Digital Image Processing and Pattern Recognition

E1528

Lecture 2

DIP Fundamentals & MATLAB Tutorials

INSTRUCTOR

DR / AYMAN SOLIMAN

➢ Contents

What is Digital Image Processing?

- An image may be defined as a two-dimensional function, f(x, y), where x and y are spatial (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point.
- When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a digital image.
- The field of digital image processing refers to processing digital images by means of a digital computer.

➢ What is Digital Image Processing? (cont.)

- a digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are referred to as picture elements, image elements, pels, and pixels.
- Pixel is the term most widely used to denote the elements of a digital image.

The Origins of Digital Image Processing

- One of the first applications of digital images was in the newspaper industry, when pictures were first sent by submarine cable between London and New York.

- Some of the initial problems in improving the visual quality of these early digital pictures were related to the selection of printing procedures and the distribution of intensity levels.

Examples of Fields that Use Digital Image Processing

- Today, there is almost no area of technical endeavor that is not impacted in some way by digital image processing.
- Images based on radiation from the EM spectrum are the most familiar, especially images in the X-ray and visual bands of the spectrum.
- Electromagnetic waves can be conceptualized as propagating sinusoidal waves of varying wavelengths, or they can be thought of as a stream of massless particles, each traveling in a wavelike pattern and moving at the speed of light. Each massless particle contains a certain amount (or bundle) of energy. Each bundle of energy is called a photon.

➢ Examples of Fields that Use Digital Image Processing (cont.)

- ➢ If spectral bands are grouped according to energy per photon, we obtain the spectrum shown ranging from gamma rays (highest energy) at one end to radio waves (lowest energy) at the other.

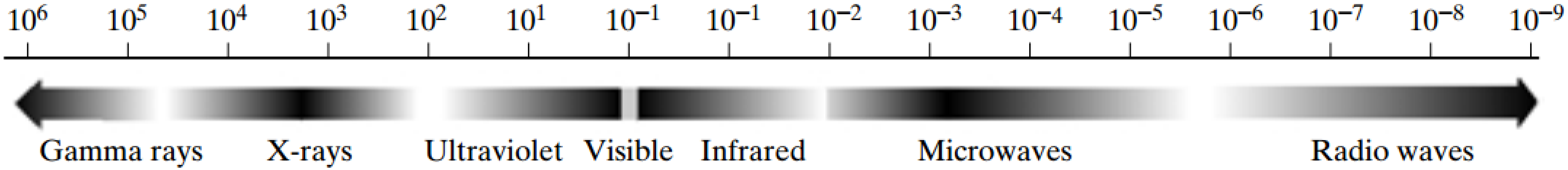


FIGURE The electromagnetic spectrum arranged according to energy per photon.

➢ Gamma-Ray Imaging

- ➢ Major uses of imaging based on gamma rays include nuclear medicine and astronomical observations. In nuclear medicine, the approach is to inject a patient with a radioactive isotope that emits gamma rays as it decays.

- ➢ Images are produced from the emissions collected by gamma ray detectors.

Figure (a) shows an image of a complete bone scan obtained by using gamma-ray imaging. Images of this sort are used to locate sites of bone pathology, such as infections or tumors.

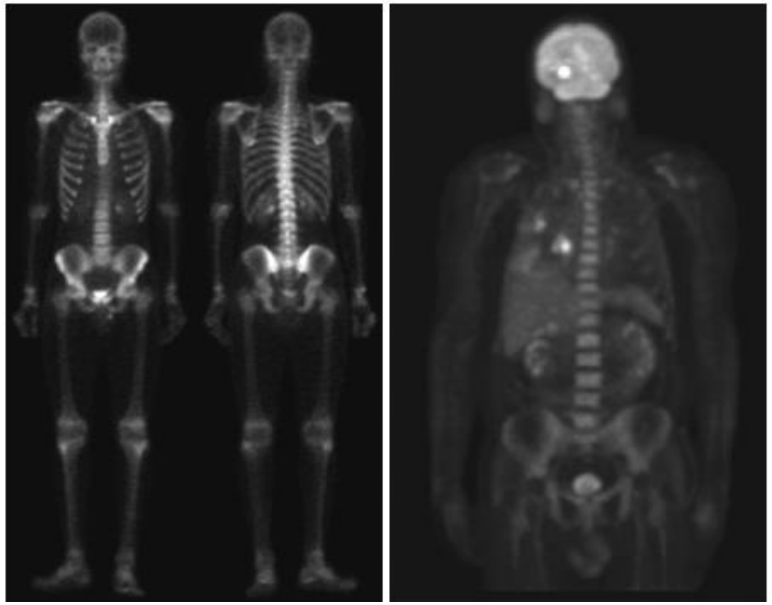
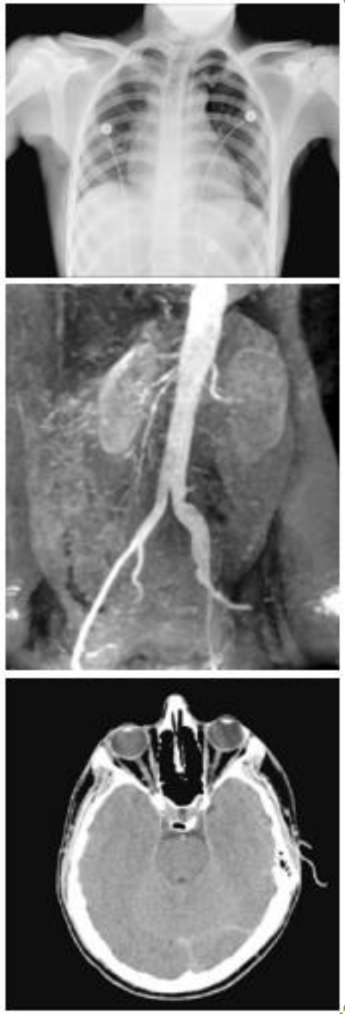


Figure (b) shows another major modality of nuclear imaging called positron emission tomography (PET).

X-ray Imaging

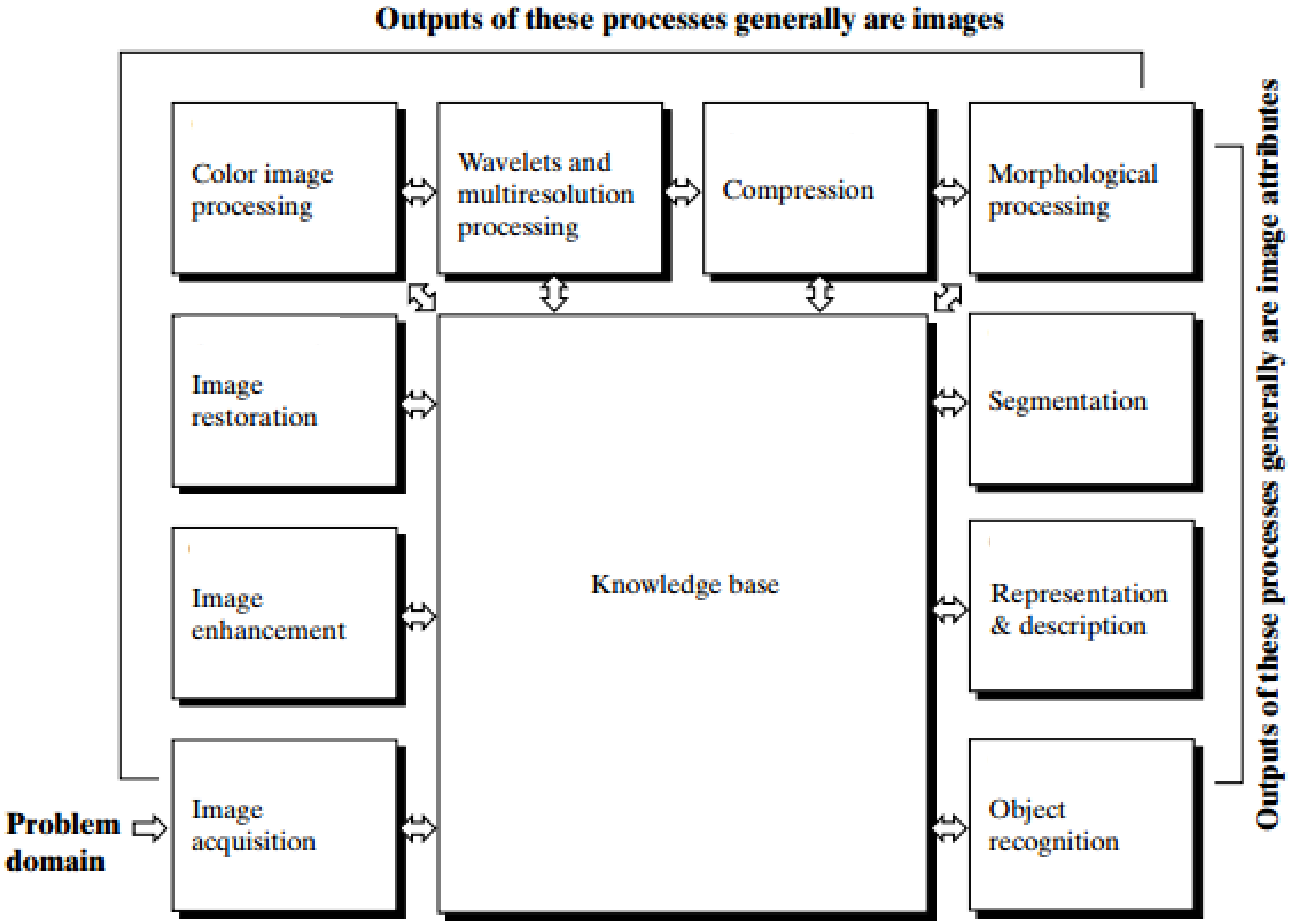
- X-rays are among the oldest sources of EM radiation used for imaging. The best-known use of X-rays is medical diagnostics, but they also are used extensively in industry and other areas, like astronomy.

- X-rays for medical and industrial imaging are generated using an X-ray tube, which is a vacuum tube with a cathode and anode.



Examples of X-ray imaging. (a) Chest X-ray. (b) Aortic angiogram. (c) Head CT.

➢ Fundamental steps in digital image processing.



MATLAB Tutorials

➢ Introduction to MATLAB

MATLAB : Matrix Laboratory

Numerical Computations with matrices

- ➢ Every number can be represented as matrix

Why MATLAB?

- ➢ User Friendly (GUI)
- ➢ Easy to work with
- ➢ Powerful tools for complex mathematics

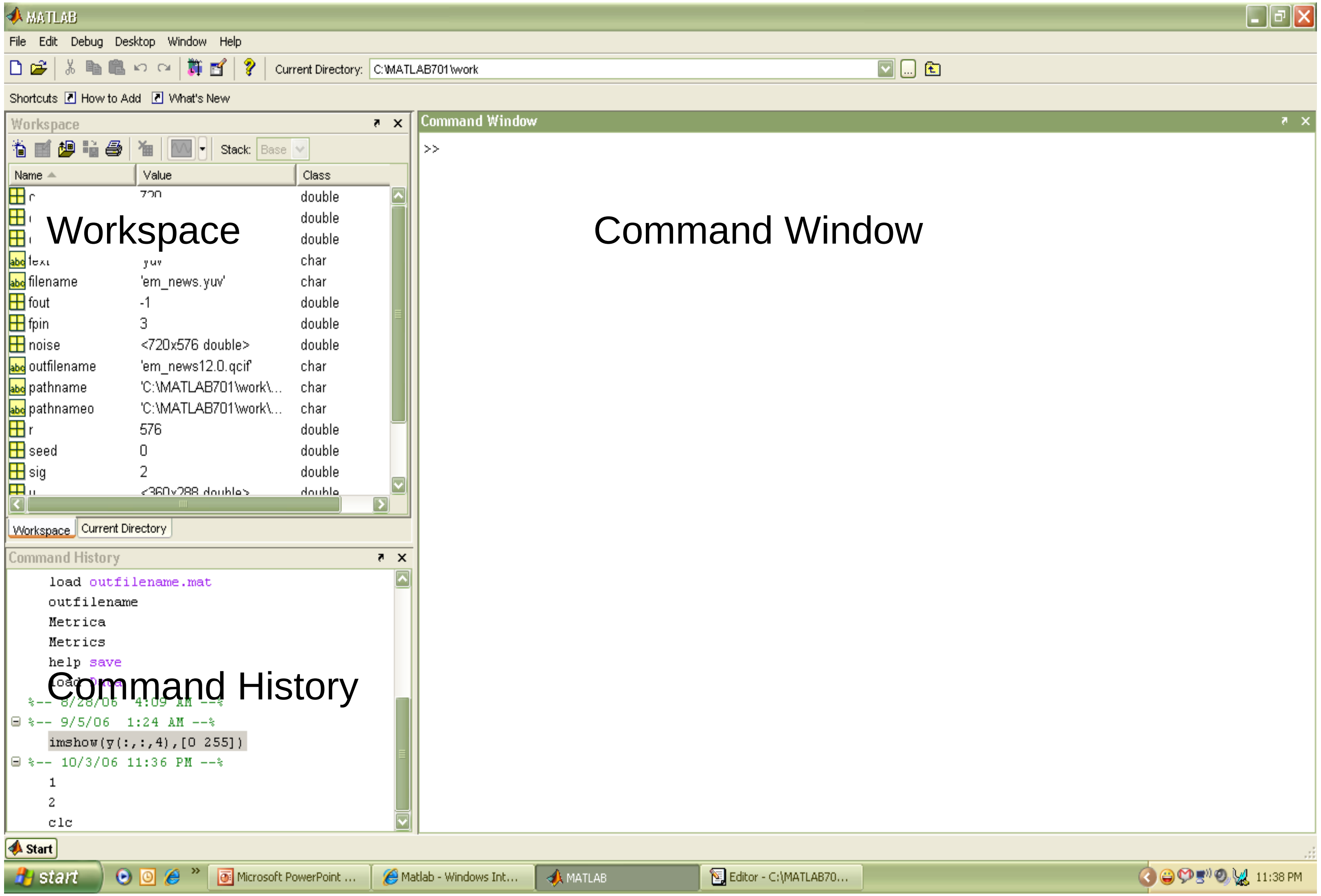


Fig: Snapshot of MATLAB

➢ Matrices in MATLAB

➢ To enter a matrix

$$\begin{matrix} 3 & 1 \\ 6 & 4 \end{matrix}$$

```
>> A = [3 1 ; 6 4]
>> A = [3, 1 ; 6, 4]
>> B = [3, 5 ; 0, 2]
```

➢ Basic Mathematical Operations

Addition:

>> C = A + B

Subtraction:

>> D = A – B

Multiplication:

>> E = A * B (Matrix multiplication)

>> E = A .* B (Element wise multiplication)

Division:

Left Division and Right Division

>> F = A . / B (Element wise division)

>> F = A / B (A * inverse of B)

>> F = A . \ B (Element wise division)

>> F = A \ B (inverse of A * B)

➢ Generating basic matrices

Matrix with ZEROS:

>> Z = ZEROS (r, c)

Matrix with ONES:

>> O = ONES (r, c)

IDENTITY Matrix:

>> I = EYE (r, c)

r Rows

c Columns

zeros, ones, eye → MATLAB functions

➢ Making the best from MATLAB

Need help ?

