Chapter 2

Digital Image Fundamentals

◇ Contents

2.1 Elements of Visual Perception

- Structure of the human eye
- Image formation in the eye

2.1.1 Structure of the Human Eye(1)

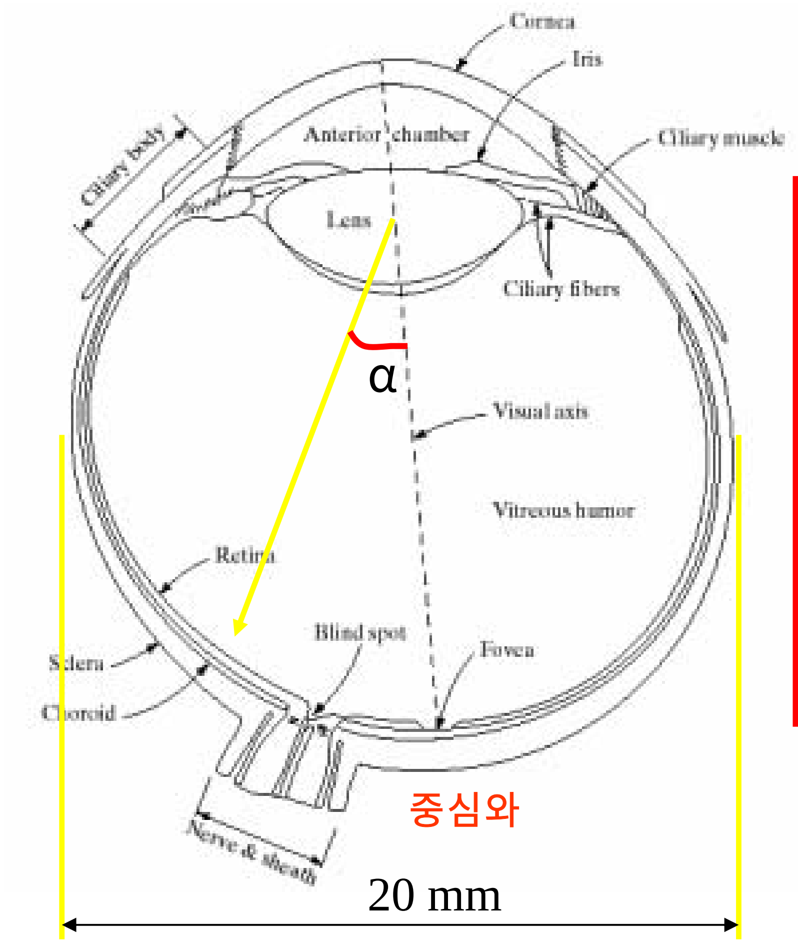


FIGURE 2.1 Simplified diagram of a cross section of the human eye.

3 membranes cover eyes.
- Cornea and Sclera(각막)
- Choroid : (맥락막)
  Ciliary - network of blood vessels
  Iris – diaphragm with Pupil
- Retina : cones and rods (망막)

◇ Structure of the Human Eye(2)

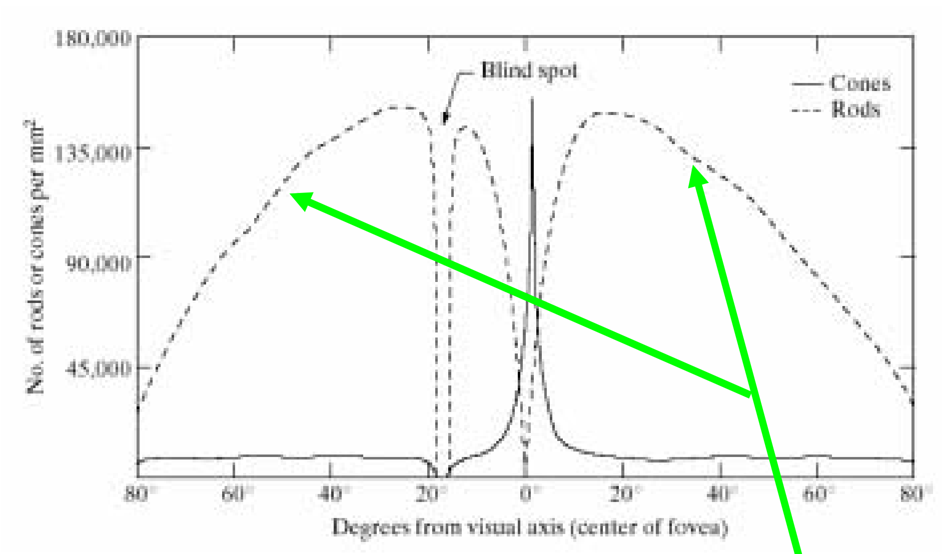


FIGURE 2.2 Distribution of rods and cones in the retina.

