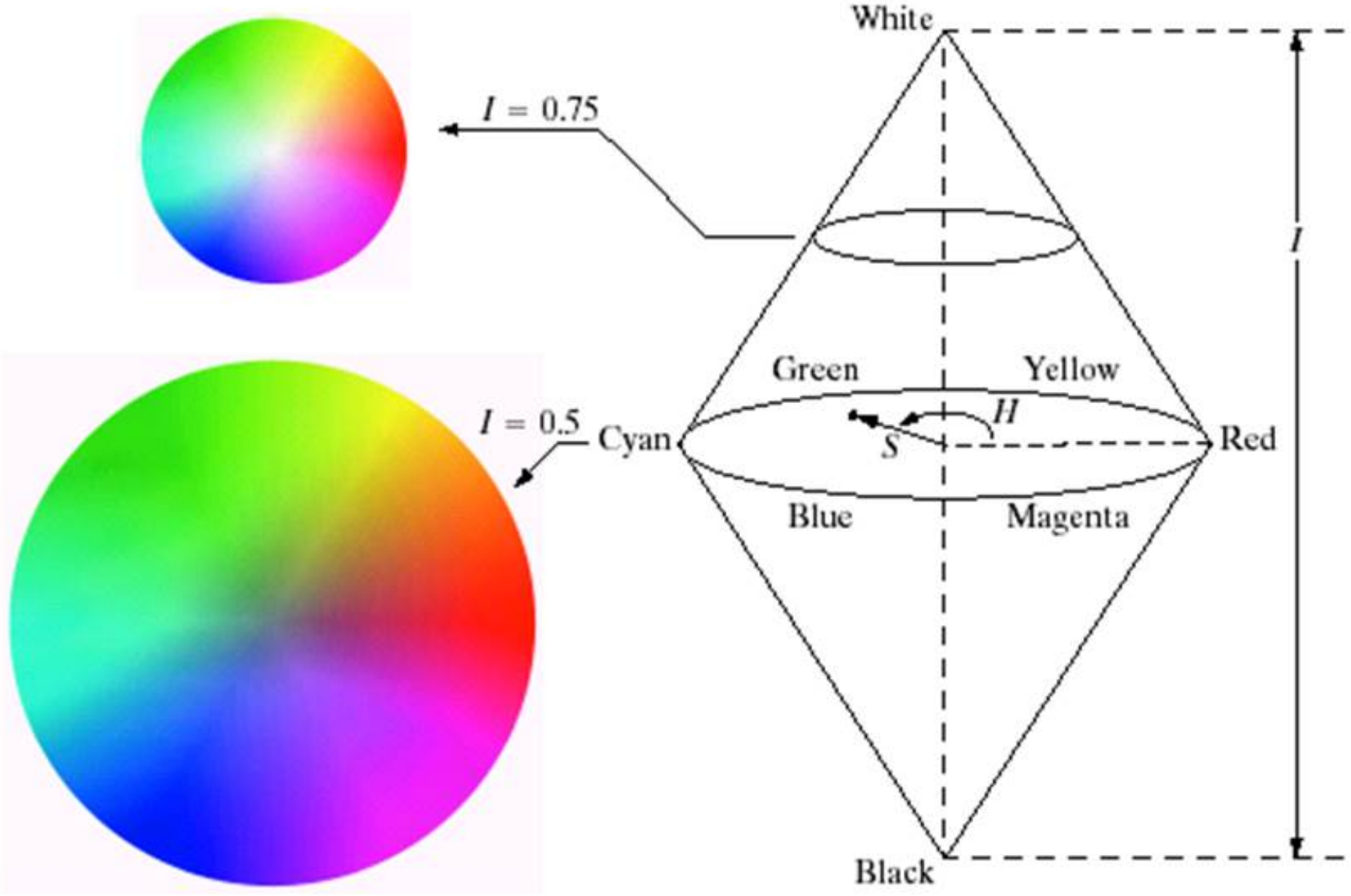


Images taken from Gonzalez & Woods, Digital Image Processing (2002)

HSI Model Examples



HSI Model Examples



Converting From RGB To HSI

Given a colour as R, G, and B its H, S, and I values are calculated as follows:

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

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

Converting From HSI To RGB

Given a colour as H, S, and I it's R, G, and B values are calculated as follows:

- RG sector ($0 <= H < 120^\circ$)

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

- GB sector ($120^\circ <= H < 240^\circ$)

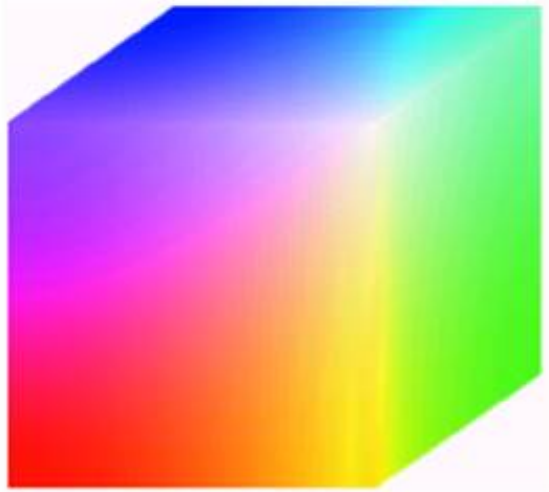
$$R = I(1 - S) \qquad G = I\left[1 + \frac{S\cos(H - 120)}{\cos(H - 60)}\right] \qquad B = 3I - (R + G)$$

Converting From HSI To RGB (cont…)

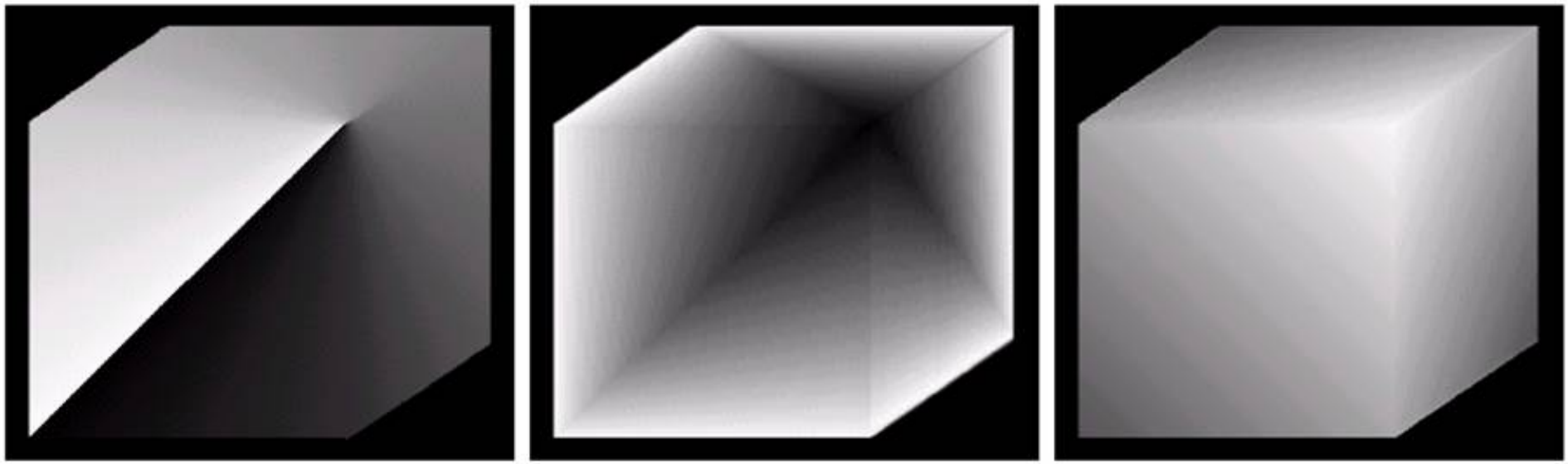
– BR sector ($240^\circ <= H <= 360^\circ$)

$$R = 3I - (G + B) \quad G = I(1 - S) \quad B = I\left[1 + \frac{S\cos(H - 240)}{\cos(H - 180)}\right]$$

HSI & RGB



RGB Colour Cube



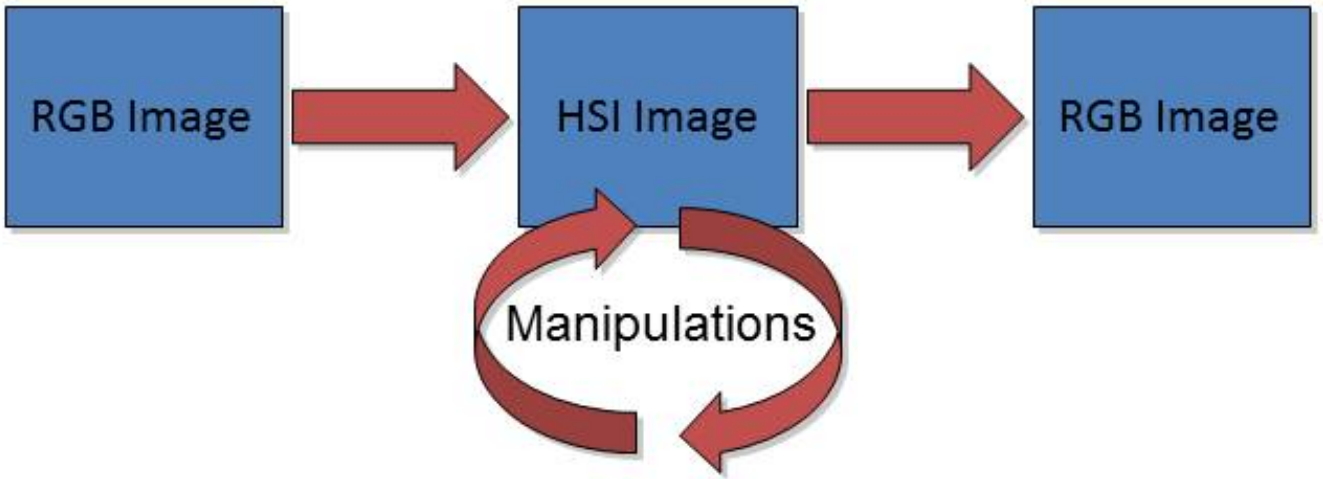
H, S, and I Components of RGB Colour Cube