HELP <function name>

M files (.m)

To write and save MATLAB commands

Save time and easy to debug

Use of semicolon (;)

Comments (%)

Documentation

www.mathworks.com

➢ Image processing and MATLAB

- ➢ Easy to work with; as Images are matrices
- ➢ Built in functions for complex operations and algorithms (Ex. FFT, DCT, etc…)
- ➢ Image processing toolbox (?)
- ➢ Supports most image formats (.bmp, .jpg, .gif, .tiff, etc.…)

Format Name	Description	Recognized Extensions
TIFF	Tagged Image File Format	.tif, .tiff
JPEG	Joint Photographic Experts Group	.jpg, .jpeg
GIF	Graphics Interchange Format†	.gif
BMP	Windows Bitmap	.bmp
PNG	Portable Network Graphics	.png
XWD	X Window Dump	.xwd

➢ Image processing in MATLAB

➢ To read and display images

im = imread(“filename.fmt”)

im is (r * c) if gray scale

im is (r * c x 3) if color image (RGB)

imshow(im)…………………………………………..………..% displays image

imwrite(im, “filename.fmt”)……………………..………% writes image

➢ Working with complex numbers

➢ **real and imaginary**

real …………………. % real part of complex number

imag …………………% imaginary part of complex number

➢ **magnitude and phase**

abs ……………………...% magnitude of complex number

angle ………………....% phase of complex number

➢ MATLAB Commands

- f= imread(chest.jpg); …………… reading the image
- [r,c]= size(f); ………….……….. gives rows and columns dimension of image
- whos f ……………………….…. gives more information about image

```
Name          Size              Bytes          Class
f             1024x1024         1048576        uint8 array
Grand total is 1048576 elements using 1048576 bytes
```

- imshow (f,G) ……………. G is number of intensity levels if omitted it defaults to 256 levels.
- imshow (f,[low high]) ……. Displays as black all values less tan or equal low, and as white all values greater than or equal high
- imshow (f,[ ]) ……. Sets variable low to minimum value of array f and high to its maximum value

➤ Data Classes

Name	Description
double	Double-precision, floating-point numbers in the approximate range -10^{308} to 10^{308} (8 bytes per element).
uint8	Unsigned 8-bit integers in the range [0, 255] (1 byte per element).
uint16	Unsigned 16-bit integers in the range [0, 65535] (2 bytes per element).
uint32	Unsigned 32-bit integers in the range [0, 4294967295] (4 bytes per element).
int8	Signed 8-bit integers in the range [−128, 127] (1 byte per element).
int16	Signed 16-bit integers in the range [−32768, 32767] (2 bytes per element).
int32	Signed 32-bit integers in the range [−2147483648, 2147483647] (4 bytes per element).
single	Single-precision floating-point numbers with values in the approximate range -10^{38} to 10^{38} (4 bytes per element).
char	Characters (2 bytes per element).
logical	Values are 0 or 1 (1 byte per element).

Image Types

- Intensity images
- Binary images
- Indexed images
- RGB images

- Most monochrome images processing operations are carriedout using binary or intensity images, so our initial focus is on these two image types.

➢ Converting between data classes and image types

- ➢ Converting between data classes

B = data_class_name (A)

- ➢ Converting between image types

G = im2uint8 (f)

Name	Converts Input to:	Valid Input Image Data Classes
im2uint8	uint8	logical, uint8, uint16, and double
im2uint16	uint16	logical, uint8, uint16, and double
mat2gray	double (in range [0, 1])	double
im2double	double	logical, uint8, uint16, and double
im2bw	logical	uint8, uint16, and double

➢ Array Indexing

➢ Vector indexing

>> V = [1 3 5 7 9];

>> v(3)

ans = 5

Ex2:-

>> w = v' ;   transpose operator to convert row to column

Ex3:-

>>v(1:3)

ans = 1 3 5

➢ Array Indexing

```
>> v(3:end)
ans= 5 7 9


>> v(1:2:end)
ans= 1 5 9            mean starts wit 1 and jump with 2 to the end


>> v(end:-2:1)
ans= 9 5 1
```

➢ Matrix Indexing

```
>> A=[1 2 3;4 5 6;7 8 9];
```

```
>>A(2,3)
ans= 6
```

```
>>A(2,:)
Ans= 4 5 6
```

```
>>A(:,3)=0
Ans= 1 2 0
     4 5 0
     7 8 0
```

```
>>A(:,3)
Ans= 3
     6
     9
```

```
>>A(1:2,2:3)
Ans= 2 3
     5 6
```

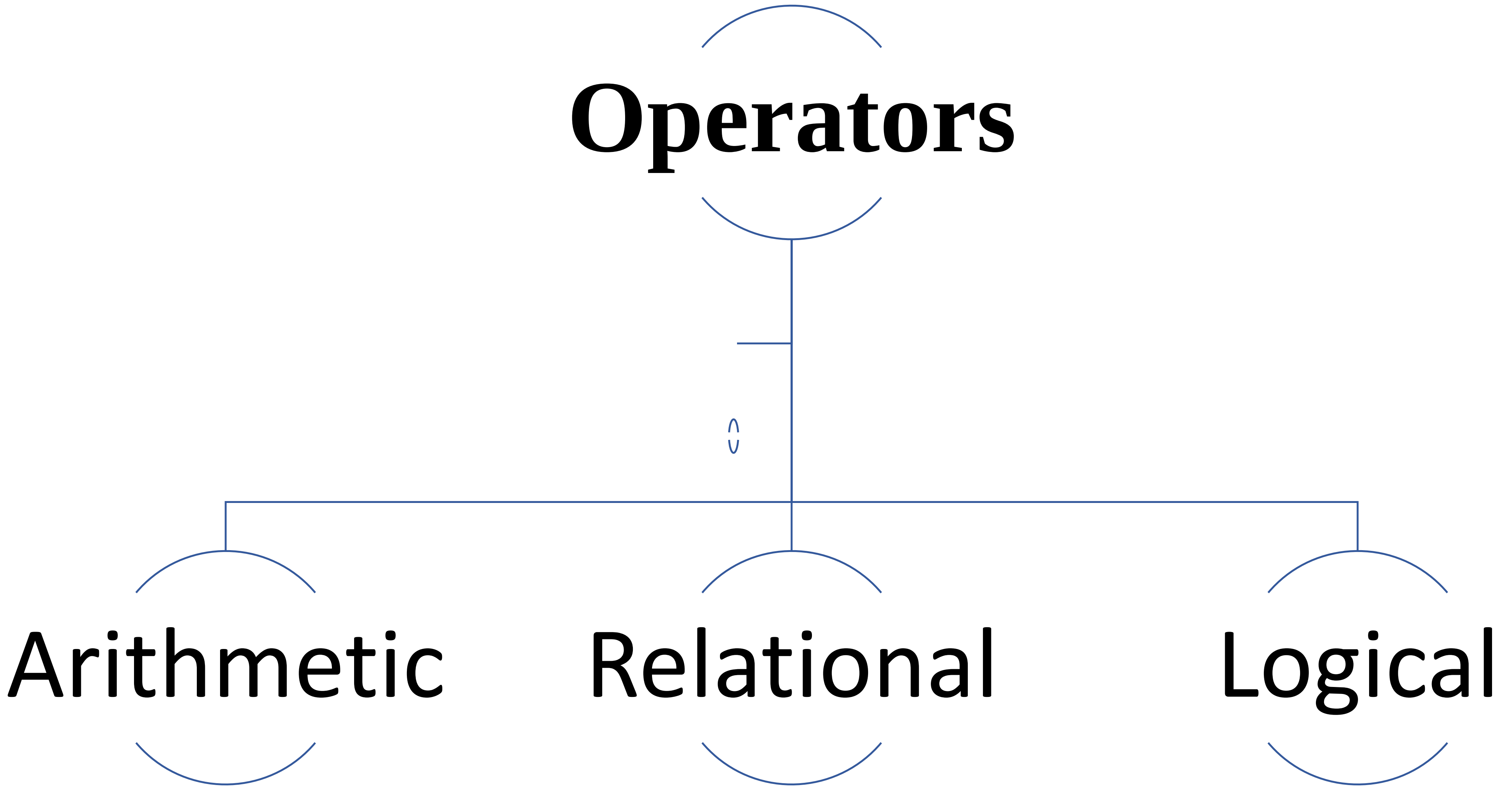
```
>>A(end,end)
Ans= 9
```

➢ Matrix Indexing

```
>> A(end, end-2)

Ans= 7


>> A(2:end , end:-2:1)

Ans= 6 4

     9 7


>>A([1 3] , [2 3])

Ans= 2 3

     8 9
```

➢ Important Standard Arrays

- `zeros(M, N)` generates an M × N matrix of 0s of class `double`.
- `ones(M, N)` generates an M × N matrix of 1s of class `double`.
- `true(M, N)` generates an M × N `logical` matrix of 1s.
- `false(M, N)` generates an M × N `logical` matrix of 0s.
- `magic(M)` generates an M × M "magic square." This is a square array in which the sum along any row, column, or main diagonal, is the same. Magic squares are useful arrays for testing purposes because they are easy to generate and their numbers are integers.
- `rand(M, N)` generates an M × N matrix whose entries are uniformly distributed random numbers in the interval [0, 1].
- `randn(M, N)` generates an M × N matrix whose numbers are normally distributed (i.e., Gaussian) random numbers with mean 0 and variance 1.


Operators
Arithmetic
Relational
Logical

➢ Arithmetic operators

Operator	Name	MATLAB Function	Comments and Examples
+	Array and matrix addition	plus(A, B)	a + b, A + B, or a + A.
−	Array and matrix subtraction	minus(A, B)	a − b, A − B, A − a, or a − A.
.*	Array multiplication	times(A, B)	C = A.*B, C(I, J) = A(I, J)*B(I, J).
*	Matrix multiplication	mtimes(A, B)	A*B, standard matrix multiplication, or a*A, multiplication of a scalar times all elements of A.
./	Array right division	rdivide(A, B)	C = A./B, C(I, J) = A(I, J)/B(I, J).
.\	Array left division	ldivide(A, B)	C = A.\B, C(I, J) = B(I, J)/A(I, J).
/	Matrix right division	mrdivide(A, B)	A/B is roughly the same as A*inv(B), depending on computational accuracy.
\	Matrix left division	mldivide(A, B)	A\B is roughly the same as inv(A)*B, depending on computational accuracy.
.^	Array power	power(A, B)	If C = A.^B, then C(I, J) = A(I, J)^B(I, J).
^	Matrix power	mpower(A, B)	See online help for a discussion of this operator.
.'	Vector and matrix transpose	transpose(A)	A.'. Standard vector and matrix transpose.
'	Vector and matrix complex conjugate transpose	ctranspose(A)	A'. Standard vector and matrix conjugate transpose. When A is real A.' = A'.
+	Unary plus	uplus (A)	+A is the same as 0 + A.
−	Unary minus	uminus (A)	−A is the same as 0 − A or −1*A.
:	Colon		Discussed in Section 2.8.

➢ Logical operators

Function	Comments
`xor` (exclusive OR)	The `xor` function returns a 1 only if both operands are logically different; otherwise `xor` returns a 0.
`all`	The `all` function returns a 1 if all the elements in a vector are nonzero; otherwise `all` returns a 0. This function operates columnwise on matrices.
`any`	The `any` function returns a 1 if any of the elements in a vector is nonzero; otherwise `any` returns a 0. This function operates columnwise on matrices.

➢ Flow control

Statement	Description
`if`	`if`, together with `else` and `elseif`, executes a group of statements based on a specified logical condition.
`for`	Executes a group of statements a fixed (specified) number of times.
`while`	Executes a group of statements an indefinite number of times, based on a specified logical condition.
`break`	Terminates execution of a `for` or `while` loop.
`continue`	Passes control to the next iteration of a `for` or `while` loop, skipping any remaining statements in the body of the loop.
`switch`	`switch`, together with `case` and `otherwise`, executes different groups of statements, depending on a specified value or string.
`return`	Causes execution to return to the invoking function.
`try...catch`	Changes flow control if an error is detected during execution.

➢ Plotting / displaying

➢ **PLOT(x,y)**

Plots y versus x.