Distribution of discrete light receptors

Receptors	Quantity/eye	Sensitive to	Population
Cones	6 to 7 M	color	cluttered around near Fovea
Rods (시신경의 간상체)	75 to 150 M	brightness	scattered radially symmetrically and indirectly proportional to radius

2.1.2 Image Formation in the Eye

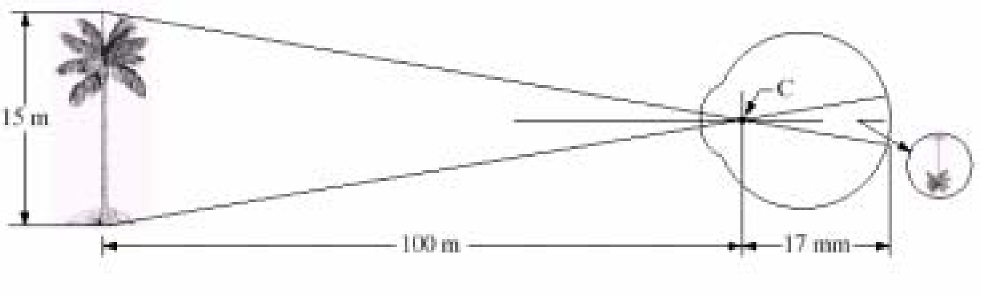


FIGURE 2.3 Graphical representation of the eye looking at a palm tree. Point C is the optical center of the lens.

- Lens of human eye is flexible.
- Ciliary muscle controls thickness of lens from 14 mm to 17 mm. → focal length
- Lens is thickened to focus on nearby objects and flattened on distant objects.
- As lens is thickened, focal length decreases.