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

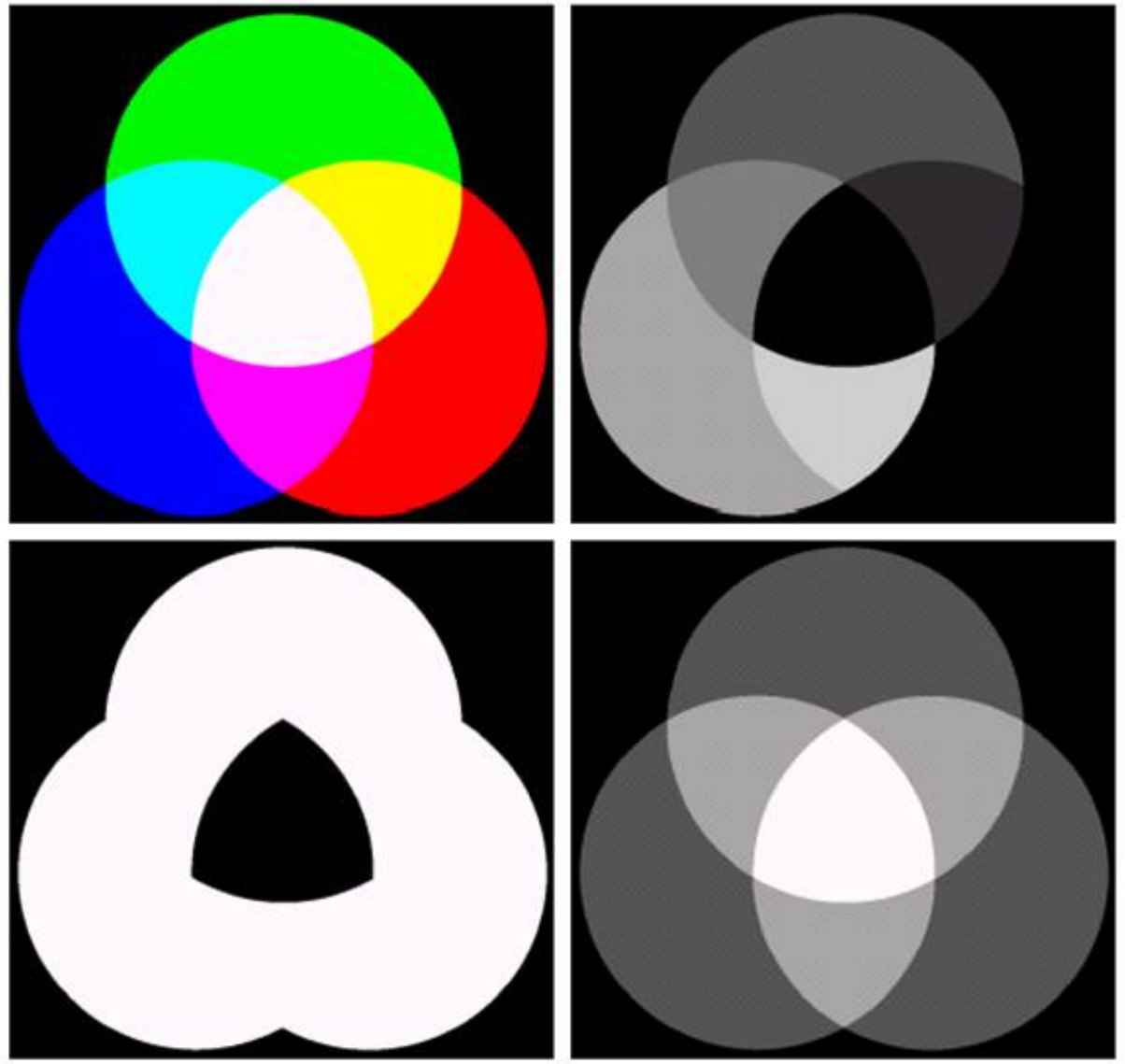
Manipulating Images In The HSI Model

In order to manipulate an image under the HIS model we:

- First convert it from RGB to HIS
- Perform our manipulations under HSI
- Finally convert the image back from HSI to RGB

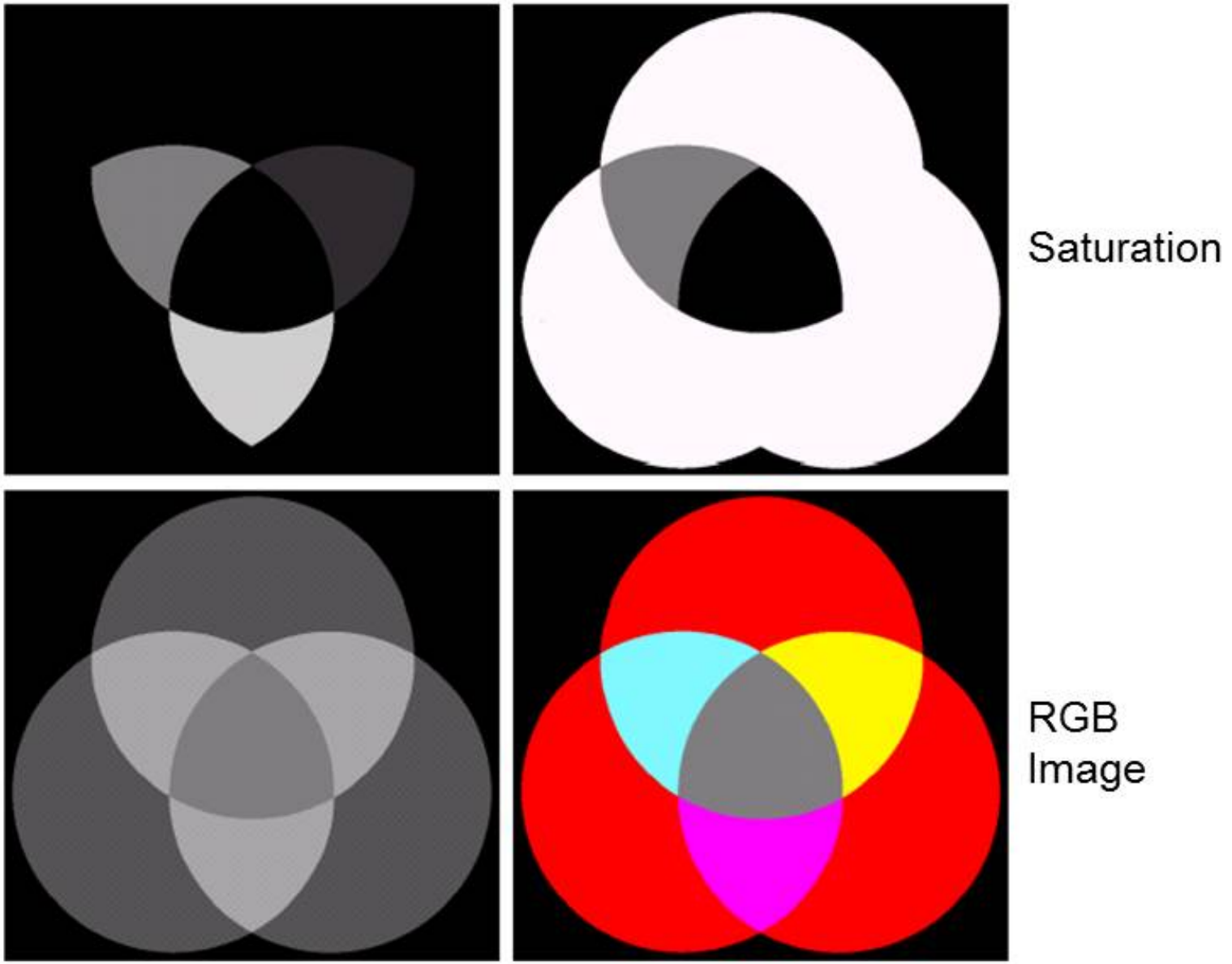


RGB -> HSI -> RGB



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

RGB -> HSI -> RGB (cont…)

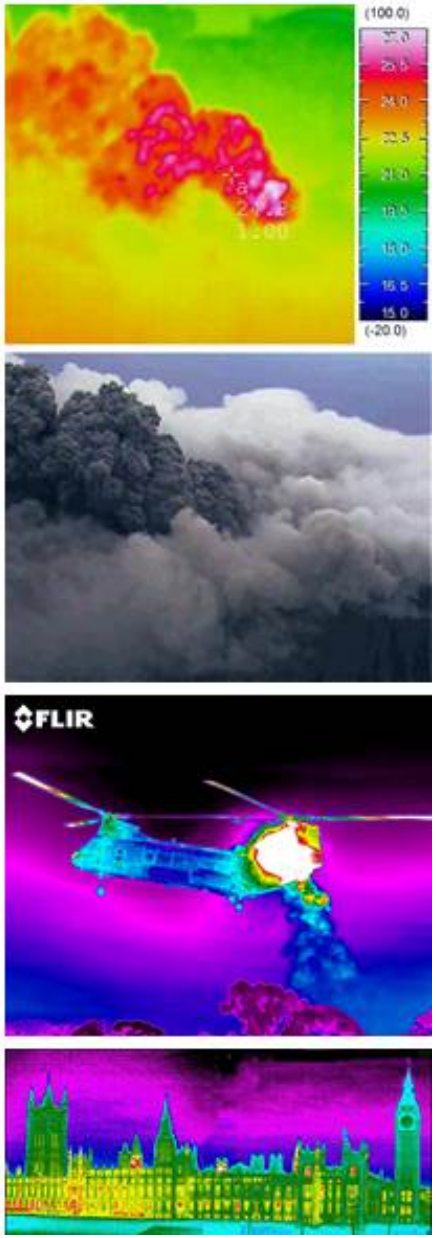


Pseudocolour Image Processing

Pseudocolour (also called false colour) image processing consists of assigning colours to grey values based on a specific criterion

The principle use of pseudocolour image processing is for human visualisation

- Humans can discern between thousands of colour shades and intensities, compared to only about two dozen or so shades of grey



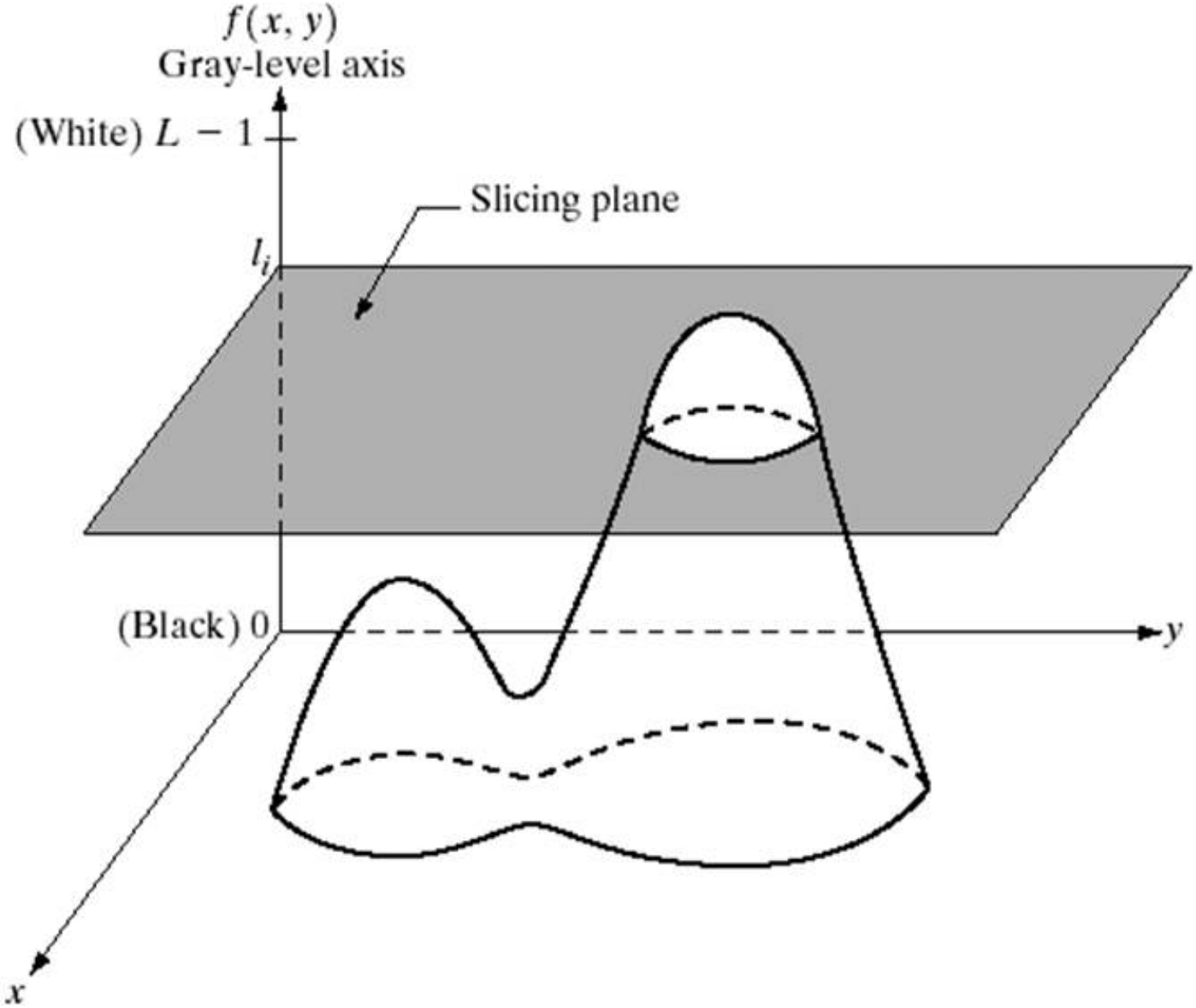
Pseudo Colour Image Processing – Intensity Slicing