Linear plot

XLABEL(‘label’)

YLABEL(‘label’)

TITLE(‘title’)

➢ **IMAGE(x)**

Displays image

➢ **3D PLOT:**

MESH

3D mesh surface (Ex. filters)

MESHGRID

Useful in 3D plots

SURF

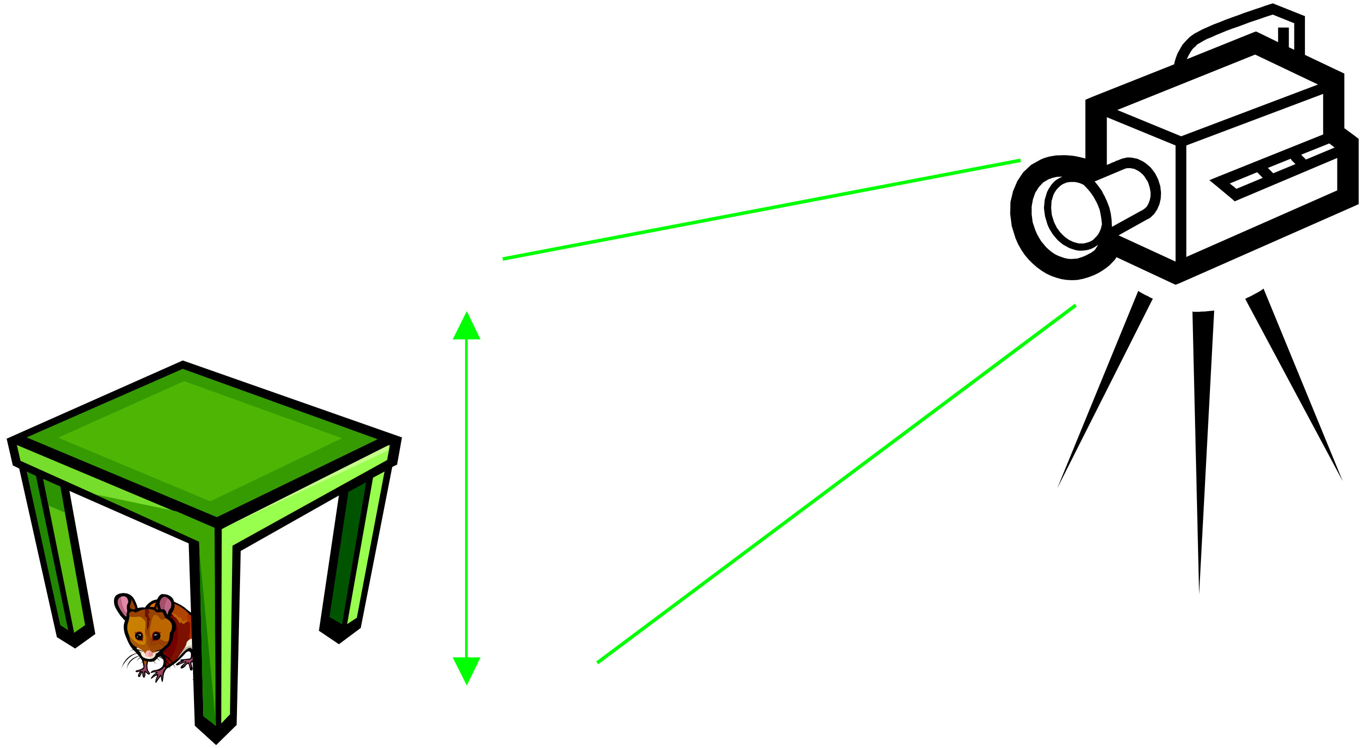
3D colored surface (Ex. filters)

Digital Image Processing

➢ Introduction to Digital Image Processing - Fundamentals

Scales of Imaging

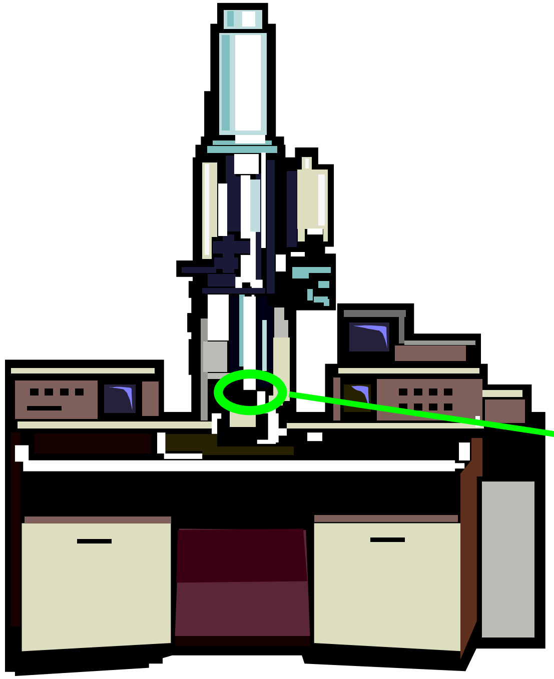
… to the everyday …

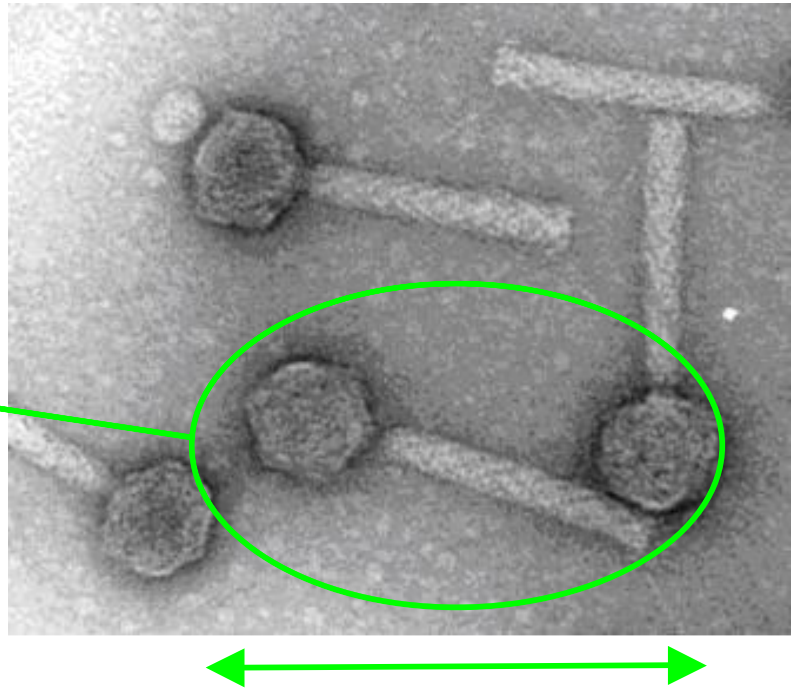


➢ Introduction to Digital Image Processing - Fundamentals

Scales of Imaging

… to the tiny …





➢ Digital Image Formation

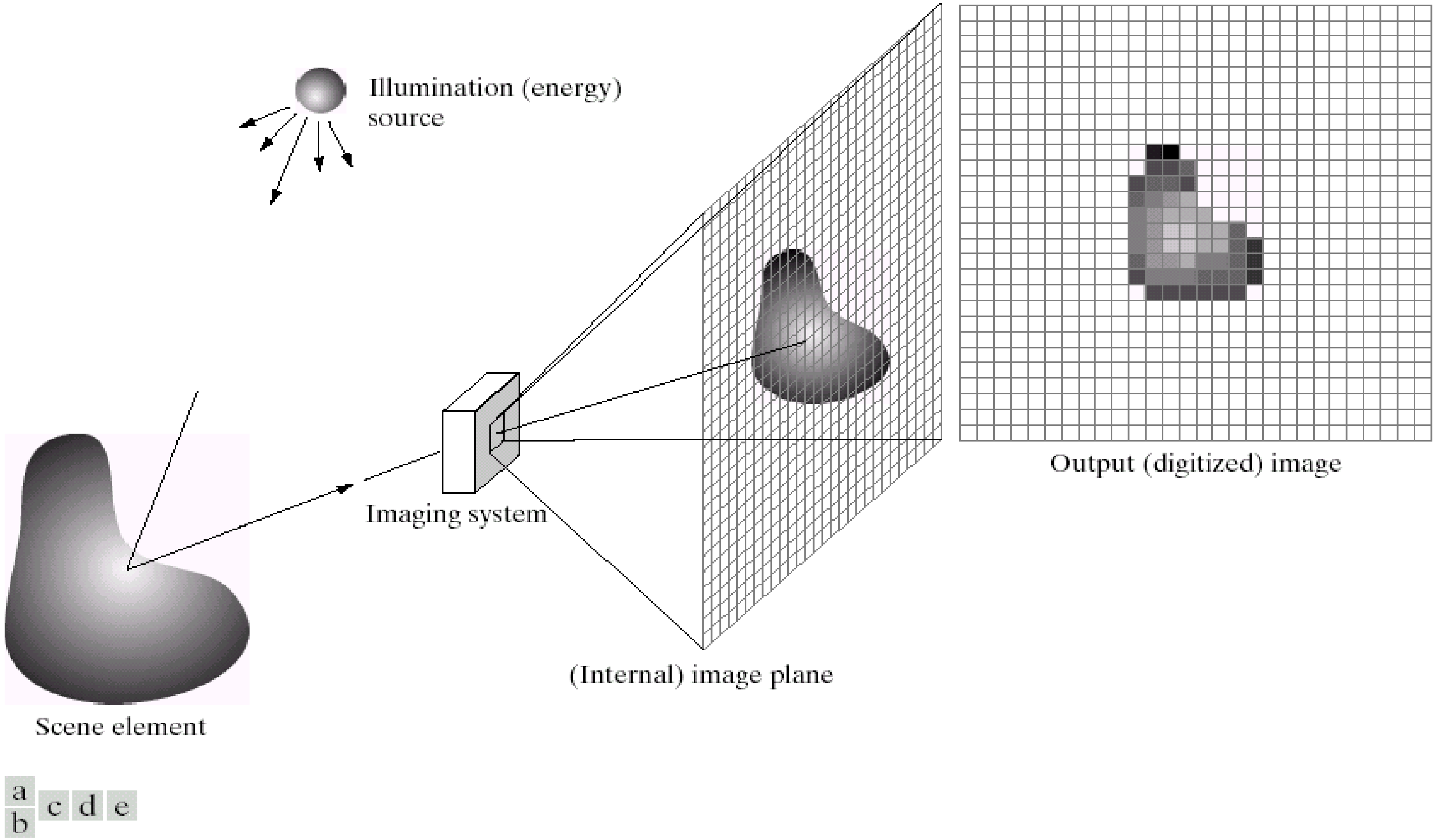


FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

From [Gonzalez & Woods]

➢ Matrix Representation

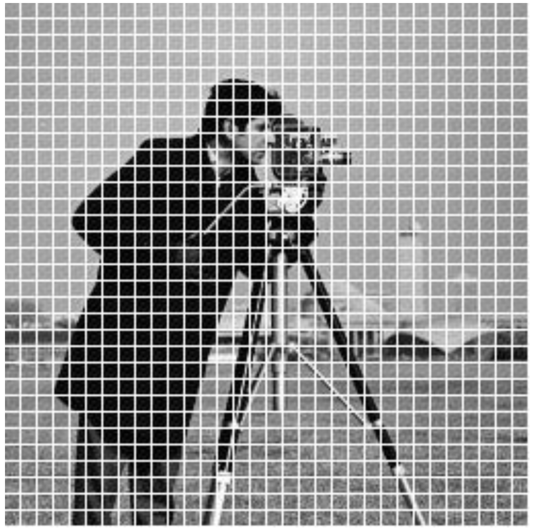
$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

183	160	94	153	194	163	132	165
183	153	116	176	187	166	130	169
179	168	171	182	179	170	131	167
177	177	179	177	179	165	131	167
178	178	179	176	182	164	130	171
179	180	180	179	183	169	132	169
179	179	180	182	183	170	129	173
180	179	181	179	181	170	130	169

H=256

W=256

Divide into 8x8 blocks



From [Gonzalez & Woods]

➢ Image Resolution

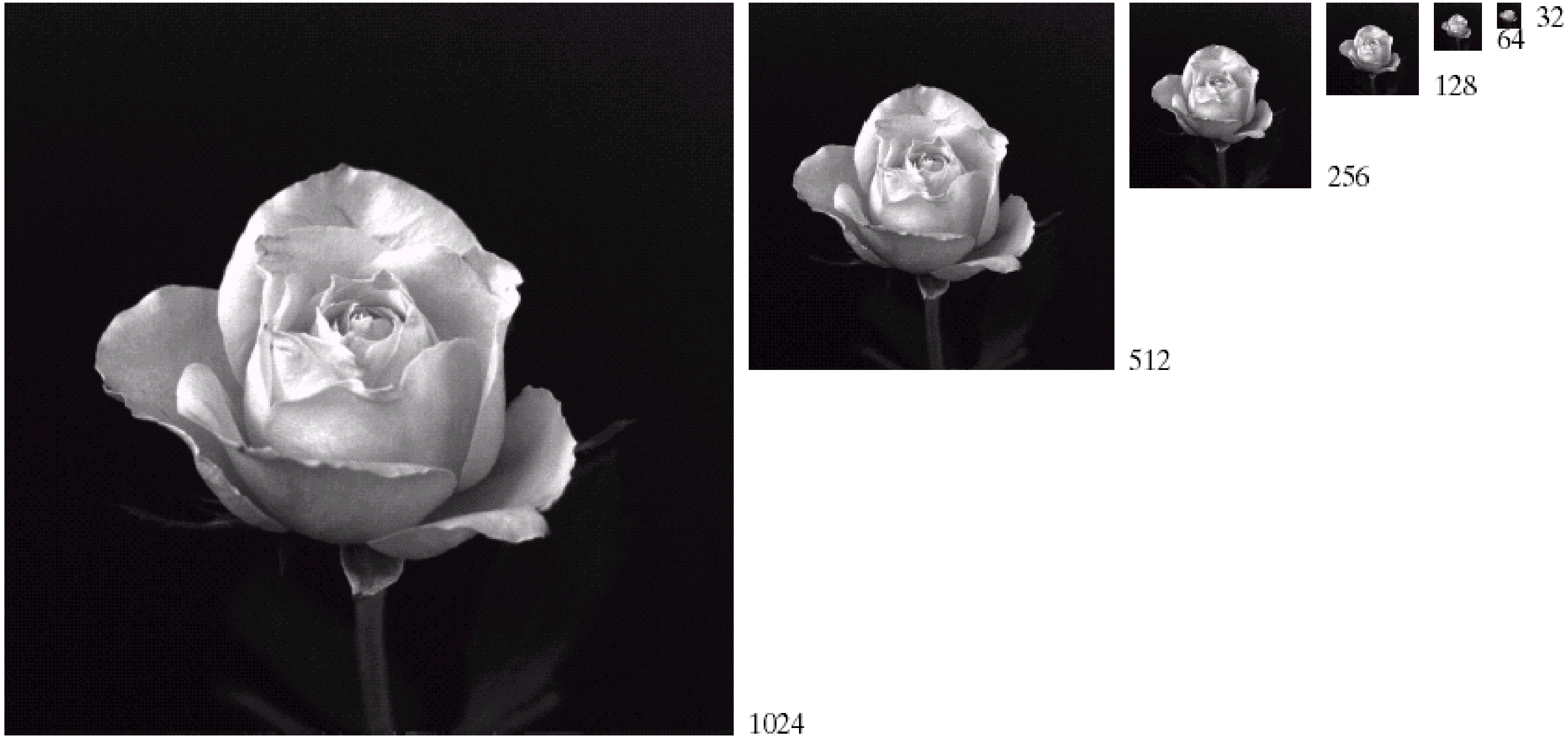


FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

From [Gonzalez & Woods]