2.2 Light and the Electromagnetic Spectrum

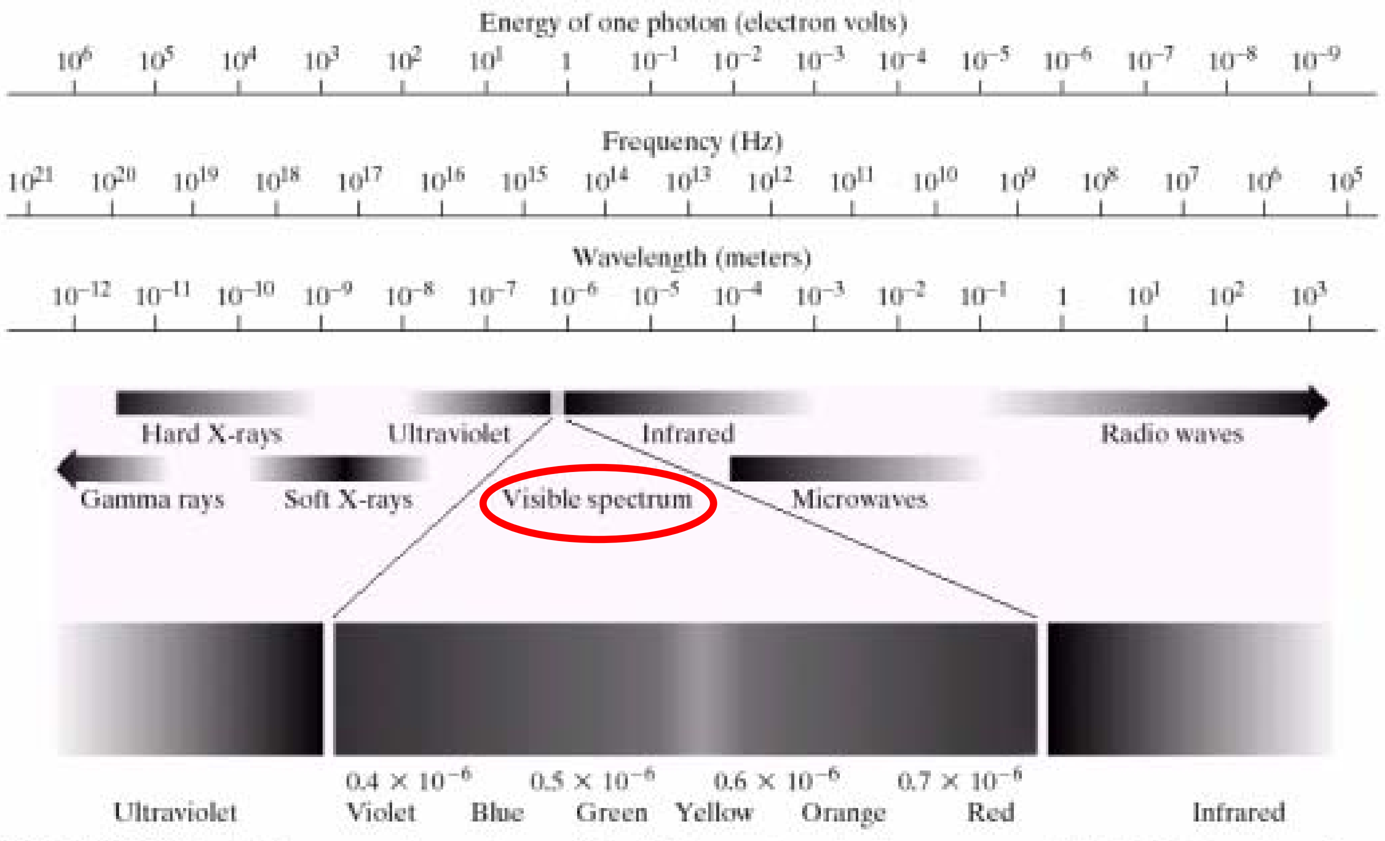


FIGURE 2.10 The electromagnetic spectrum. The visible spectrum is shown zoomed to facilitate explanation, but note that the visible spectrum is a rather narrow portion of the EM spectrum.