Intensity slicing and colour coding is one of the simplest kinds of pseudocolour image processing

First we consider an image as a 3D function mapping spatial coordinates to intensities (that we can consider heights)

Now consider placing planes at certain levels parallel to the coordinate plane

If a value is one side of such a plane it is rendered in one colour, and a different colour if on the other side



In general intensity slicing can be summarised as:

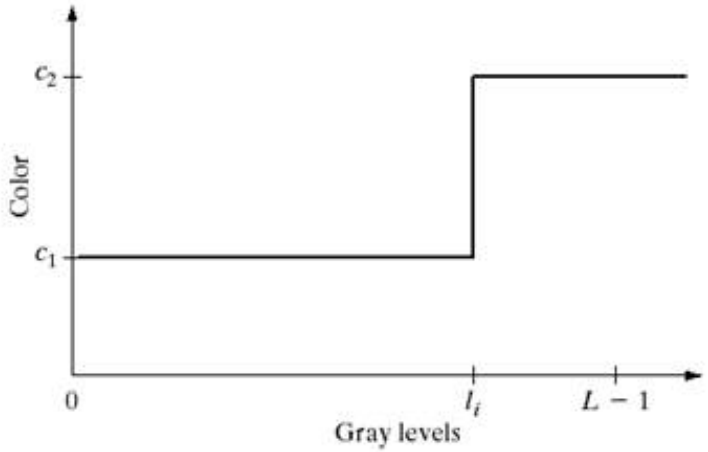
- Let $[0, L\text{-}1]$ represent the grey scale
- Let l_0 represent black $[f(x, y) = 0]$ and let l_{L-1} represent white $[f(x, y) = L\text{-}1]$
- Suppose P planes perpendicular to the intensity axis are defined at levels $l_1, l_2, \ldots, l_p$
- Assuming that $0 < P < L\text{-}1$ then the P planes partition the grey scale into $P + 1$ intervals $V_1, V_2, \ldots, V_{P+1}$

– Grey level colour assignments can then be made according to the relation:

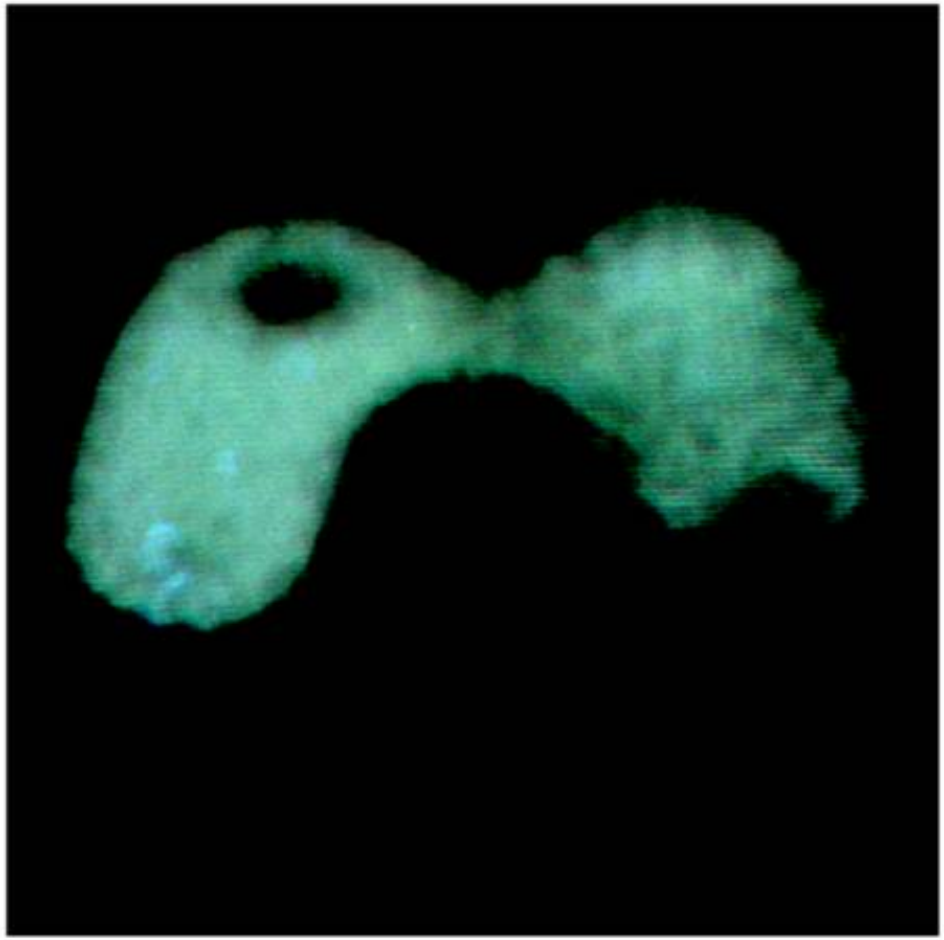
$$f(x,y) = c_k \qquad \text{if } f(x,y) \in V_k$$

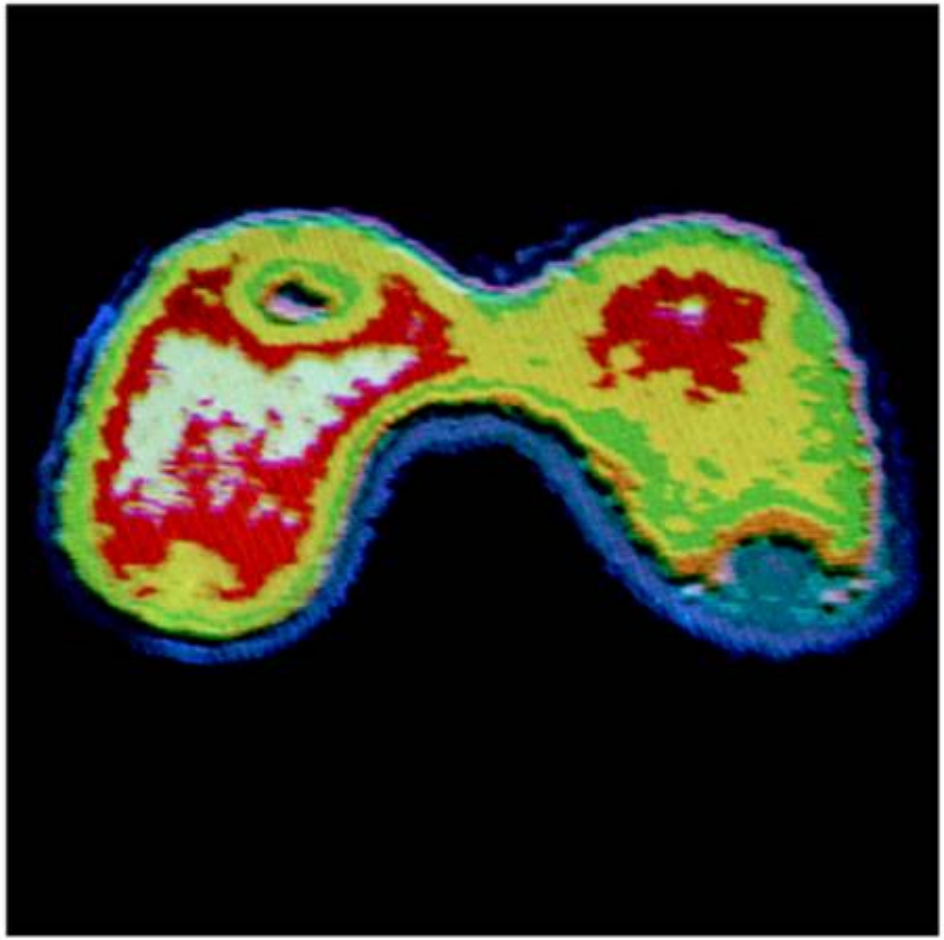
– where ck is the colour associated with the k^{th} intensity level V_k defined by the partitioning planes at $l = k - 1$ and $l = k$

RGB -> HSI -> RGB (cont…)



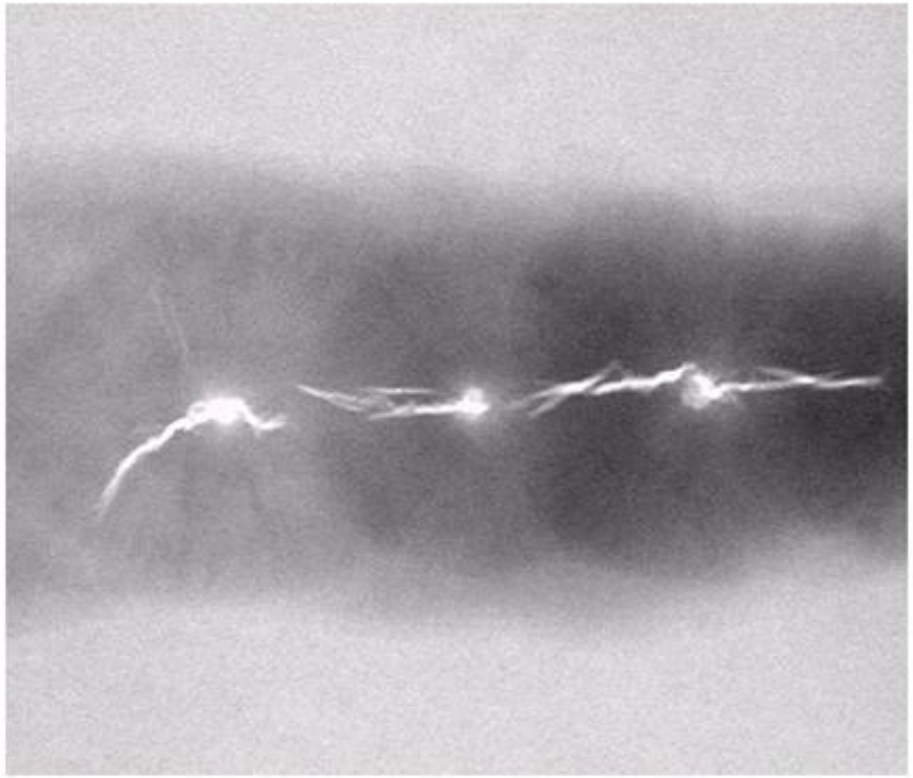
RGB -> HSI -> RGB (cont…)

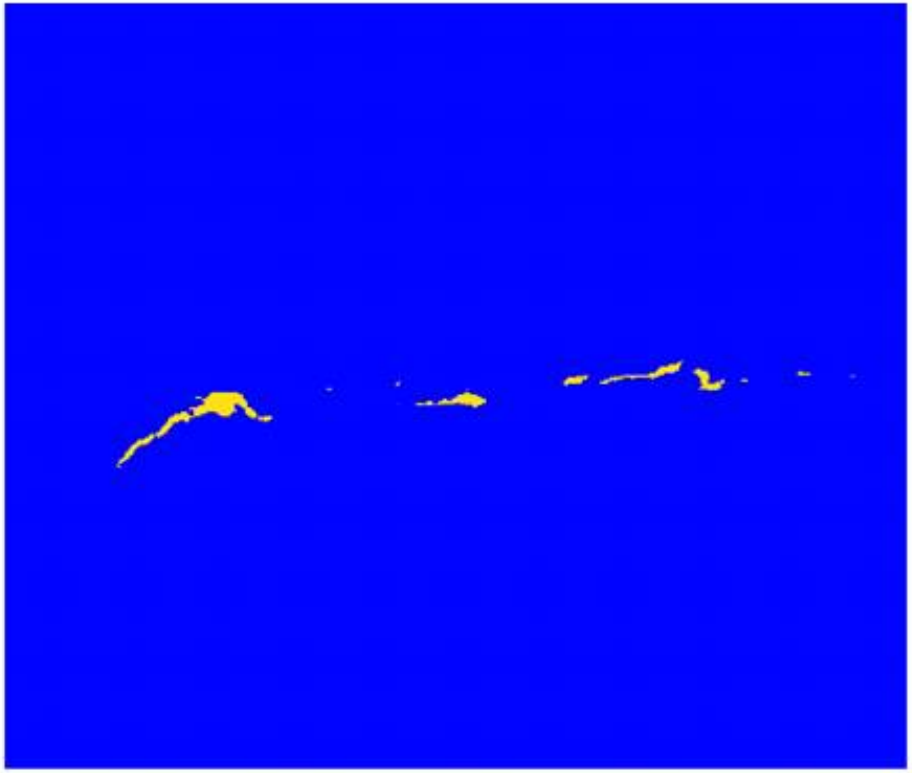




Images taken from Gonzalez & Woods, Digital Image Processing (2002)