➢ Image Resolution

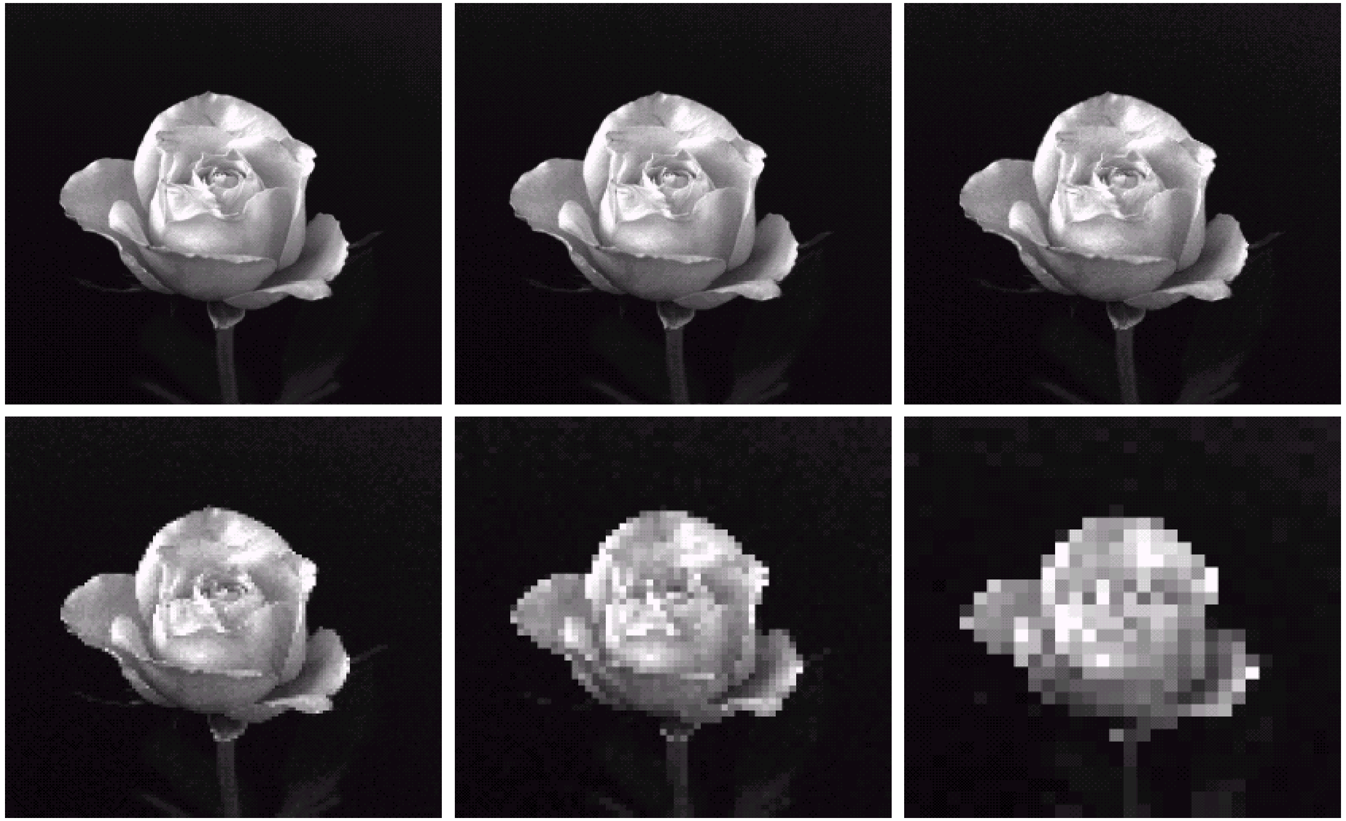


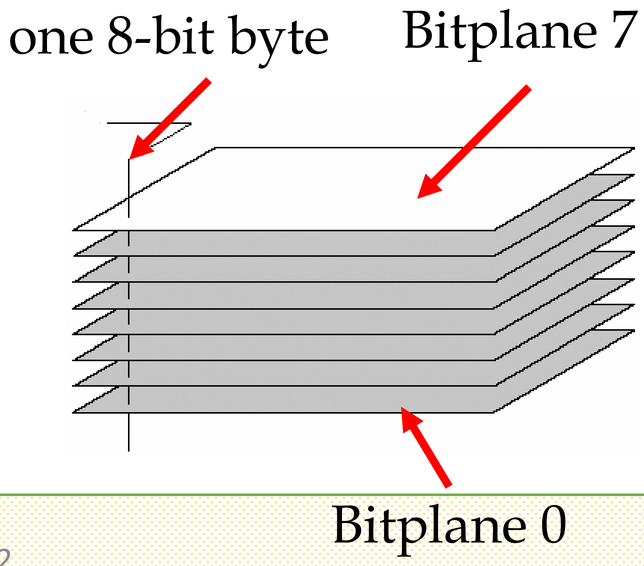
FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