2.3 Image Sensing and Acquisition – Sensor Types

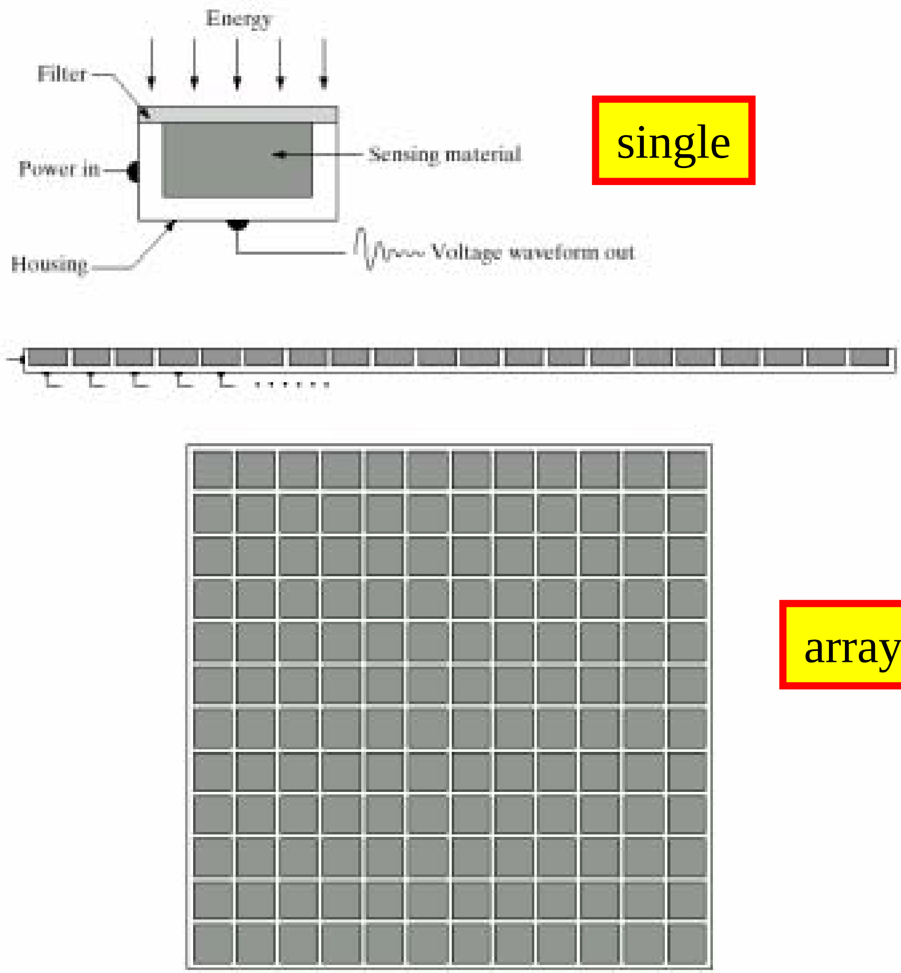


FIGURE 2.12 (a) Single imaging sensor. (b) Line sensor. (c) Array sensor.

single

line/strip

array

2.3.1 2D Image with a Single Sensor

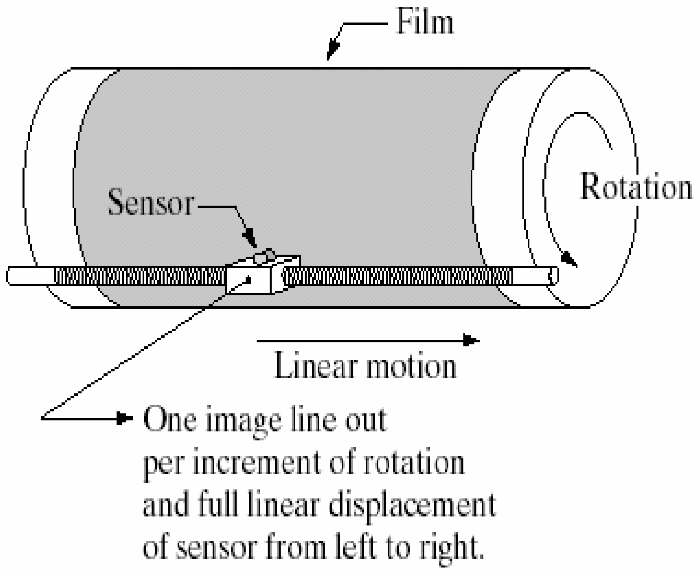
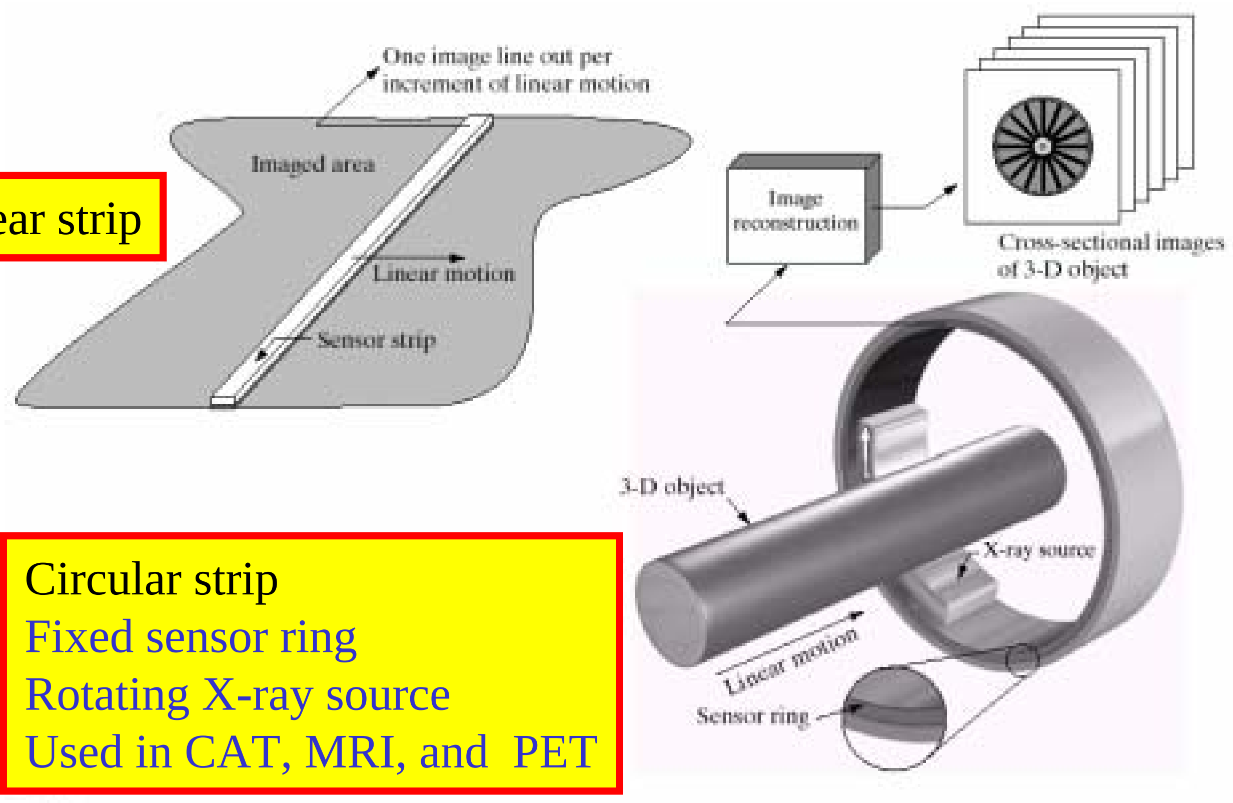