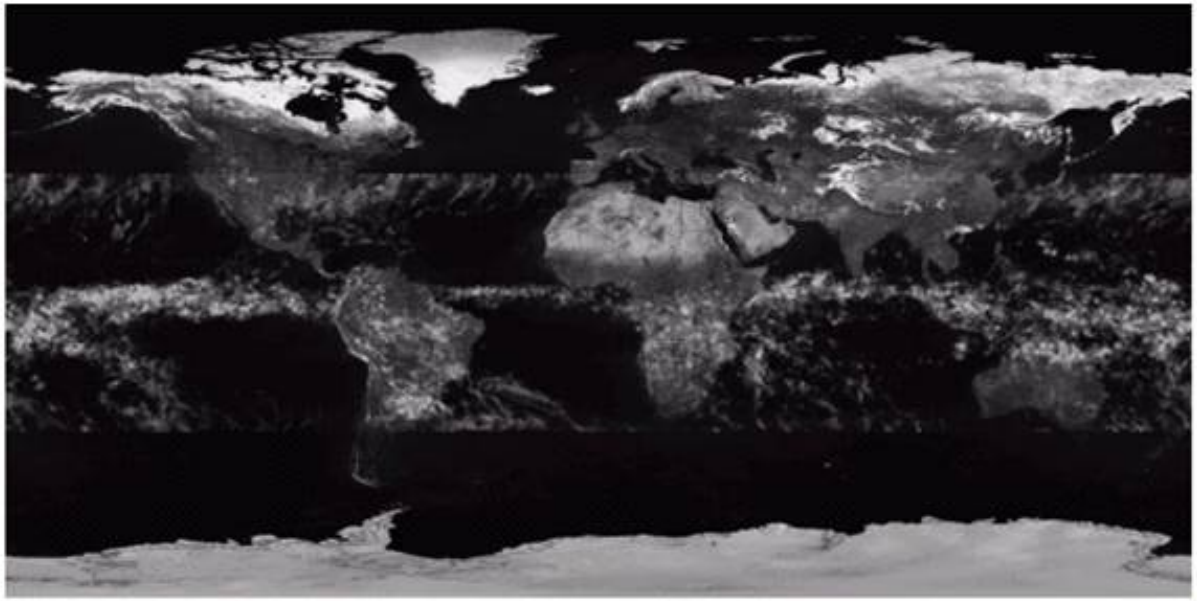
RGB -> HSI -> RGB (cont…)

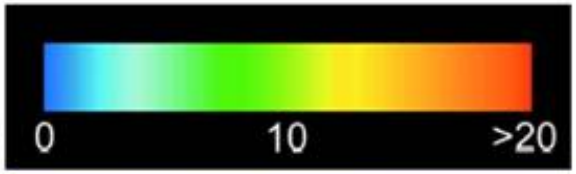


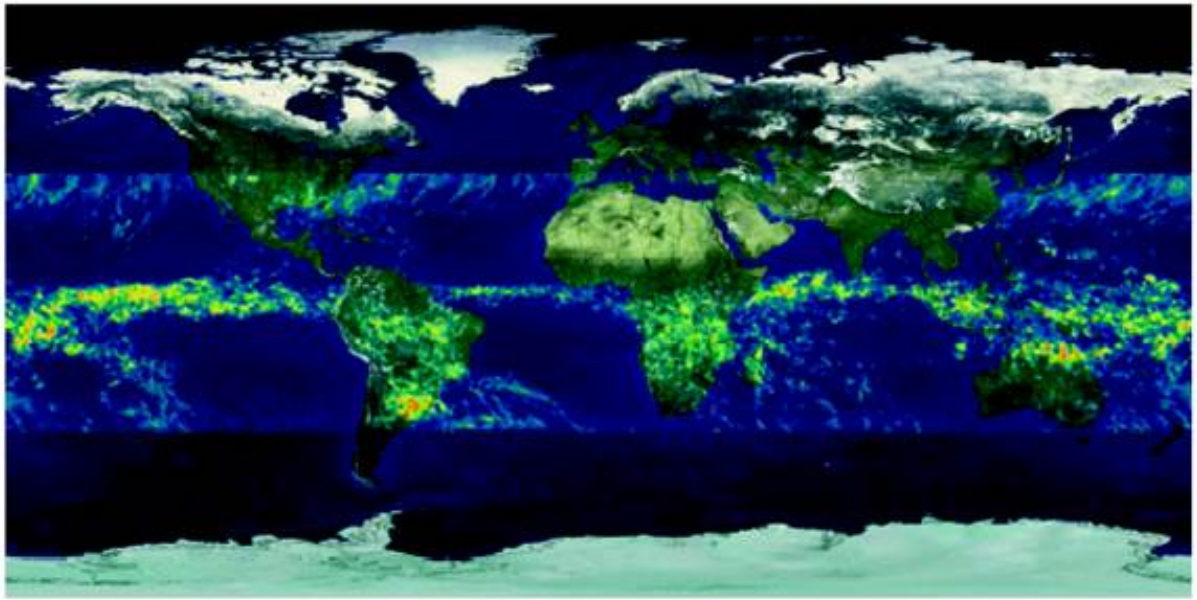


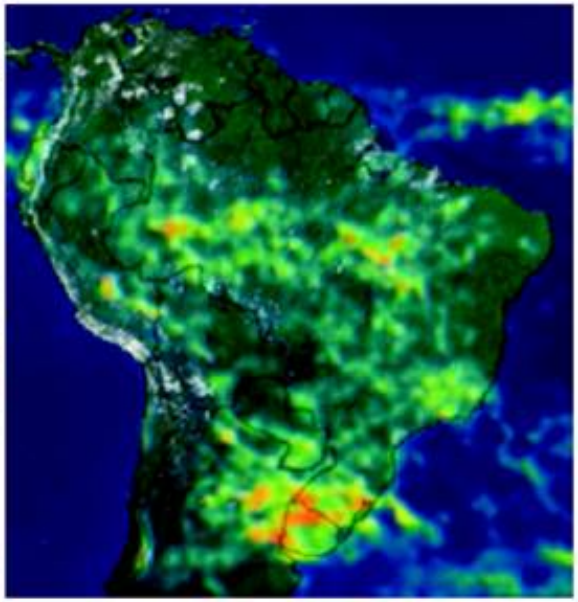
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

RGB -> HSI -> RGB (cont…)









Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Chapter 6
Color Image Processing

Chapter 6
Color Image Processing

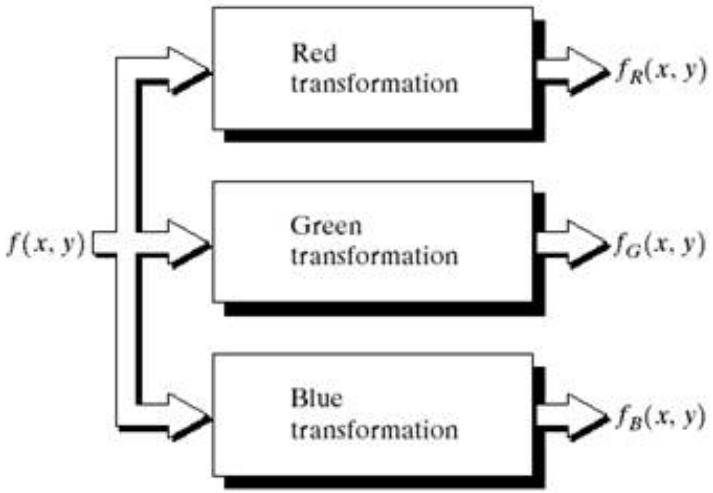
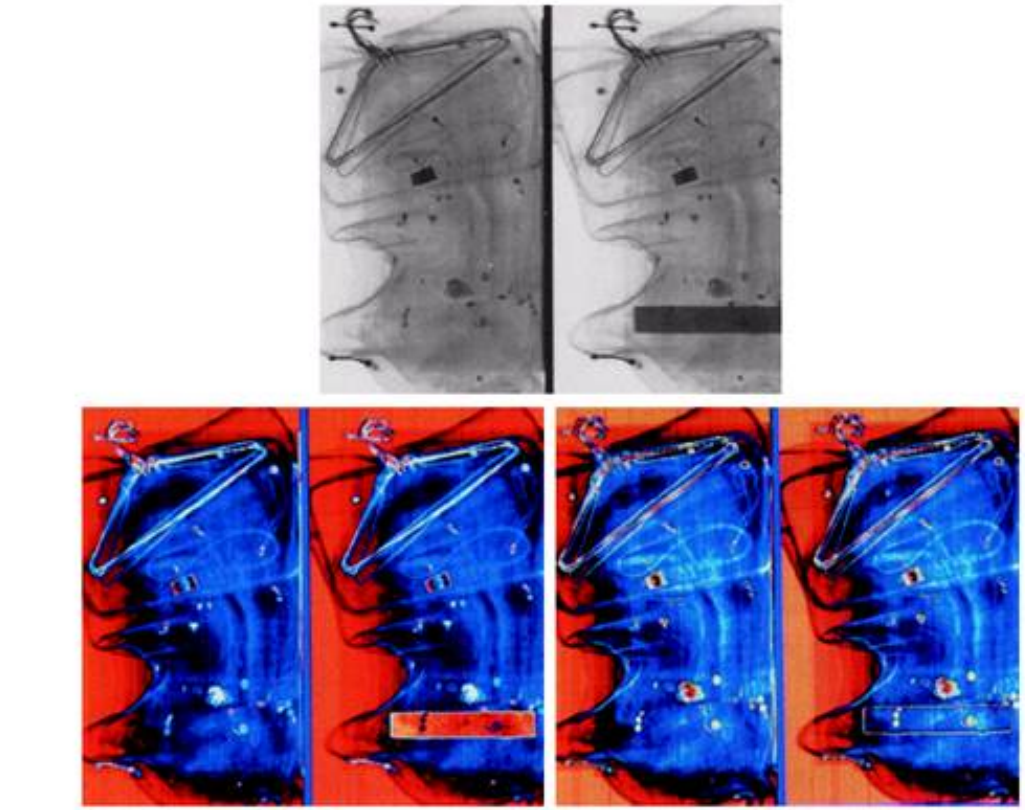


FIGURE 6.23 Functional block diagram for pseudocolor image processing. f_R, f_G, and f_B are fed into the corresponding red, green, and blue inputs of an RGB color monitor.

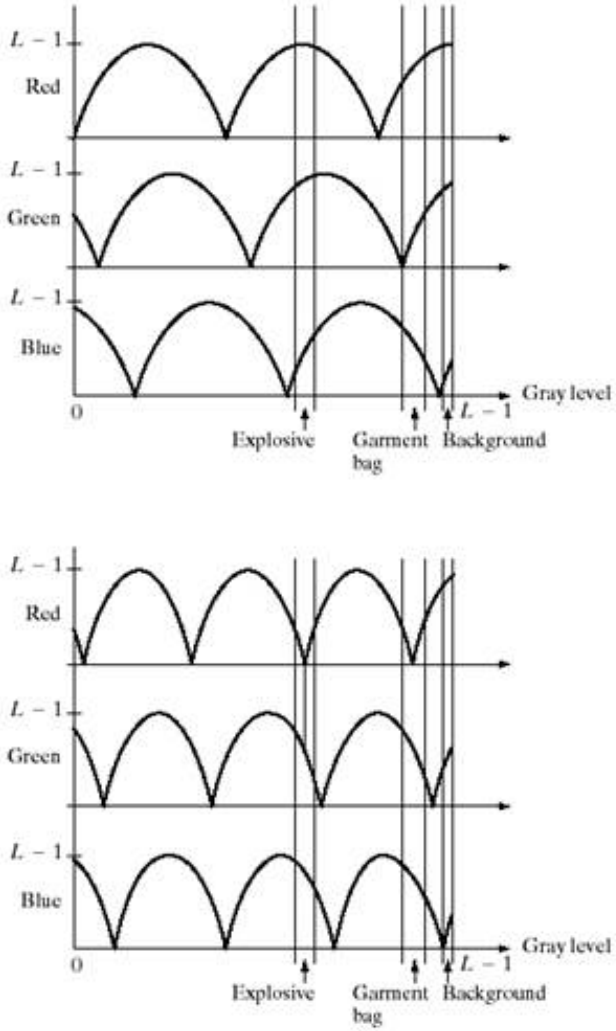
Chapter 6
Color Image Processing



a
b c

FIGURE 6.24 Pseudocolor enhancement by using the gray-level to color transformations in Fig. 6.25. (Original image courtesy of Dr. Mike Hurwitz, Westinghouse.)