➢ Bitplanes

Original 8bits/pixel

Bitplane 7

Bitplane 6



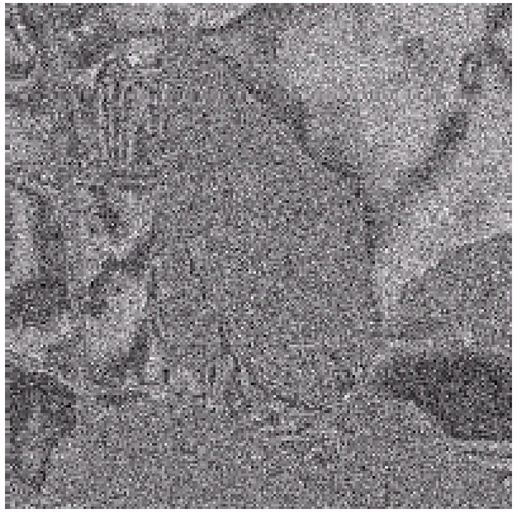
Bitplane 5

Bitplane 4

➢ Bitplanes

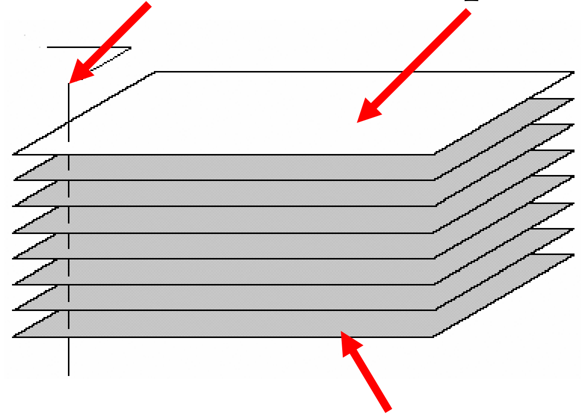
Original 8bits/pixel

Bitplane 3

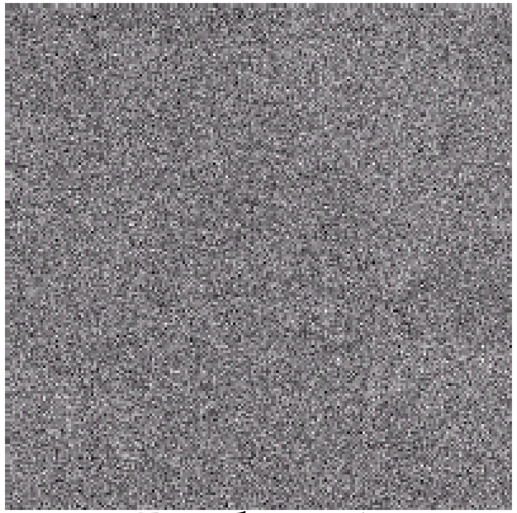


Bitplane 2

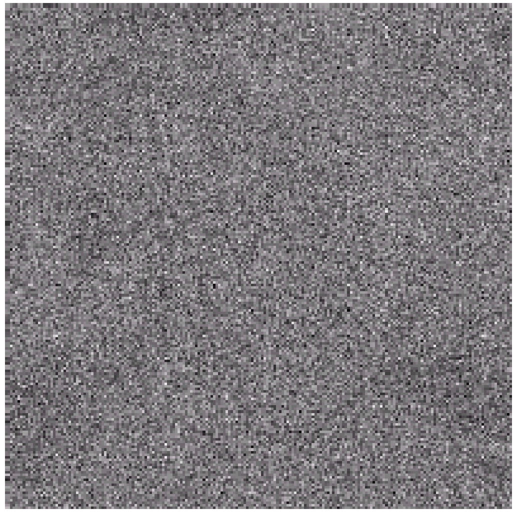
one 8-bit byte Bitplane 7



Bitplane 0



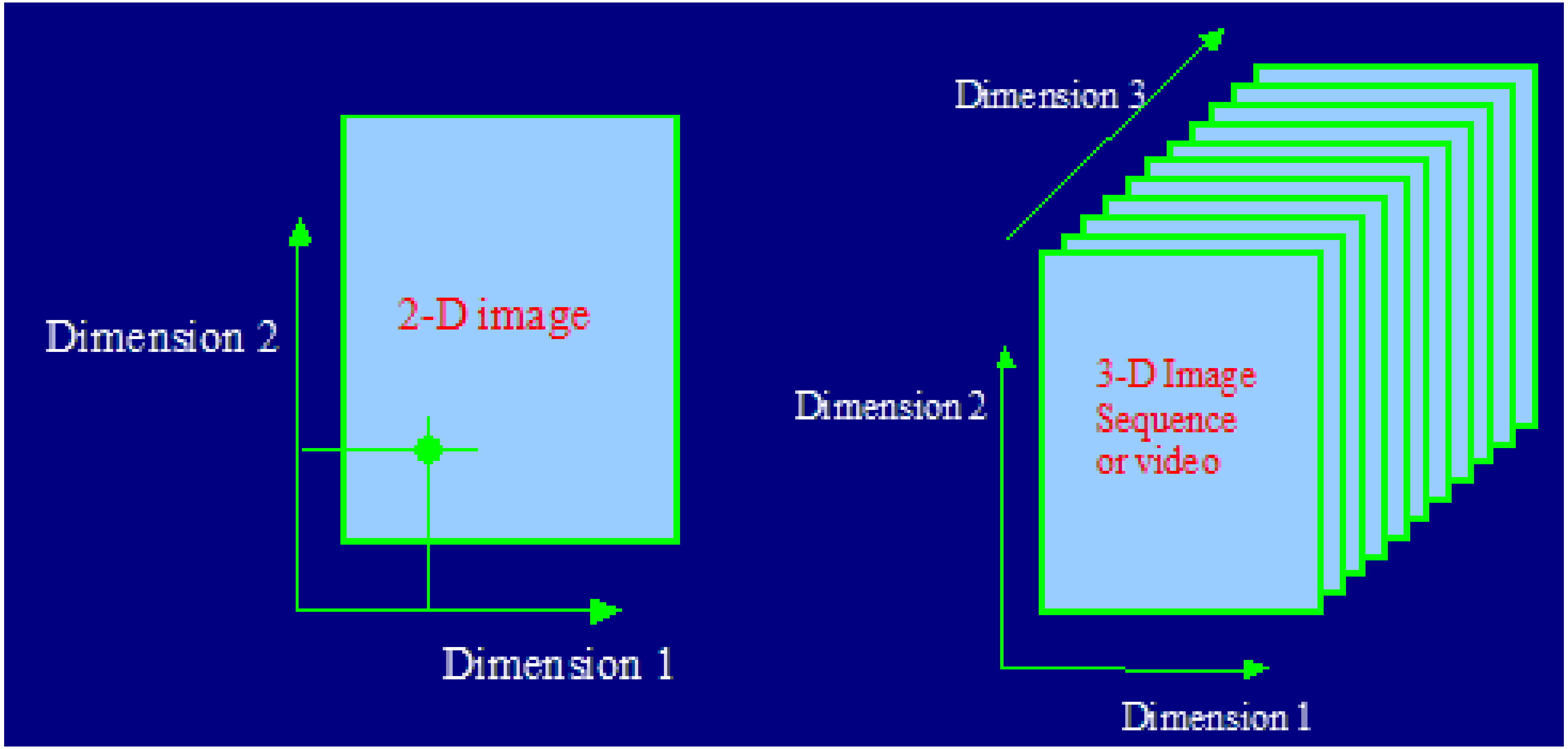
Bitplane 1



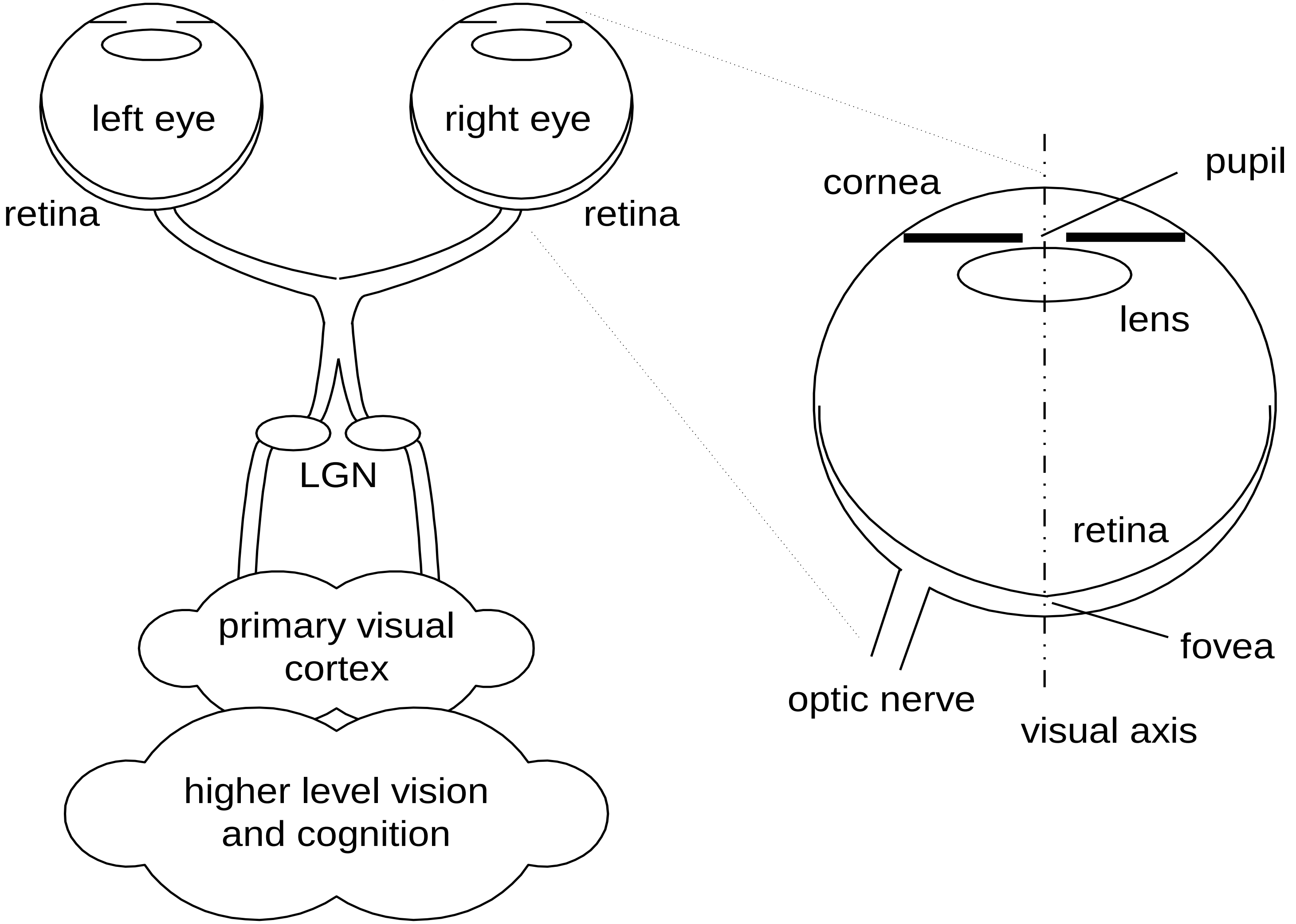
Bitplane 0

➢ Dimensionality of Digital Images

➢ Images and videos are multi-dimensional (≥ 2 dimensions) signals.



➢ The Human Visual System (HVS)



➢ HVS: Foveated Vision

- ➢ Foveated vision: non-uniform resolution of the visual field, highest at the point of fixation and decreasing rapidly

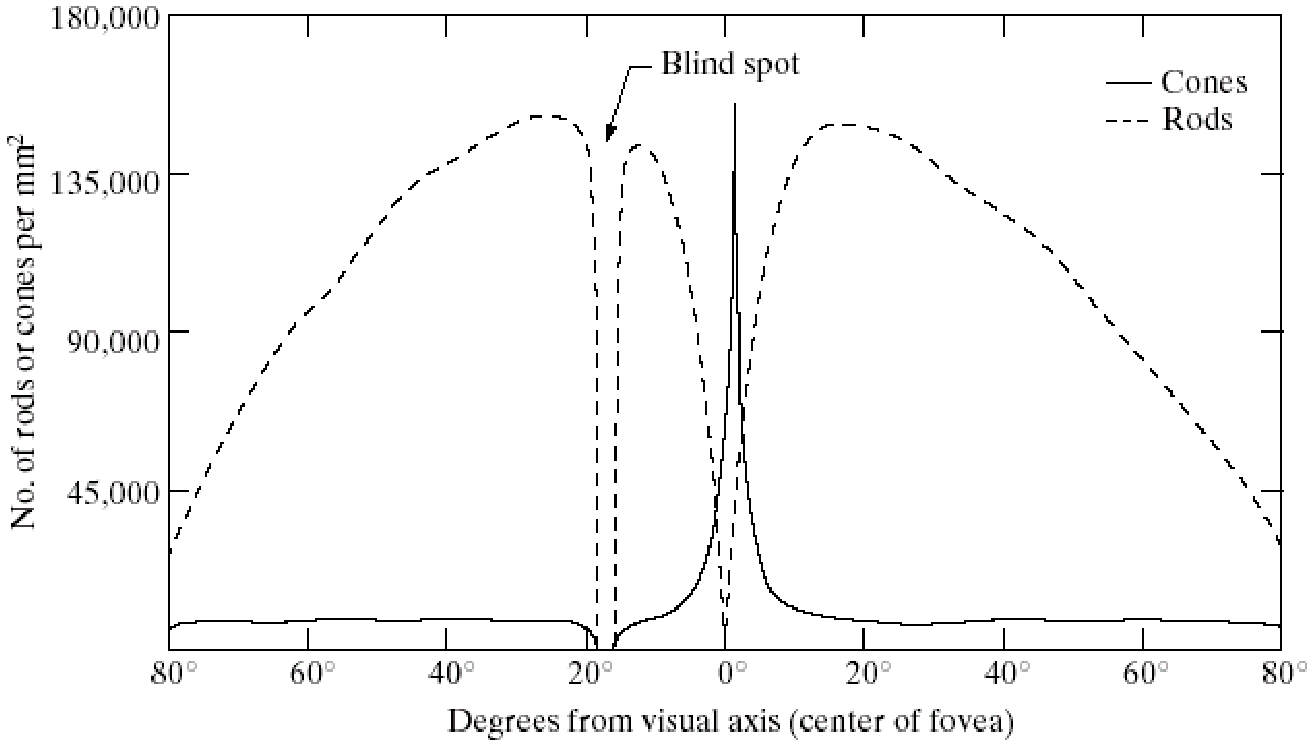
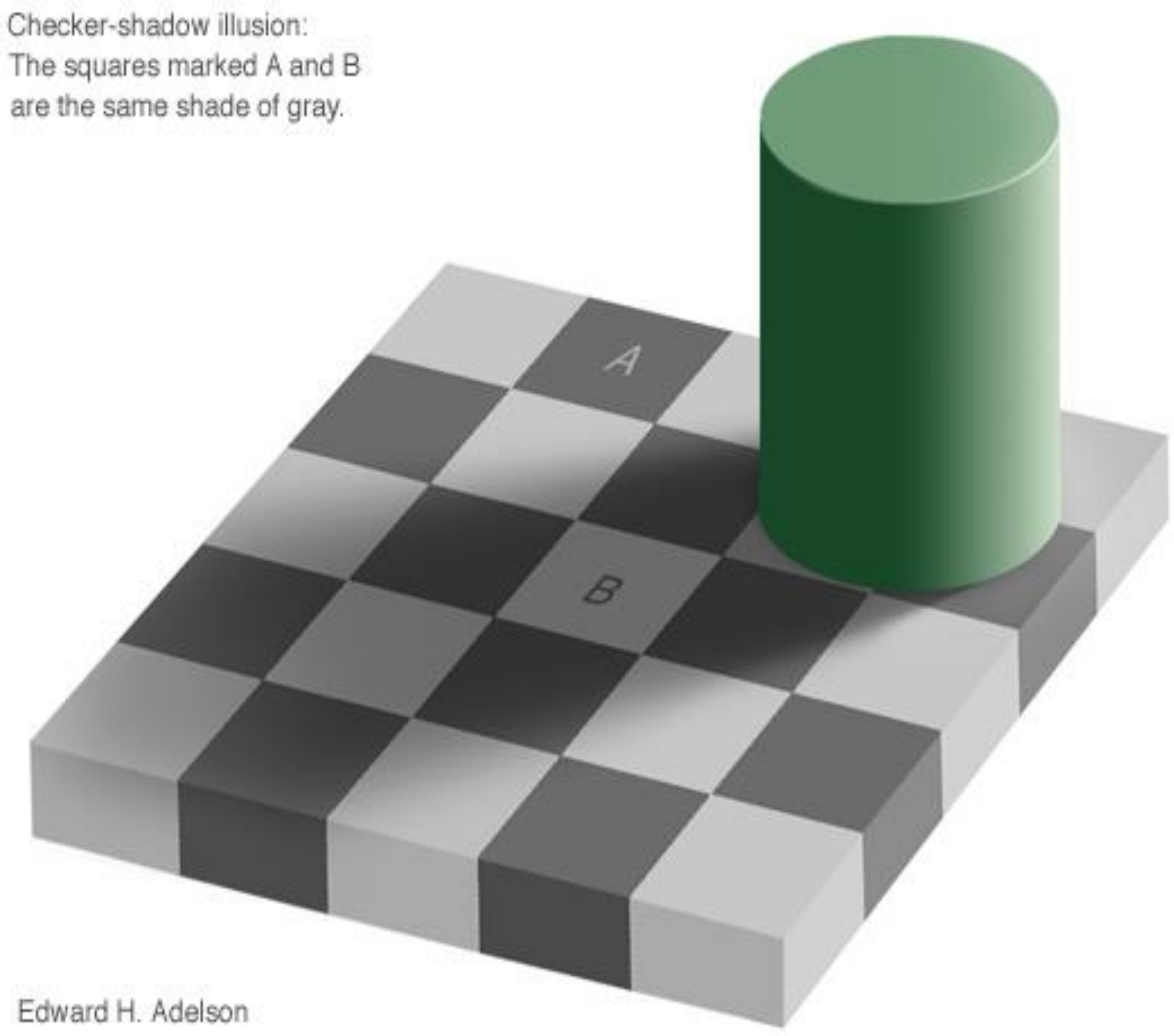
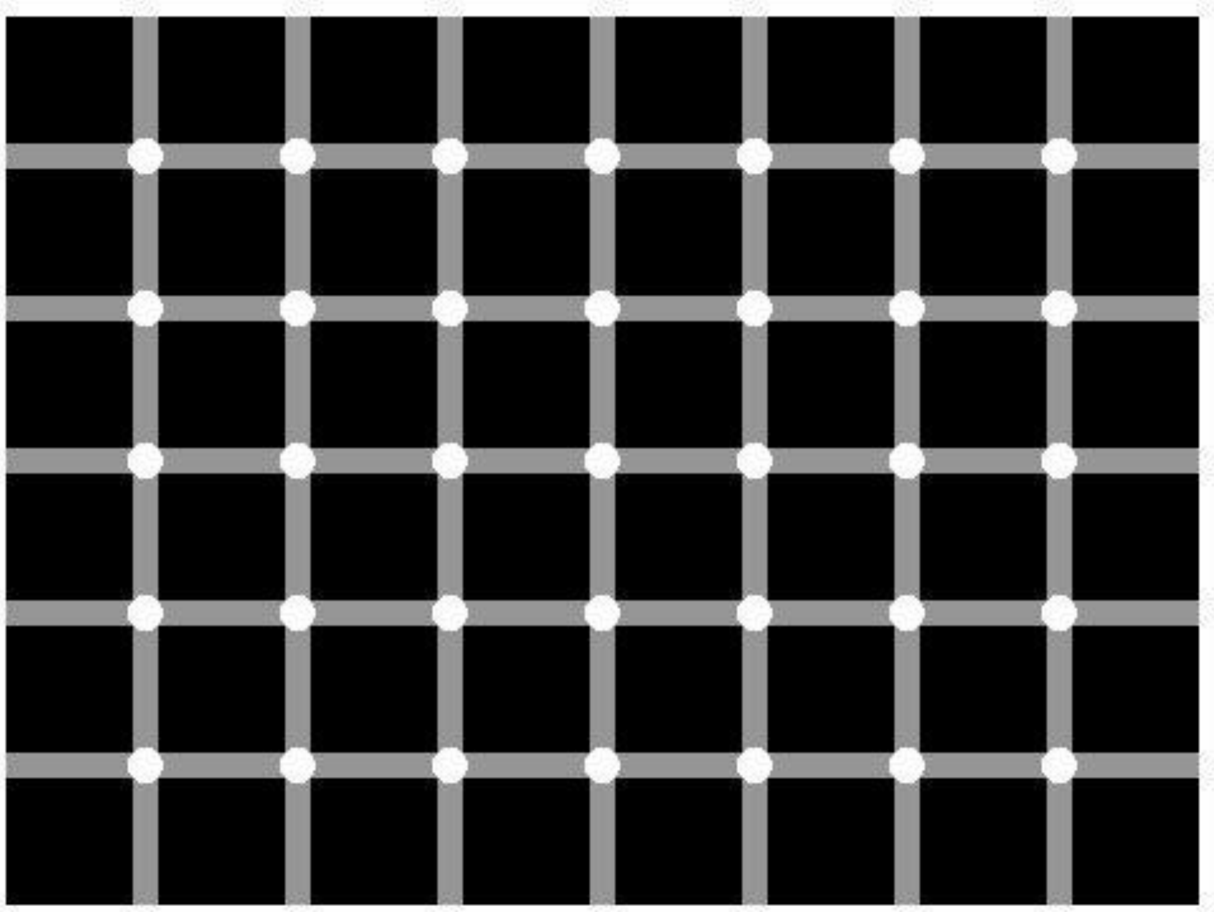


FIGURE 2.2 Distribution of rods and cones in the retina.