Film
Sensor
Rotation
Linear motion
One image line out
per increment of rotation
and full linear displacement
of sensor from left to right.

FIGURE 2.13 Combining a single sensor with motion to generate a 2-D image.

2.3.2 2D Image with Sensor Strips

Linear strip



Circular strip
Fixed sensor ring
Rotating X-ray source
Used in CAT, MRI, and PET

FIGURE 2.14 (a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

2.3.3 2D Image with Sensor Arrays

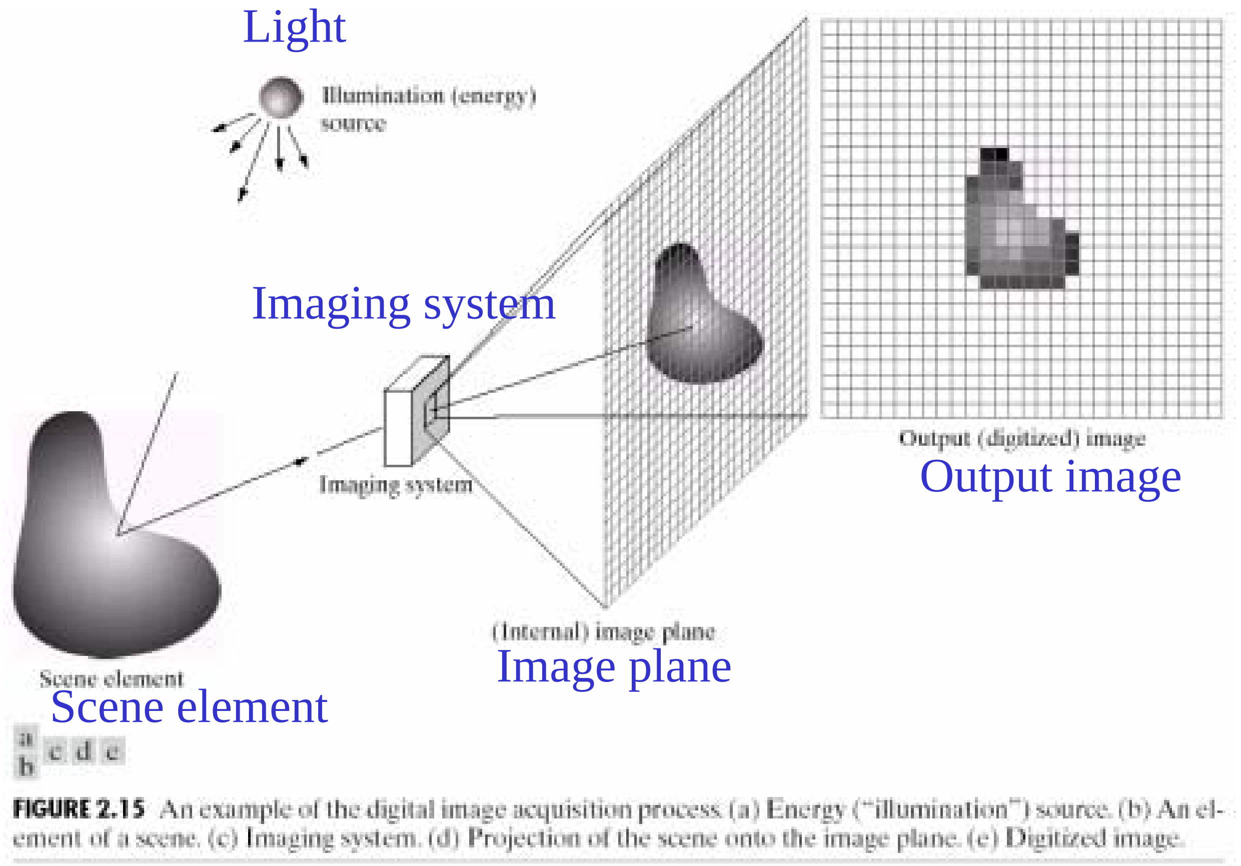


FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

2.3.4 A Simple Image Model (1)