Chapter 6
Color Image Processing



a
b

FIGURE 6.25 Transformation functions used to obtain the images in Fig. 6.24.

Chapter 6
Color Image Processing

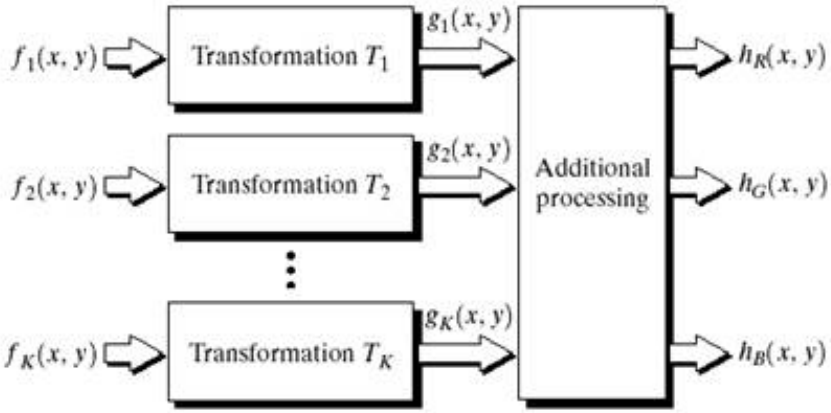
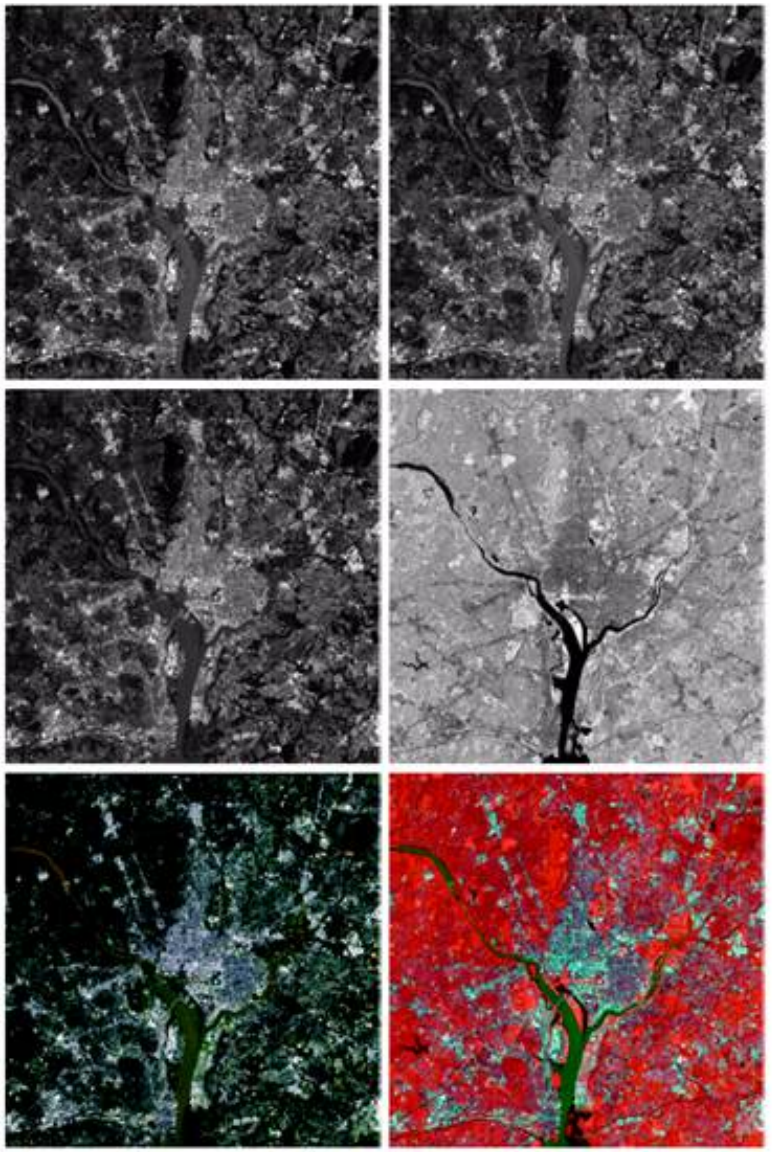


FIGURE 6.26 A pseudocolor coding approach used when several monochrome images are available.

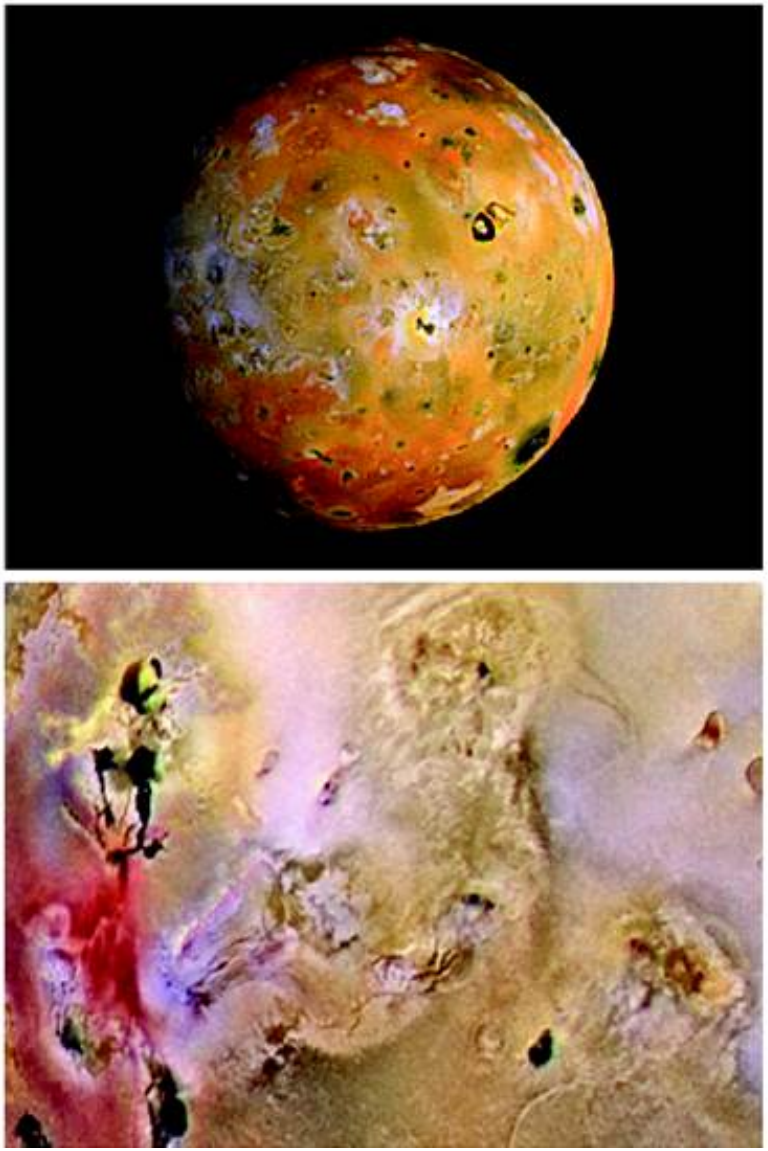
Chapter 6
Color Image Processing



a b
c d
e f

FIGURE 6.27 (a)–(d) Images in bands 1–4 in Fig. 1.10 (see Table 1.1). (e) Color composite image obtained by treating (a), (b), and (c) as the red, green, blue components of an RGB image. (f) Image obtained in the same manner, but using in the red channel the near-infrared image in (d). (Original multispectral images courtesy of NASA.)

Chapter 6
Color Image Processing



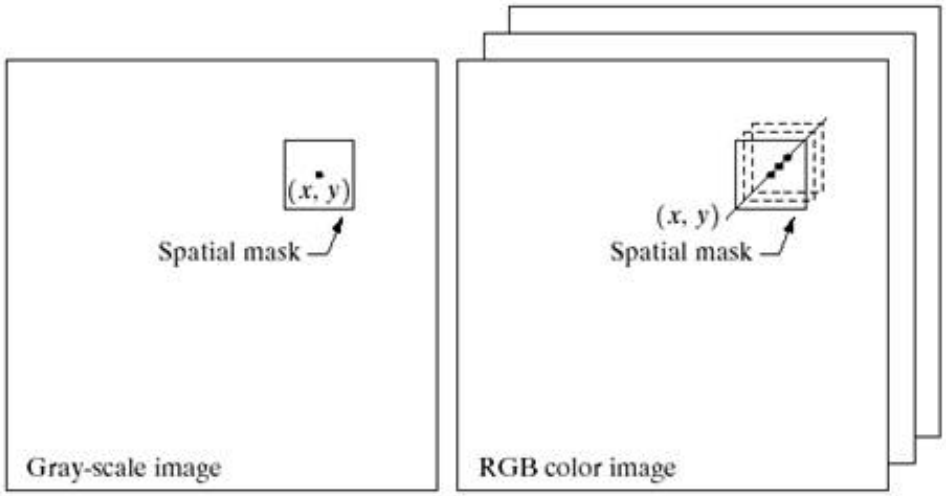
a
b

FIGURE 6.28
(a) Pseudocolor rendition of Jupiter Moon Io. (b) A close-up. (Courtesy of NASA.)

Chapter 6
Color Image Processing

a b

FIGURE 6.29
Spatial masks for gray-scale and RGB color images.



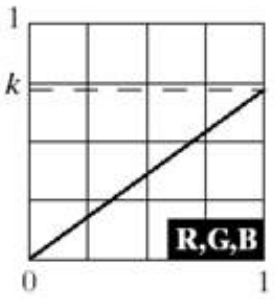
Chapter 6
Color Image Processing

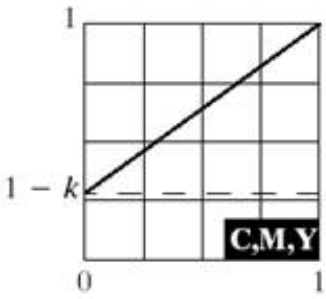
FIGURE 6.30 A full-color image and its various color-space components. (Original image courtesy of Med-Data Interactive.)

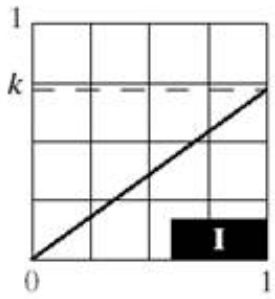
Chapter 6
Color Image Processing

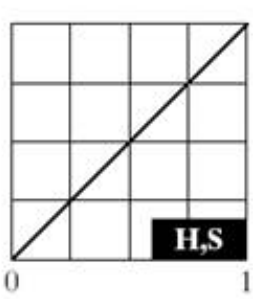
a b
c d e

FIGURE 6.31
Adjusting the intensity of an image using color transformations. (a) Original image. (b) Result of decreasing its intensity by 30% (i.e., letting $k = 0.7$). (c)–(e) The required RGB, CMY, and HSI transformation functions. (Original image courtesy of MedData Interactive.)