➢ HVS: Visual Illusion

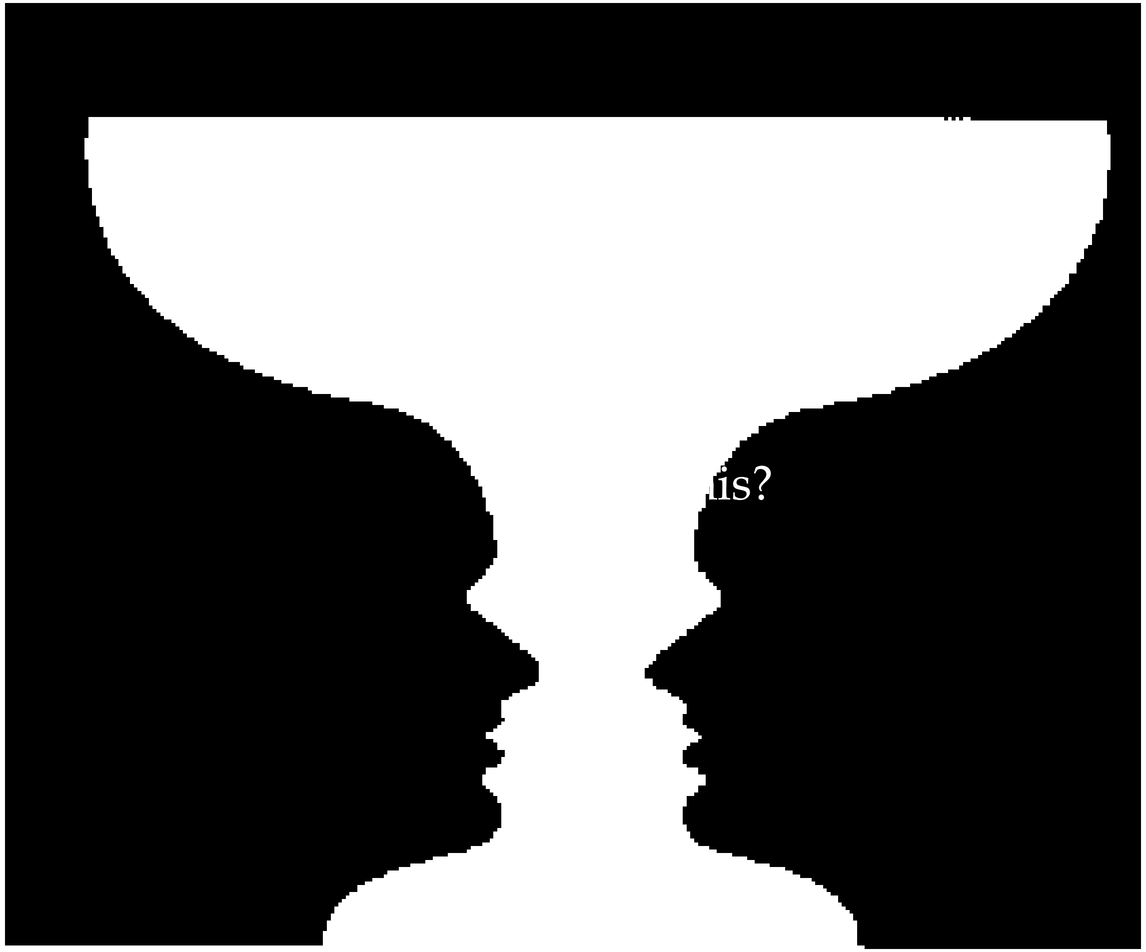


➢ HVS: Visual Illusion



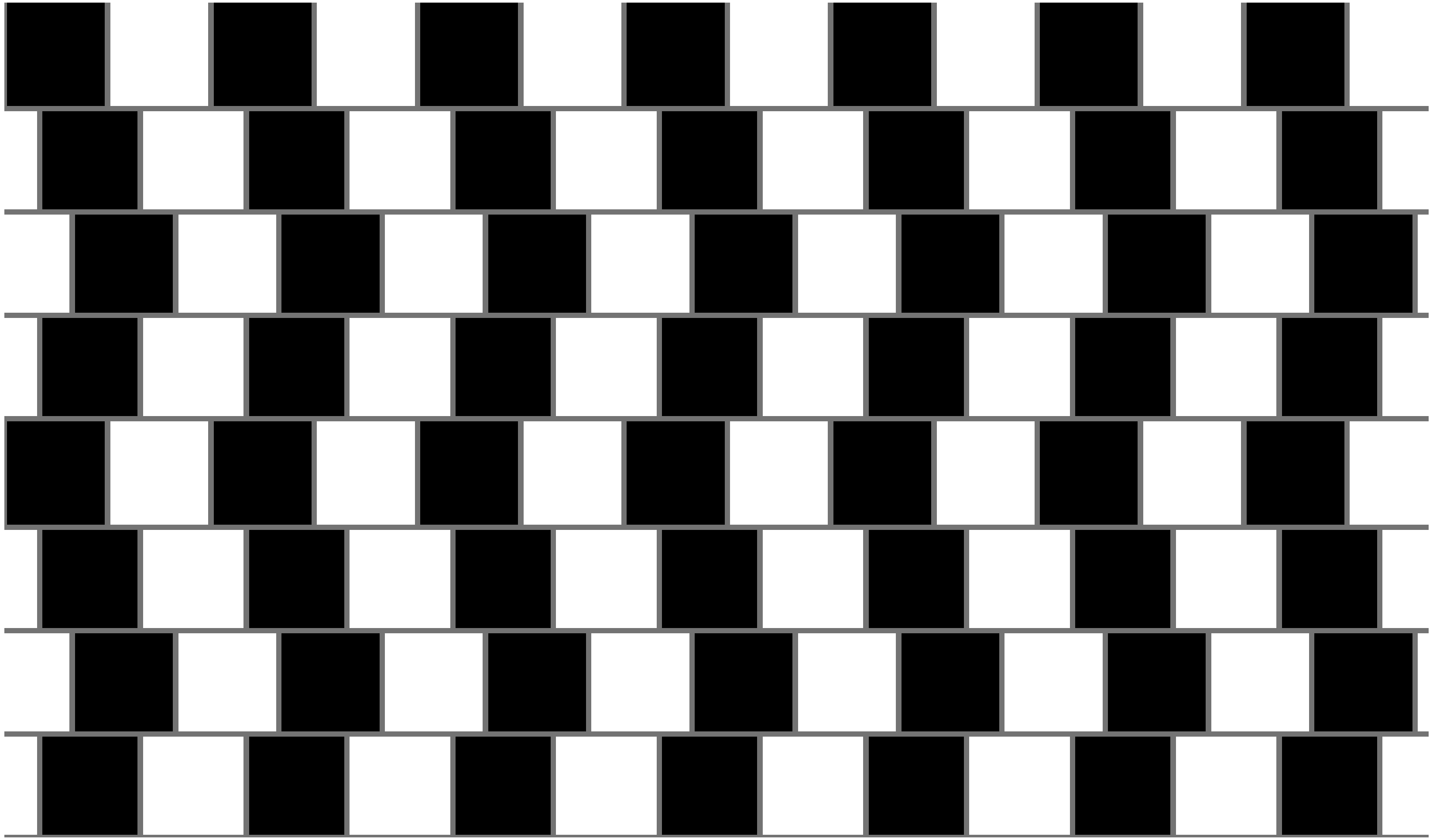
Find the black dot

➢ HVS: Visual Illusion



What is this?

➢ HVS: Visual Illusion



Which lines are straight?

➢ Color

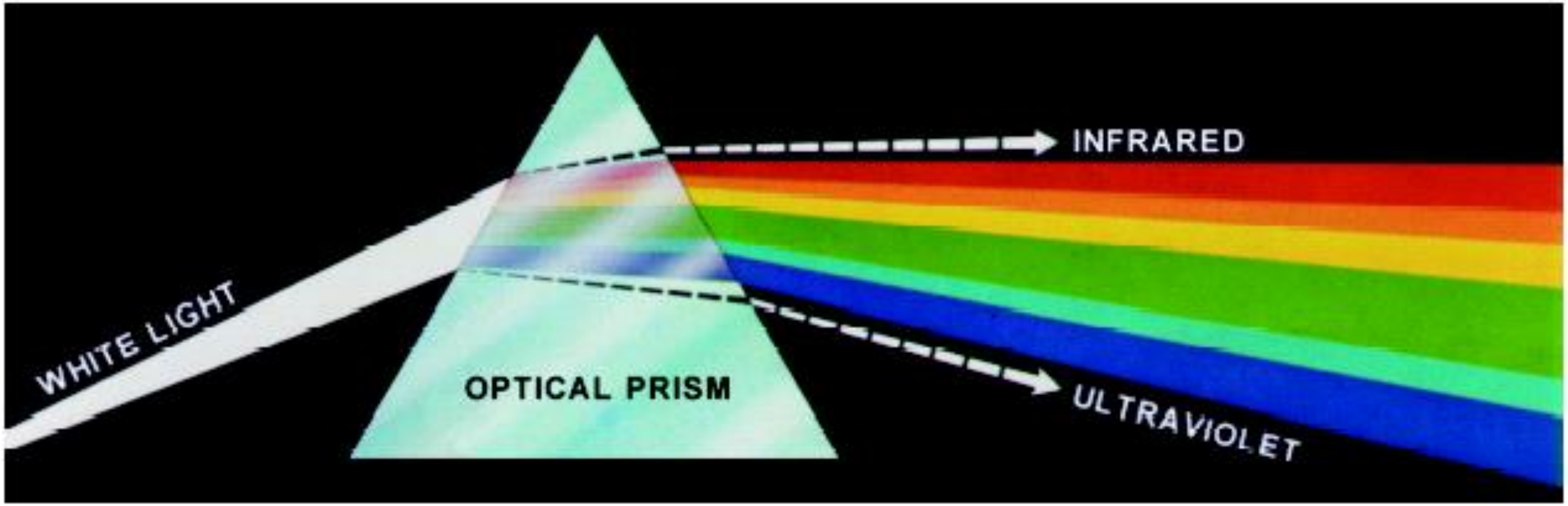
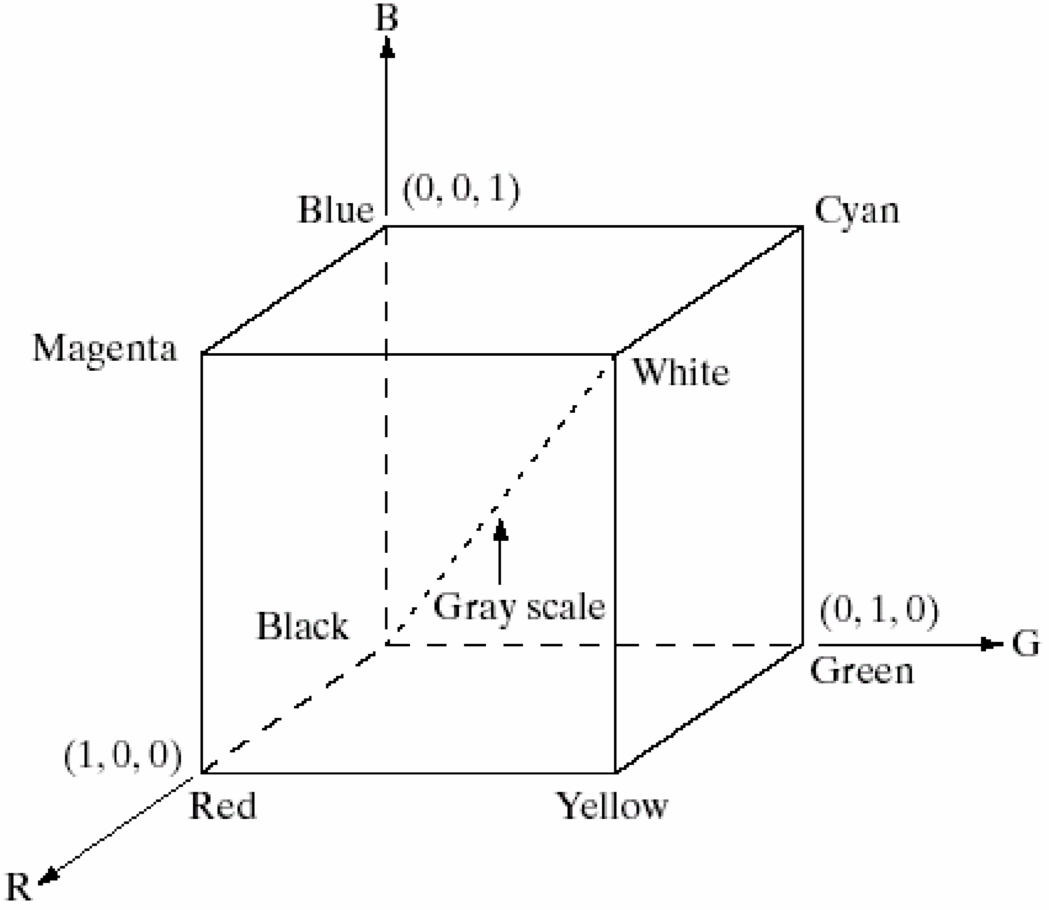
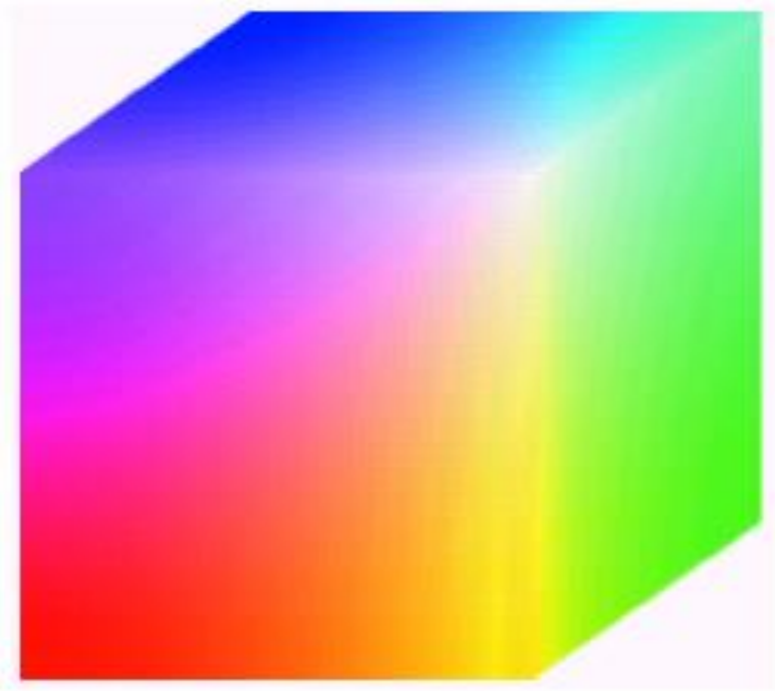