- As light is a form of energy,

$$0 < f(x, y) < \infty$$

- The basic nature of *f(x, y)* may be characterized by two components :
  - **illumination** ***i(x, y)***
    - the amount of source light incident on the scene
  - **reflectance** ***r(x, y)***
    - the amount of light reflected by the objects in the scene

◇ A Simple Image Model (2)

- Amplitude of *f* at *(x, y)*

  = intensity of the image at that point

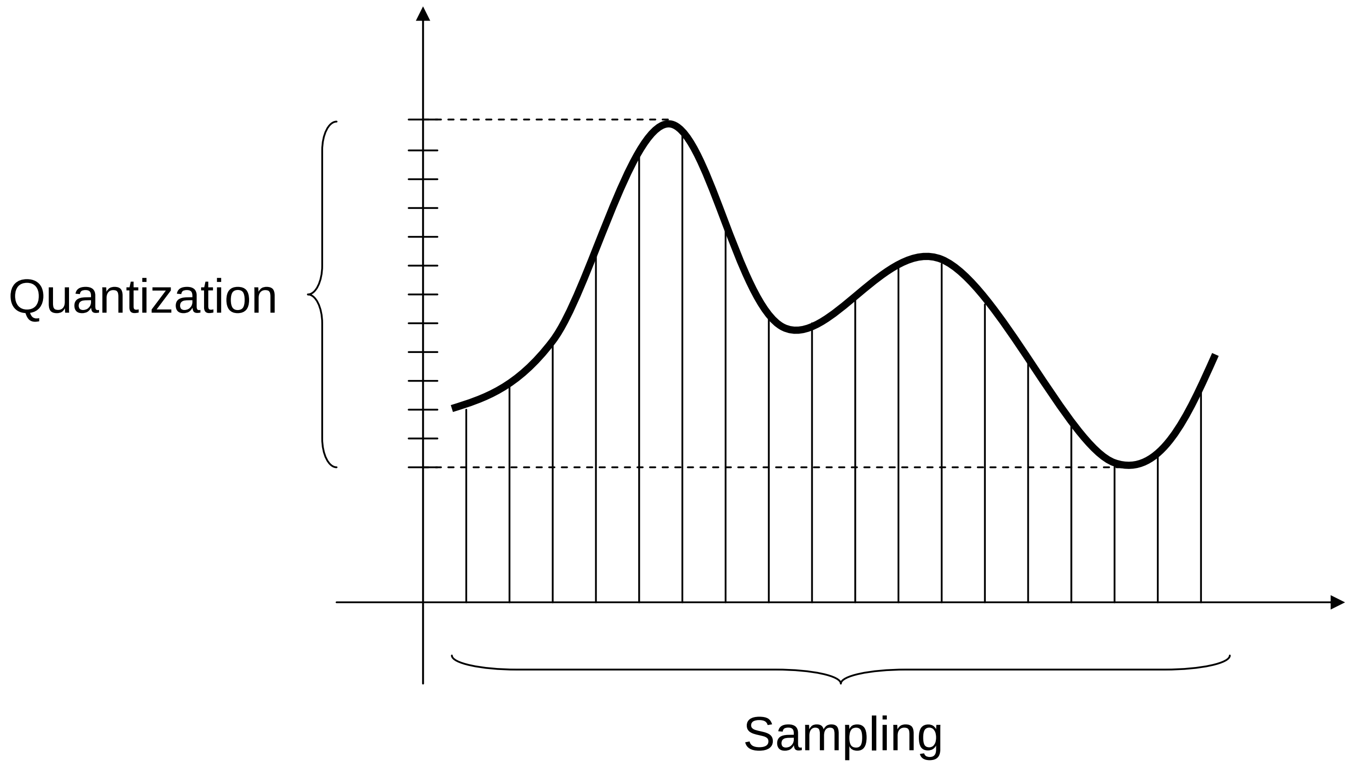
$$f(x, y) = i(x, y)r(x, y)$$

$$0 < i(x, y) < \infty$$
$$0 < r(x, y) < 1$$

◇ A Simple Image Model (3)