Chapter 6
Color Image Processing

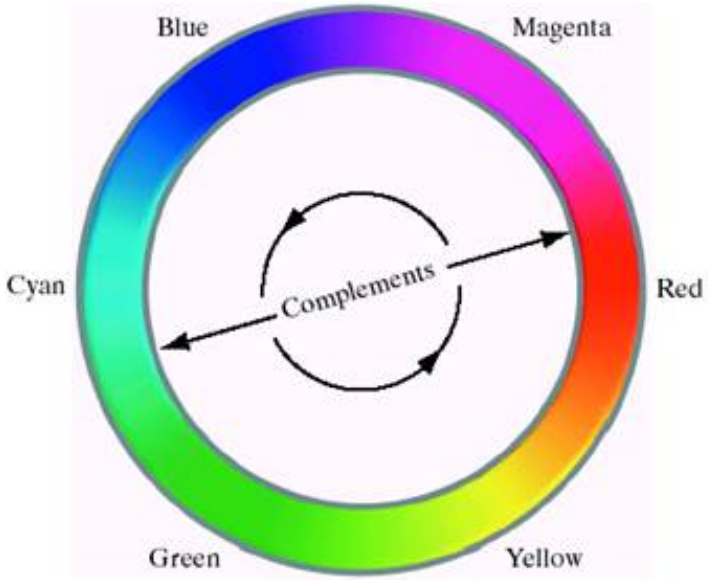
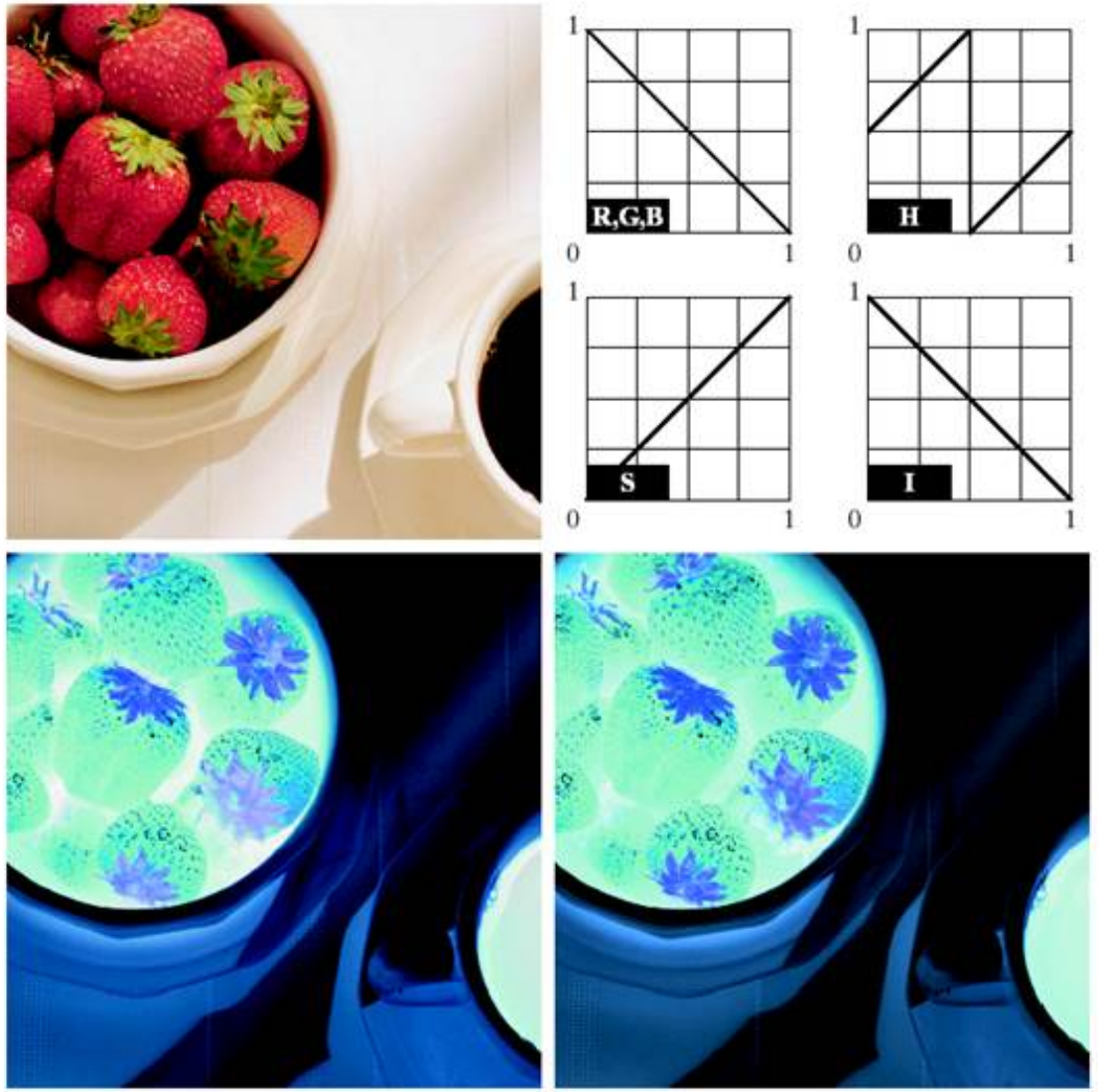


FIGURE 6.32
Complements on the color circle.

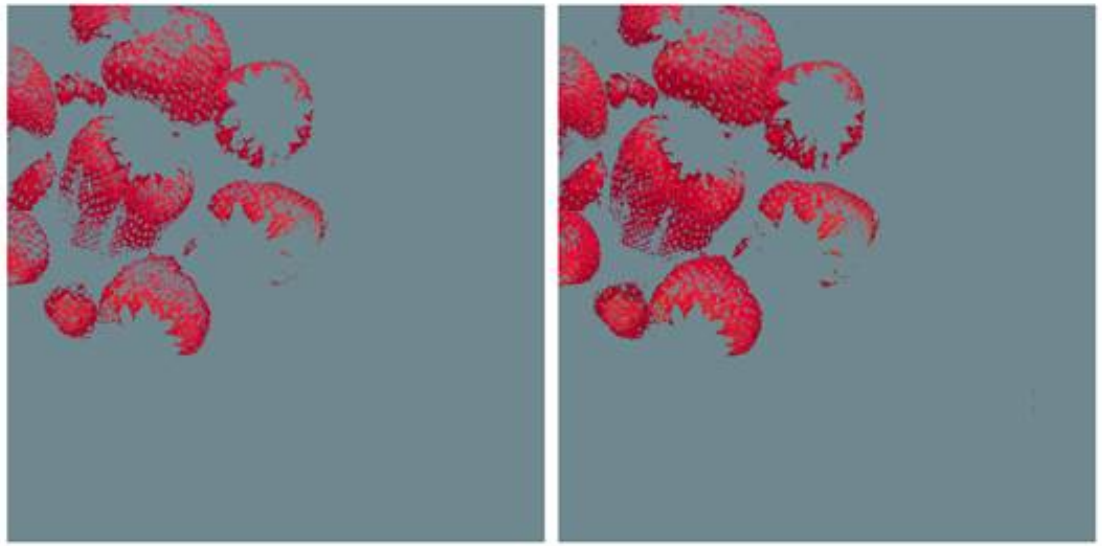
Chapter 6
Color Image Processing



a b
c d

FIGURE 6.33 Color complement transformations. (a) Original image. (b) Complement transformation functions. (c) Complement of (a) based on the RGB mapping functions. (d) An approximation of the RGB complement using HSI transformations.

Chapter 6
Color Image Processing



a b

FIGURE 6.34 Color slicing transformations that detect (a) reds within an RGB cube of width $W = 0.2549$ centered at (0.6863, 0.1608, 0.1922), and (b) reds within an RGB sphere of radius 0.1765 centered at the same point. Pixels outside the cube and sphere were replaced by color (0.5, 0.5, 0.5).

Chapter 6
Color Image Processing

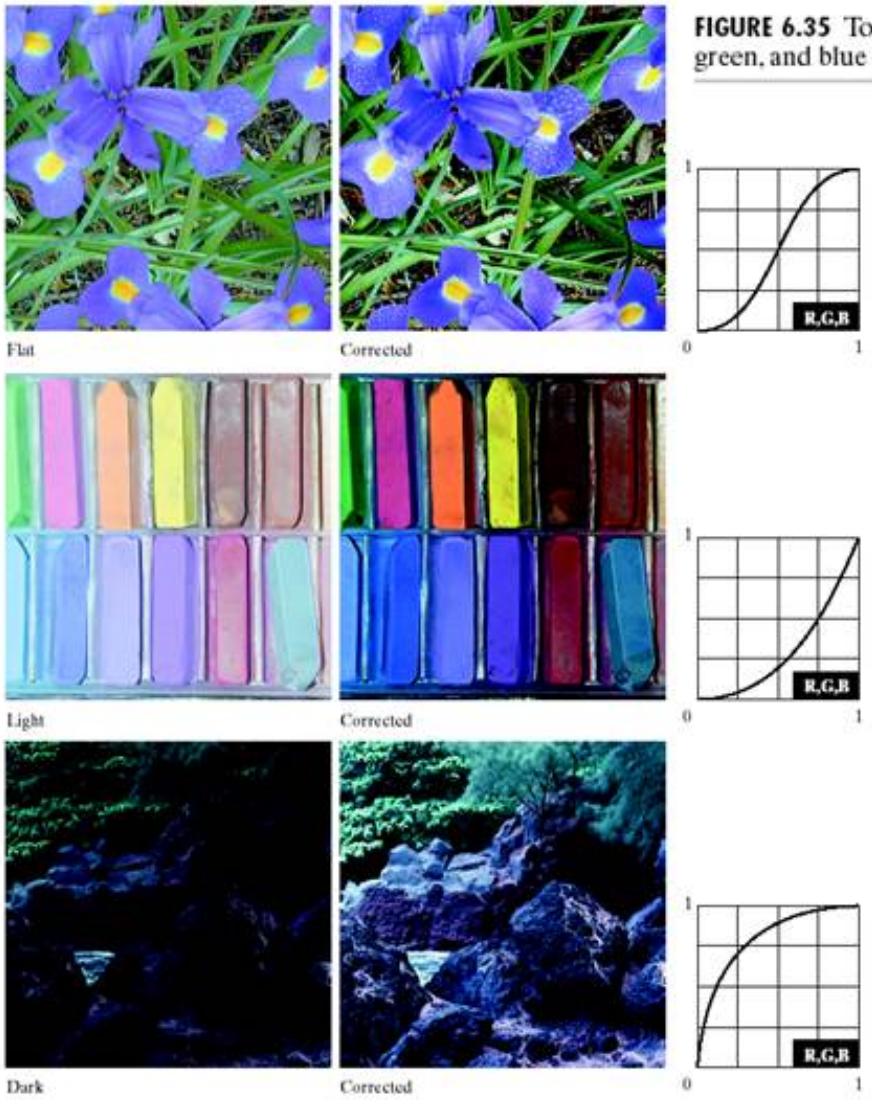


FIGURE 6.35 Tonal corrections for flat, light (high key), and dark (low key) color images. Adjusting the red, green, and blue components equally does not alter the image hues.