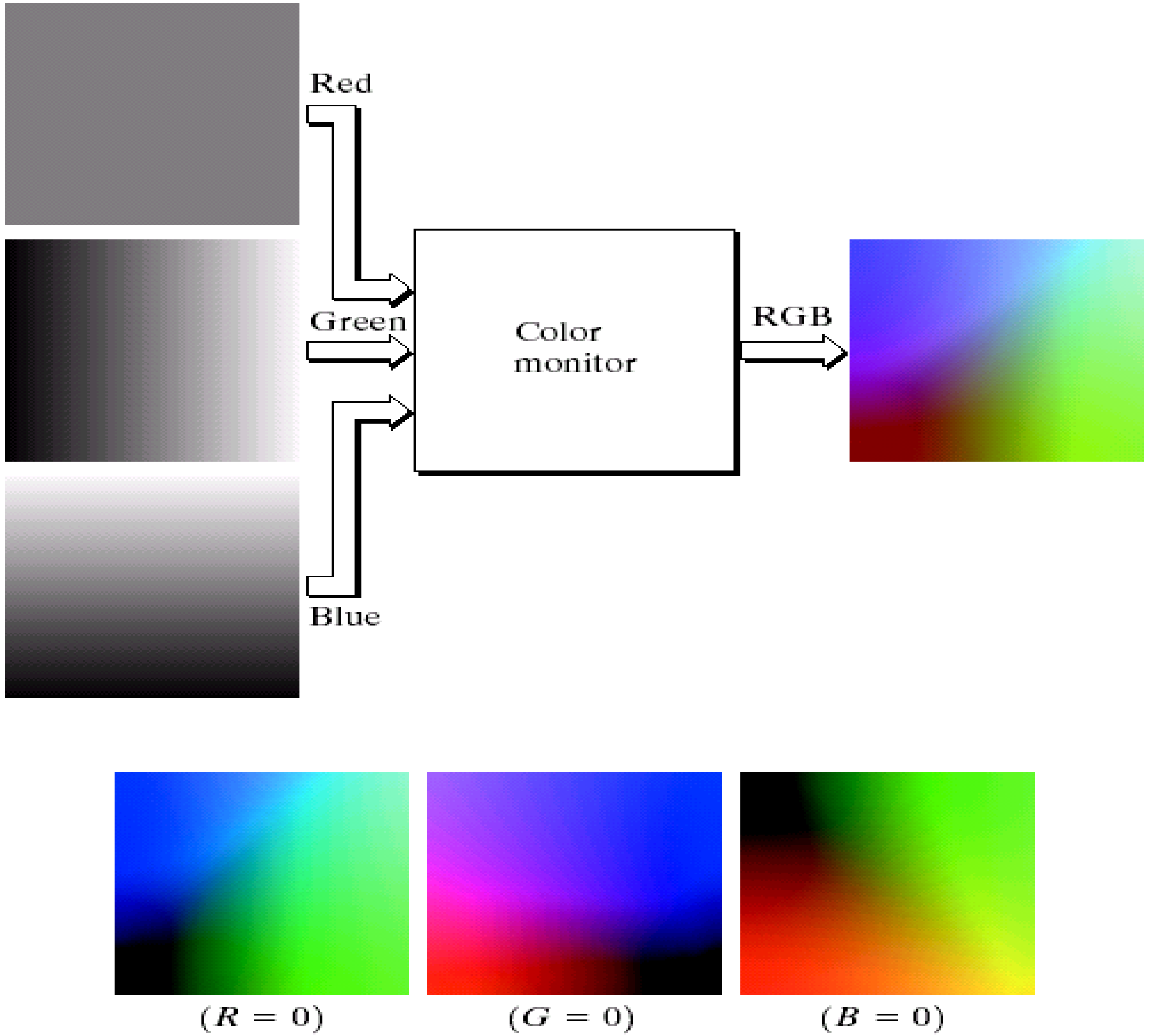
FIGURE 6.1 Color spectrum seen by passing white light through a prism. (Courtesy of the General Electric Co., Lamp Business Division.)

➢ Color: RGB Cube





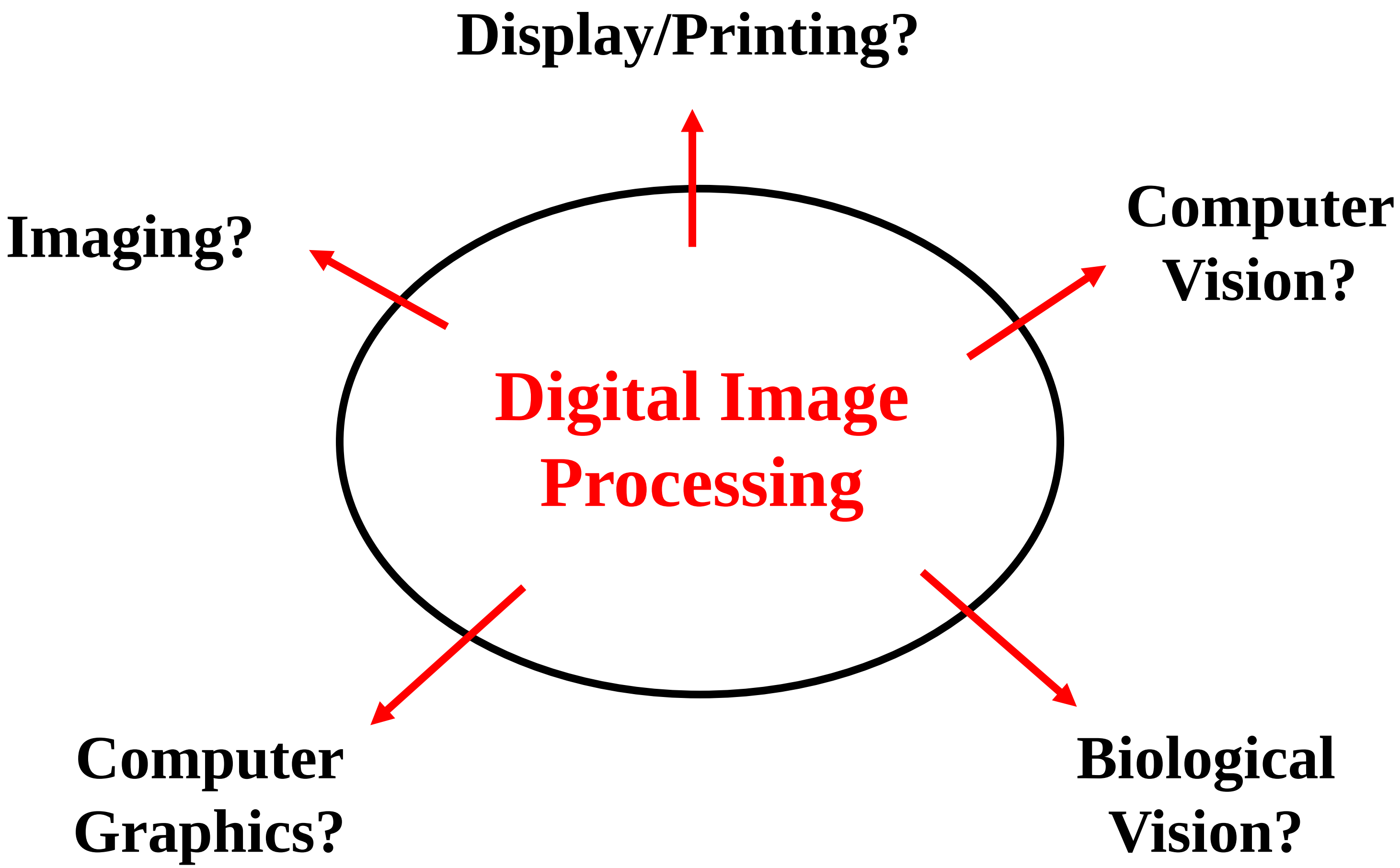
➢ Color: RGB Representation



a
b

FIGURE 6.9
(a) Generating the RGB image of the cross-sectional color plane (127, G, B).
(b) The three hidden surface planes in the color cube of Fig. 6.8.

➢ Where Are We?


Display/Printing?
Imaging?
Computer Vision?
Digital Image Processing
Computer Graphics?
Biological Vision?

➢ What Do We Do?

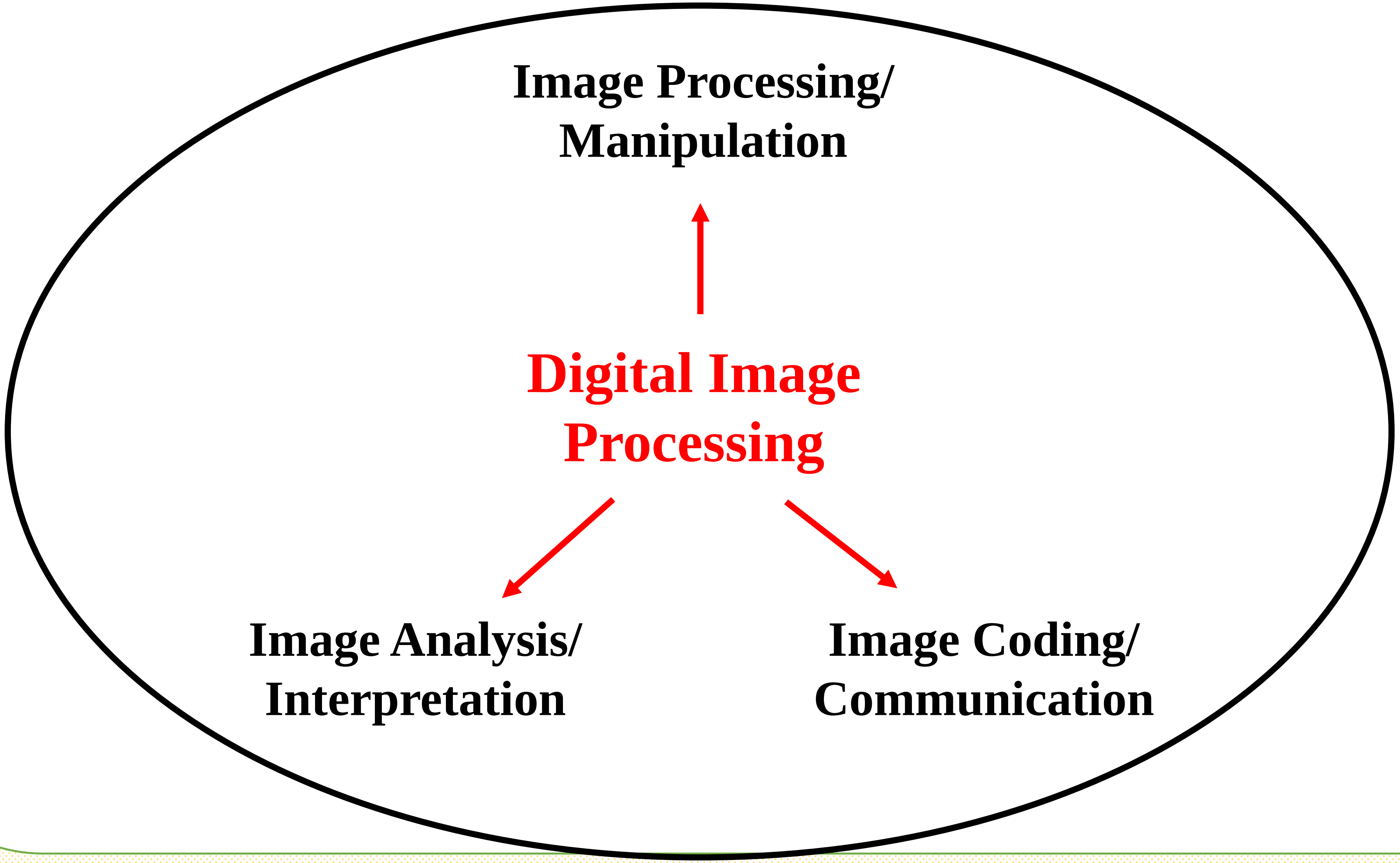
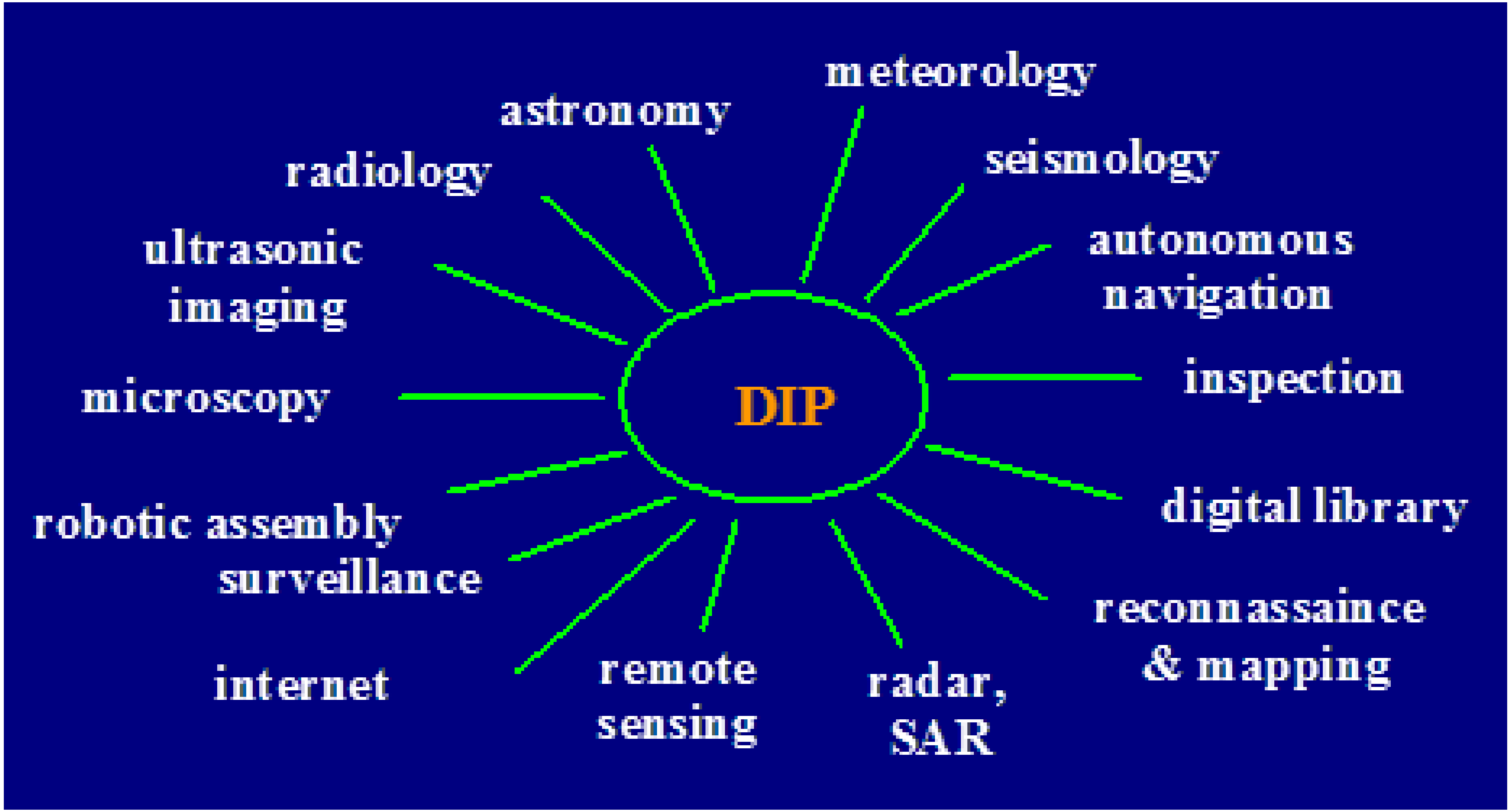