$i(x, y)$		$r(x, y)$	
Sunny	*90,000 lm/m^2*	*Black velvet*	*0.01*
Cloudy	*<10,000*	*Stainless steel*	*0.65*
Full moon	*0.1*	*White wall*	*0.80*
Office	*1000*	*Silver*	*0.90*
		Snow	*0.93*

2.4 Image Sampling and Quantization



2.4.1 Basic Concepts(1)

Continuous image

Continuous scan line

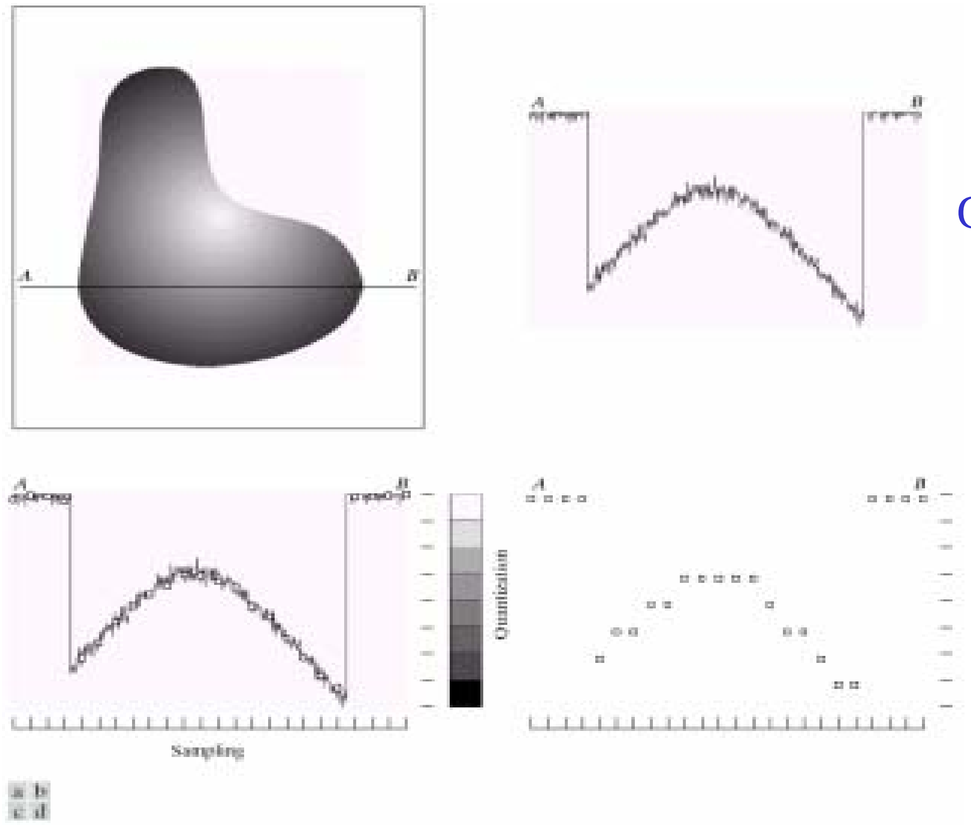


FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Sampling & quantization

Digital Scan line

◇ Basic Concepts(2)