Chapter 6
Color Image Processing

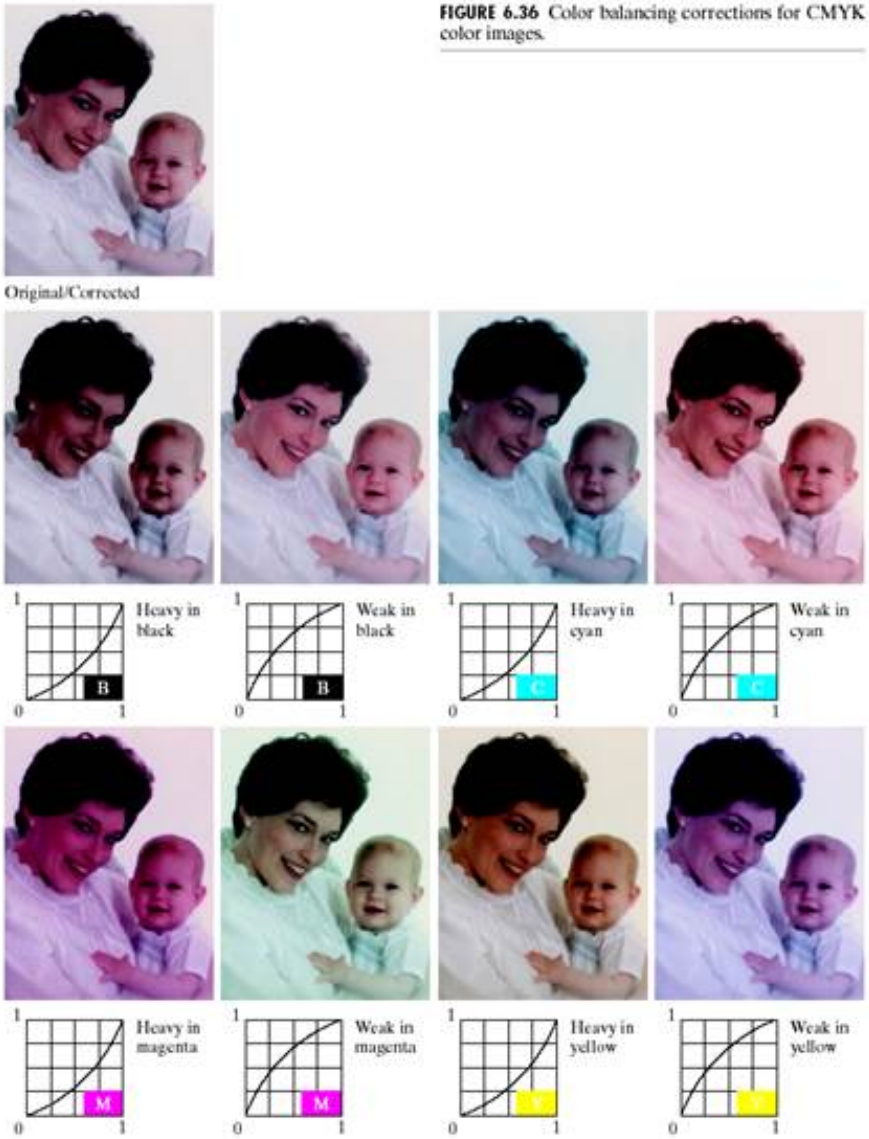
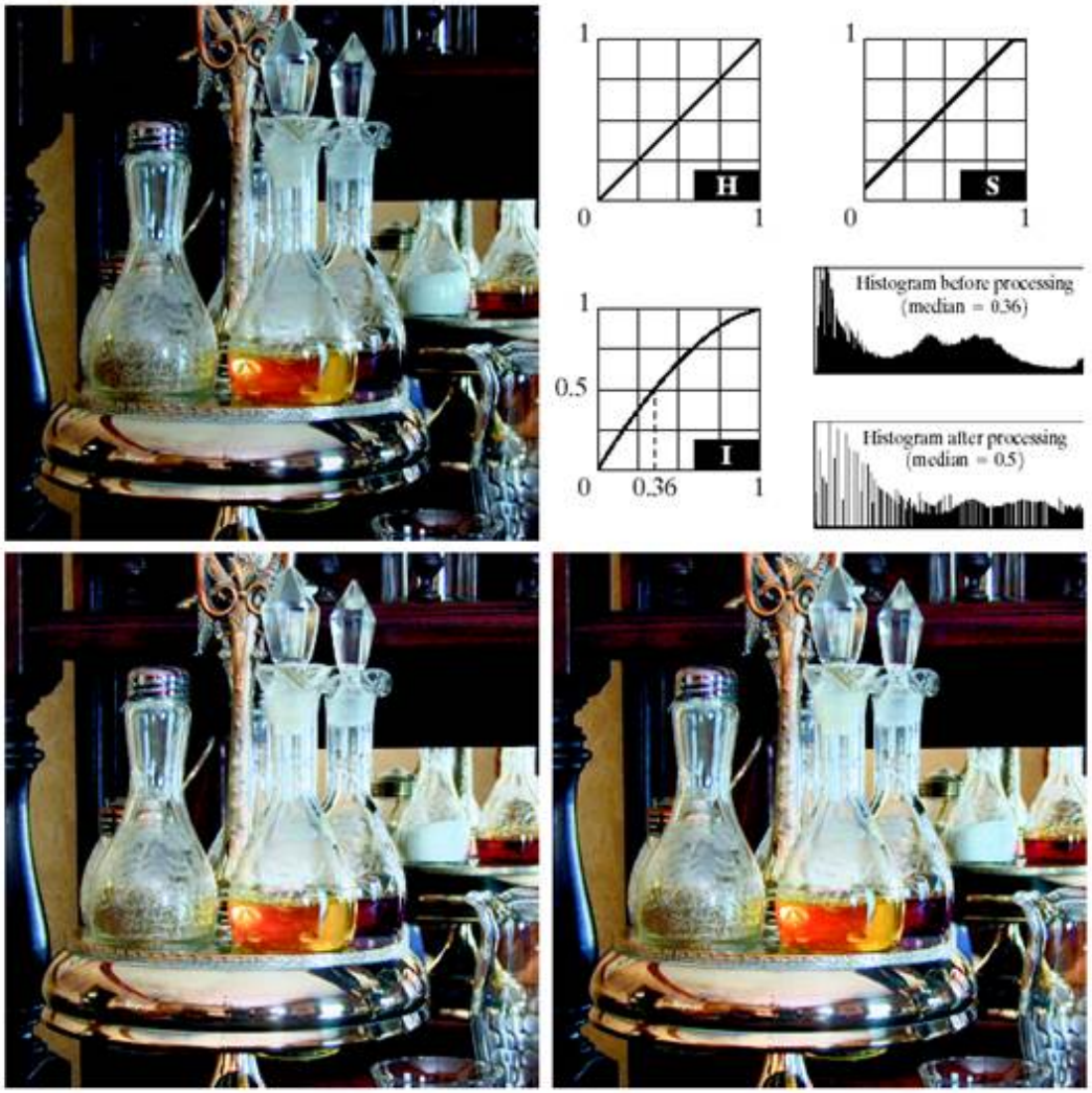


FIGURE 6.36 Color balancing corrections for CMYK color images.

Chapter 6
Color Image Processing



a b
c d

FIGURE 6.37
Histogram equalization (followed by saturation adjustment) in the HSI color space.

Chapter 6
Color Image Processing

a b
c d

FIGURE 6.38
(a) RGB image.
(b) Red component image.
(c) Green component.
(d) Blue component.

Chapter 6
Color Image Processing

a b c

FIGURE 6.39 HSI components of the RGB color image in Fig. 6.38(a). (a) Hue. (b) Saturation. (c) Intensity.

Chapter 6
Color Image Processing

a b c

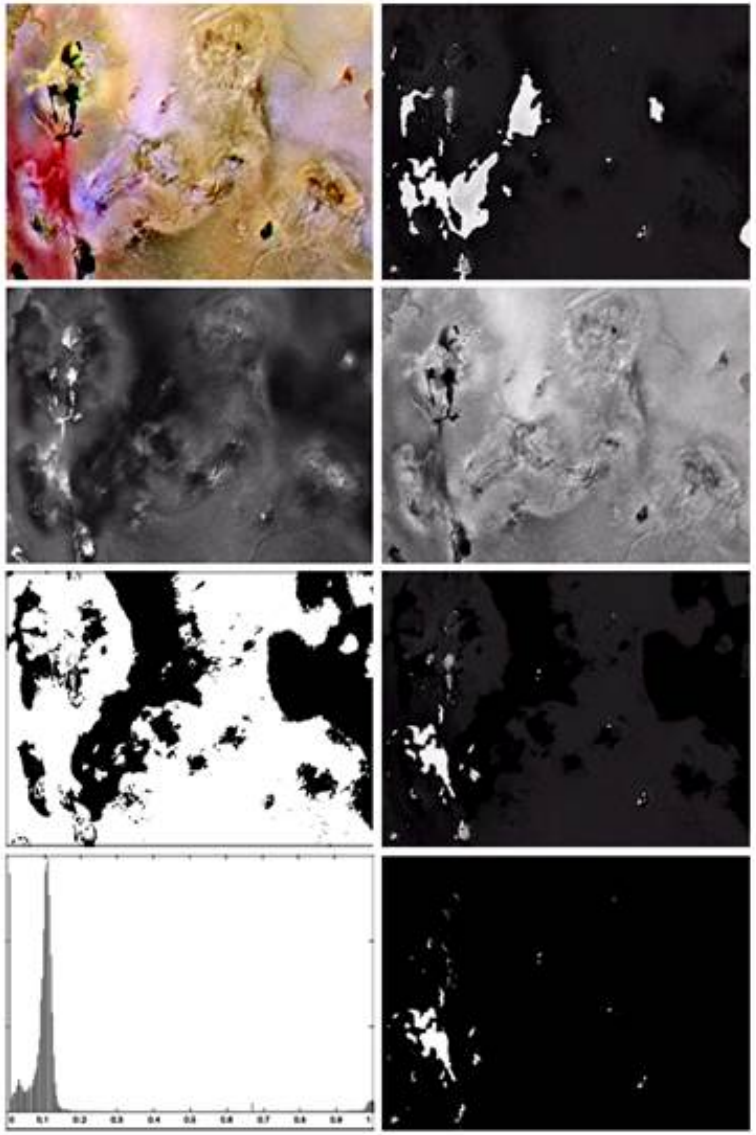
FIGURE 6.40 Image smoothing with a 5 × 5 averaging mask. (a) Result of processing each RGB component image. (b) Result of processing the intensity component of the HSI image and converting to RGB. (c) Difference between the two results.

Chapter 6
Color Image Processing

a b c

FIGURE 6.41 Image sharpening with the Laplacian. (a) Result of processing each RGB channel. (b) Result of processing the intensity component and converting to RGB. (c) Difference between the two results.

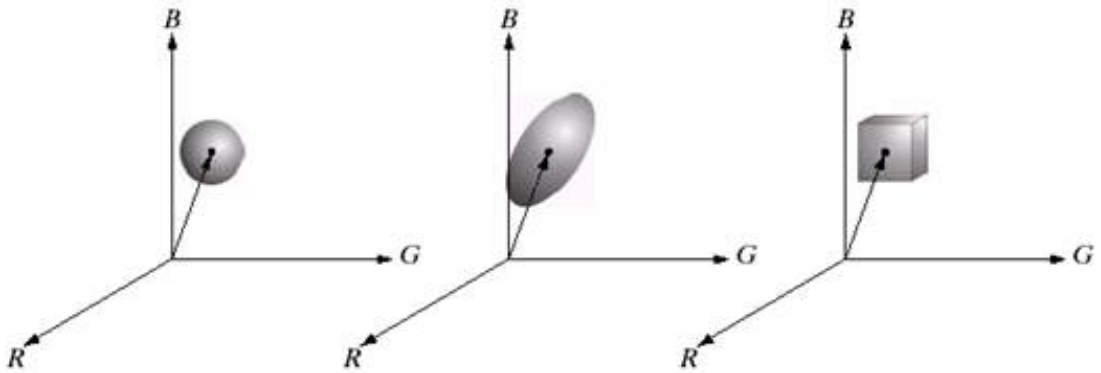
Chapter 6
Color Image Processing



a b
c d
e f
g h

FIGURE 6.42 Image segmentation in HSI space. (a) Original. (b) Hue. (c) Saturation. (d) Intensity. (e) Binary saturation mask (black = 0). (f) Product of (b) and (e). (g) Histogram of (f). (h) Segmentation of red components in (a).