Image Processing/
Manipulation
Digital Image
Processing
Image Analysis/
Interpretation
Image Coding/
Communication

➢ Applications of DIP



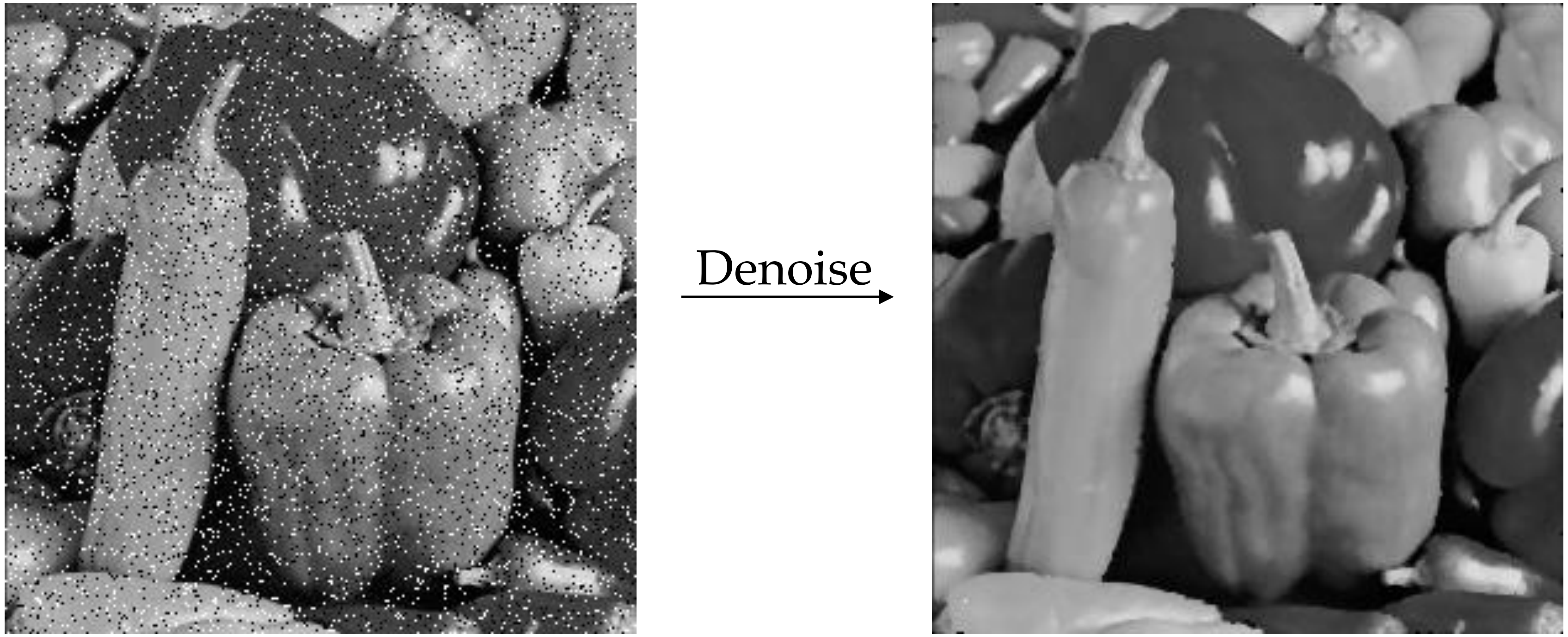
➢ Image Processing: Image Enhancement

resolution

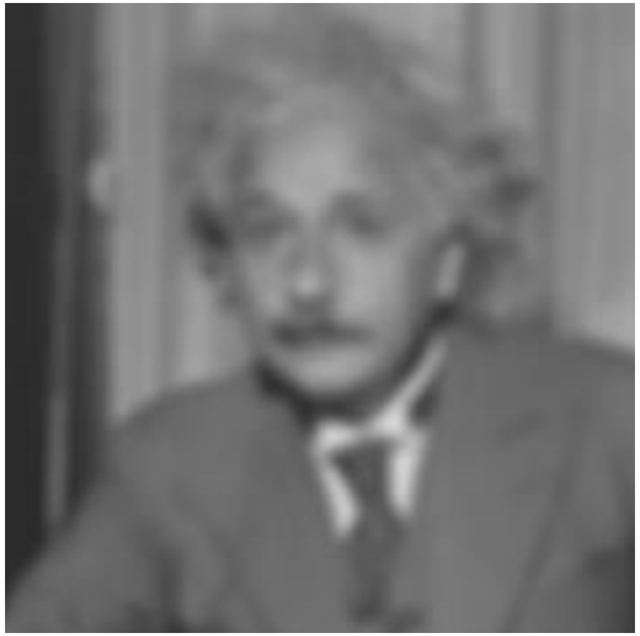
Enhance →

From [Gonzalez & Woods]

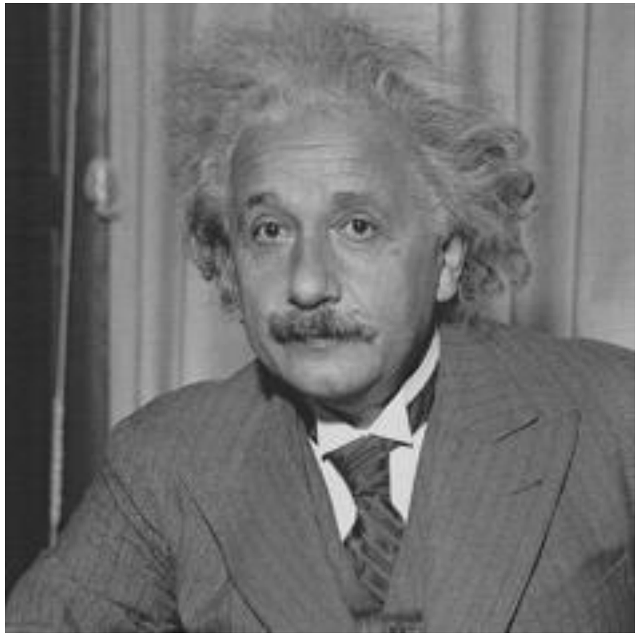
➢ Image Processing: Image Denoising



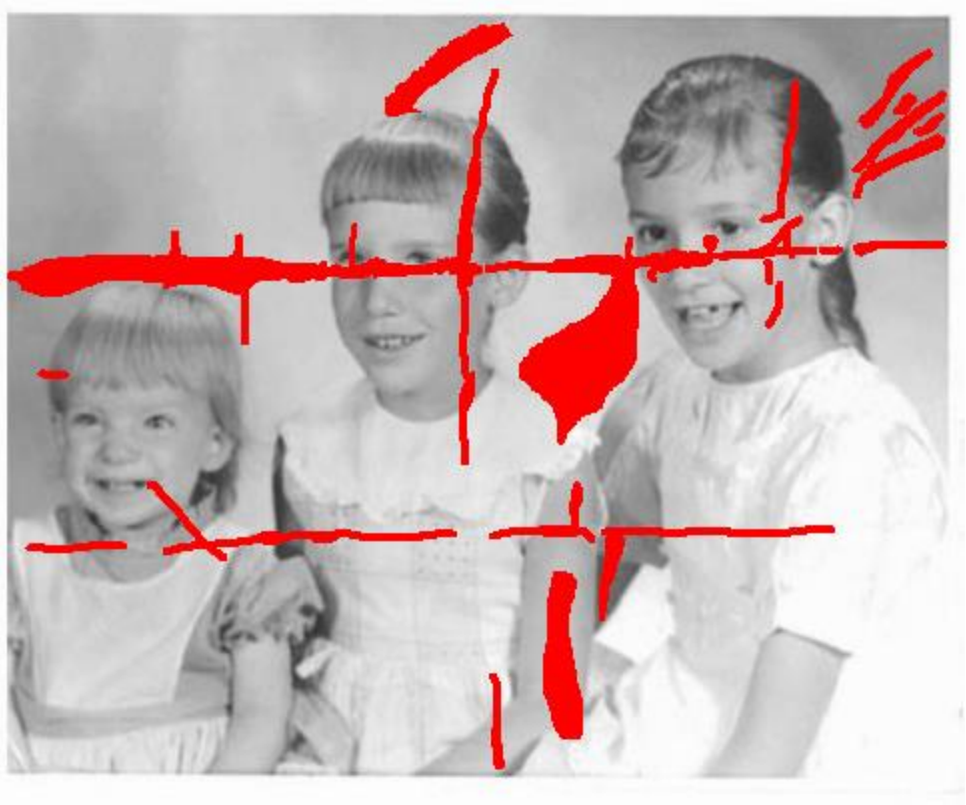
➢ Image Processing: Image Deblurring



Deblur →



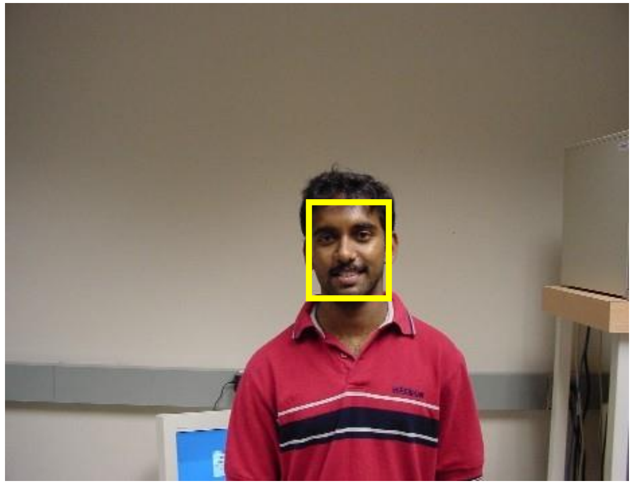
➢ Image Processing: Image Inpainting

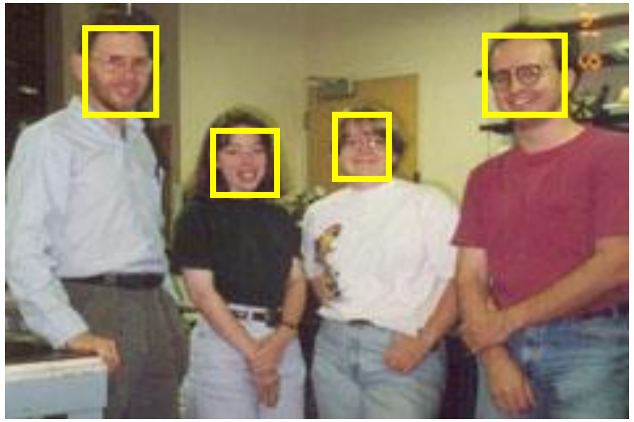


➢ Image Analysis: Edge Detection

From [Gonzalez & Woods]

➢ Image Analysis: Face Detection





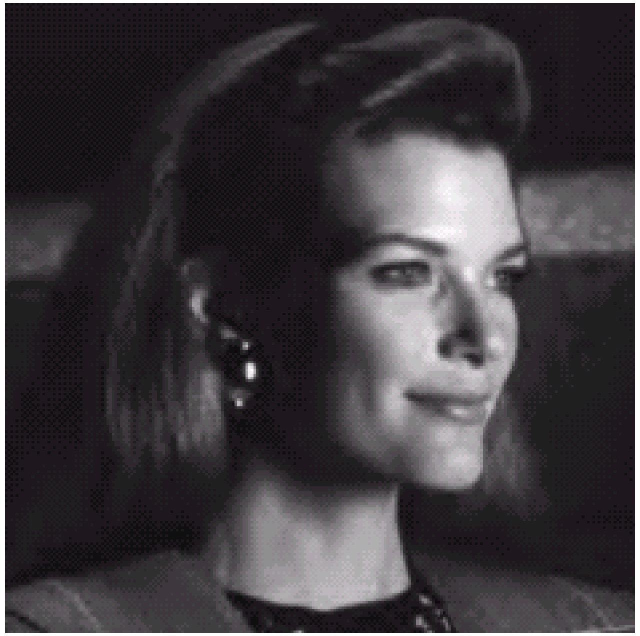
➢ Image Analysis: Image Matching

Two deceivingly similar fingerprints of two different people

➢ Image Coding: Image Compression

original image
262144 Bytes

From [Gonzalez & Woods]



From [Gonzalez & Woods]

image encoder → compressed bitstream 00111000001001101… (2428 Bytes) → **image decoder**

compression ratio (CR) = 108:1


Thank you