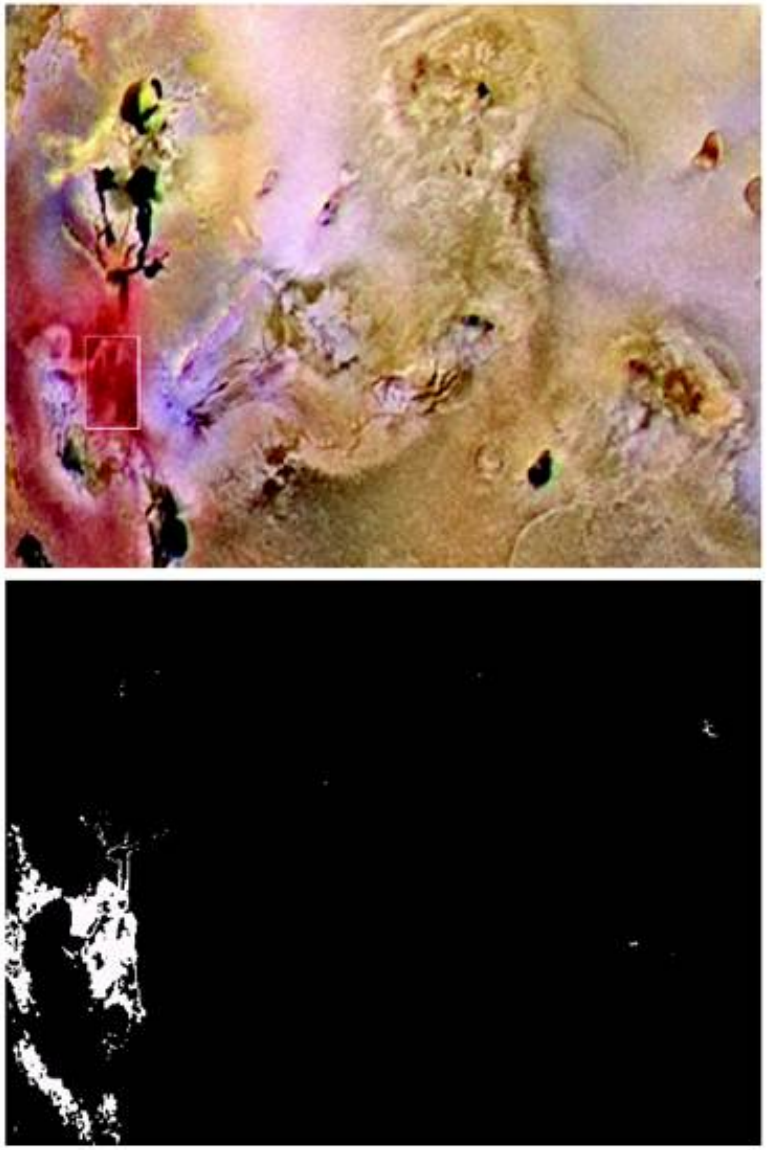
Chapter 6
Color Image Processing



a b c

FIGURE 6.43
Three approaches for enclosing data regions for RGB vector segmentation.

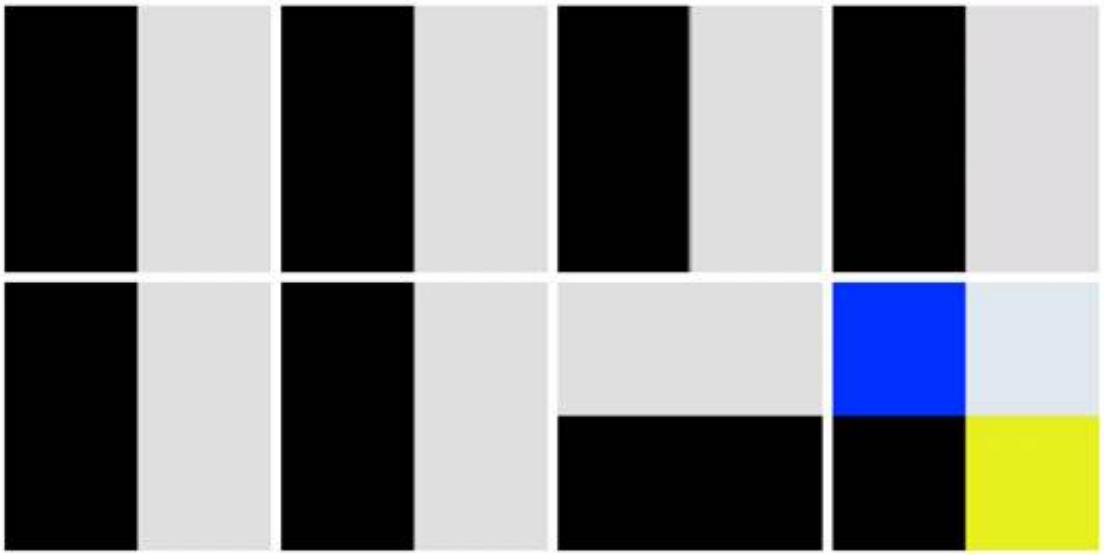
Chapter 6
Color Image Processing



a
b

FIGURE 6.44
Segmentation in RGB space. (a) Original image with colors of interest shown enclosed by a rectangle. (b) Result of segmentation in RGB vector space. Compare with Fig. 6.42(h).

Chapter 6
Color Image Processing



a b c d
e f g h

FIGURE 6.45 (a)–(c) R, G, and B component images and (d) resulting RGB color image. (f)–(g) R, G, and B component images and (h) resulting RGB color image.

Chapter 6
Color Image Processing

a b
c d

FIGURE 6.46
(a) RGB image.
(b) Gradient computed in RGB color vector space.
(c) Gradients computed on a per-image basis and then added.
(d) Difference between (b) and (c).

Chapter 6
Color Image Processing

a b c

FIGURE 6.47 Component gradient images of the color image in Fig. 6.46. (a) Red component, (b) green component, and (c) blue component. These three images were added and scaled to produce the image in Fig. 6.46(c).

Chapter 6
Color Image Processing

a b
c d

FIGURE 6.48
(a)–(c) Red, green, and blue component images corrupted by additive Gaussian noise of mean 0 and variance 800. (d) Resulting RGB image. [Compare (d) with Fig. 6.46(a).]

Chapter 6
Color Image Processing

a b c

FIGURE 6.49 HSI components of the noisy color image in Fig. 6.48(d). (a) Hue. (b) Saturation. (c) Intensity.

Chapter 6
Color Image Processing

a b
c d

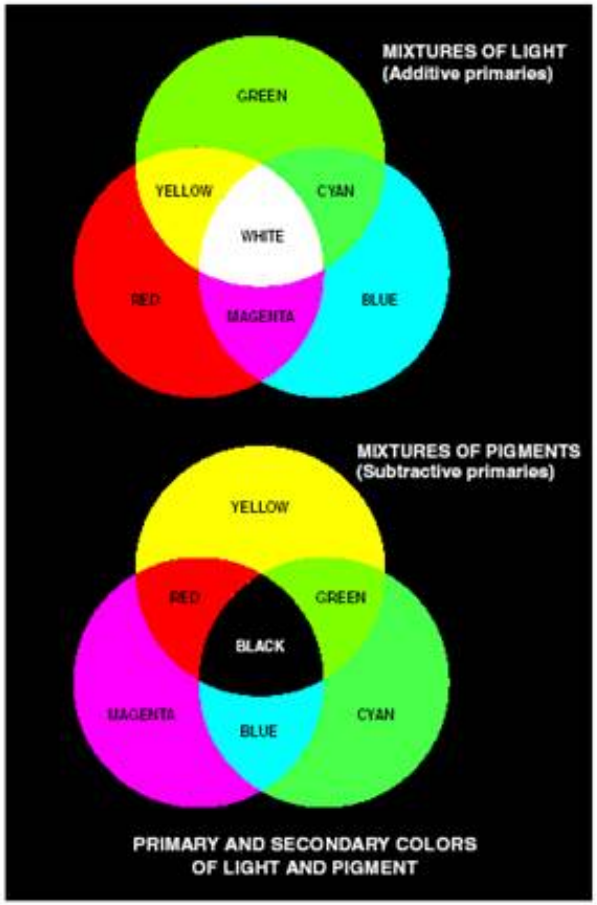
FIGURE 6.50
(a) RGB image with green plane corrupted by salt-and-pepper noise. (b) Hue component of HSI image. (c) Saturation component. (d) Intensity component.

Chapter 6
Color Image Processing

a b
c d

FIGURE 6.51
Color image compression. (a) Original RGB image. (b) Result of compressing and decompressing the image in (a).

Chapter 6
Color Image Processing



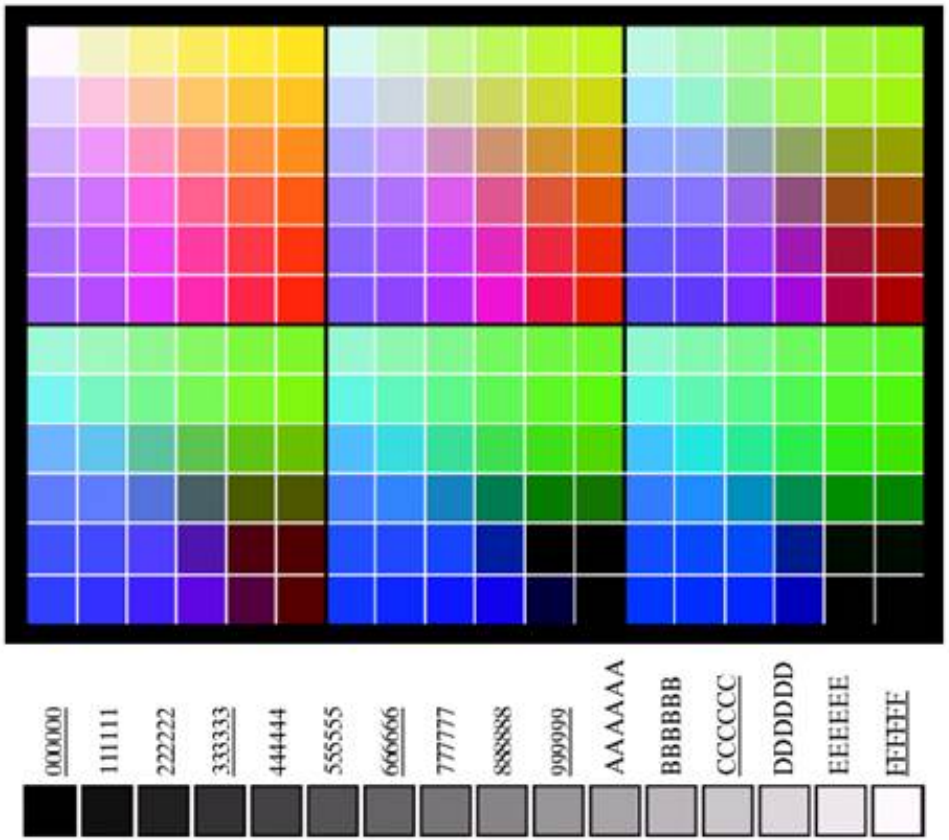
a
b

FIGURE 6.4 Primary and secondary colors of light and pigments. (Courtesy of the General Electric Co., Lamp Business Division.)

Chapter 6
Color Image Processing

Number System	Color Equivalents					
Hex	00	33	66	99	CC	FF
Decimal	0	51	102	153	204	255

TABLE 6.1
Valid values of each RGB component in a safe color.



a
b

FIGURE 6.10
(a) The 216 safe RGB colors. (b) All the grays in the 256-color RGB system (grays that are part of the safe color group are shown underlined).

Chapter 6
Color Image Processing

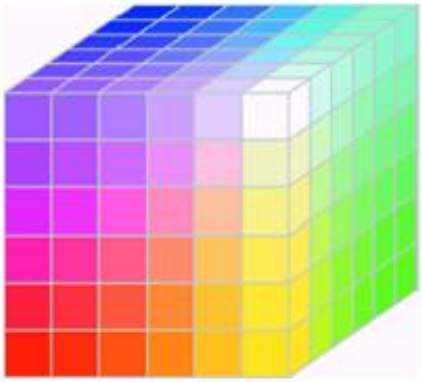


FIGURE 6.11 The RGB safe-color cube.