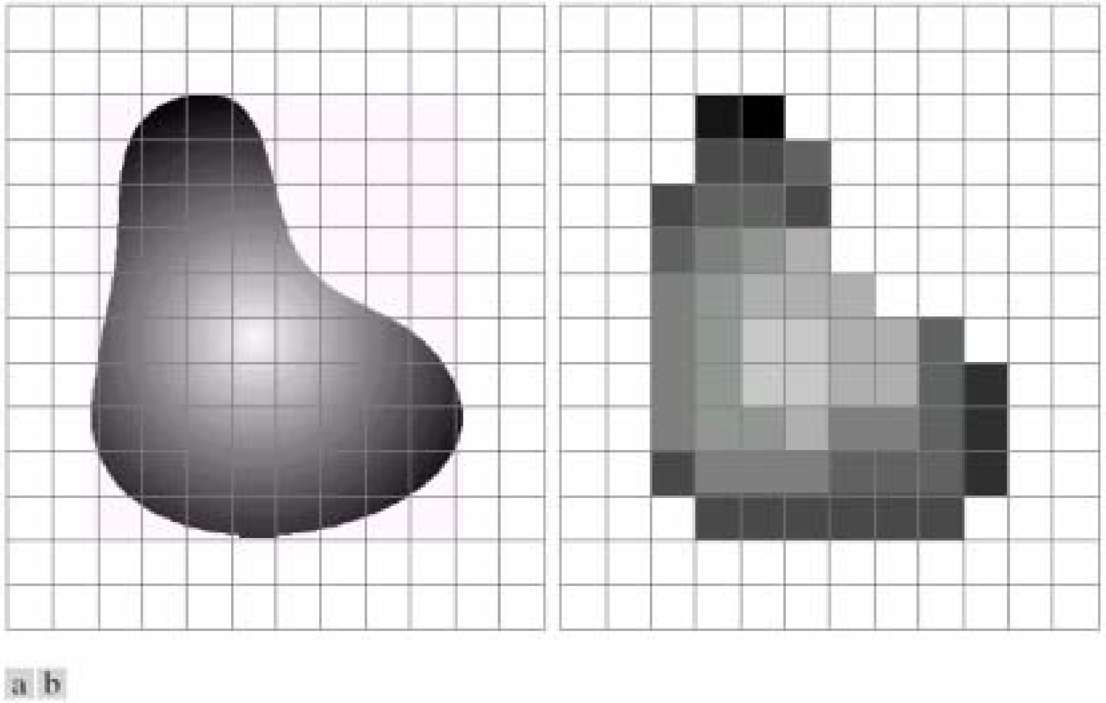


FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

◇ Basic Concepts(3)

- The **resolution** of an image depends on the # of samples and # of gray levels.
- **Sampling** determines **Spatial Resolution.**
- **Quantization** determines **Gray-level Resolution.**
- **Digitization of the spatial coordinates (x,y)** is called **image sampling**.
- **Digitization of amplitude** is called **gray-level quantization.**

2.4.2 Representing Digital Images(1)

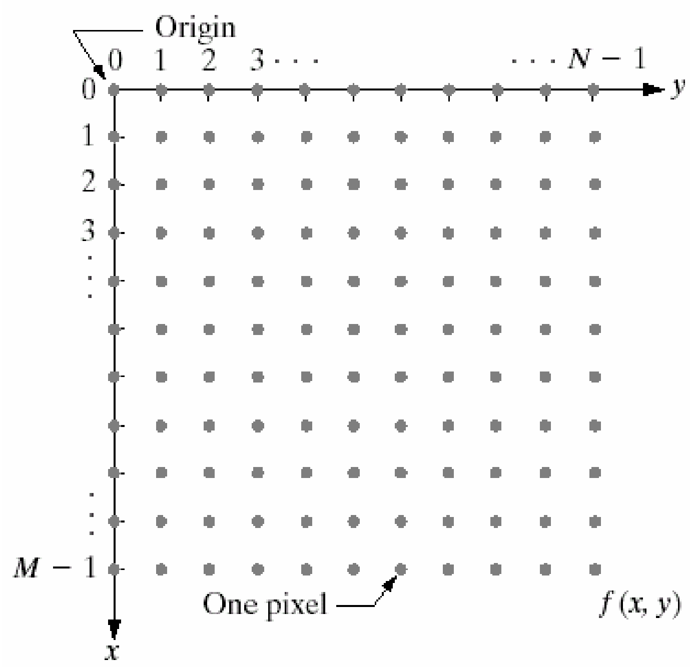


FIGURE 2.18 Coordinate convention used in this book to represent digital images.

◇ Representing Digital Images(2)

$$f(x,y) \approx \underbrace{\begin{bmatrix} f(0,0) & \Lambda & \\ M & O & M \\ & \Lambda & f(M-1,N-1) \end{bmatrix}}_{N} \dashrightarrow M$$

M = **y** size of image,   N = **x** size of image

G = 2^k = number of gray levels

The # of bits required to store a digitized image

= M × N × k

◇ Representing Digital Images(3)

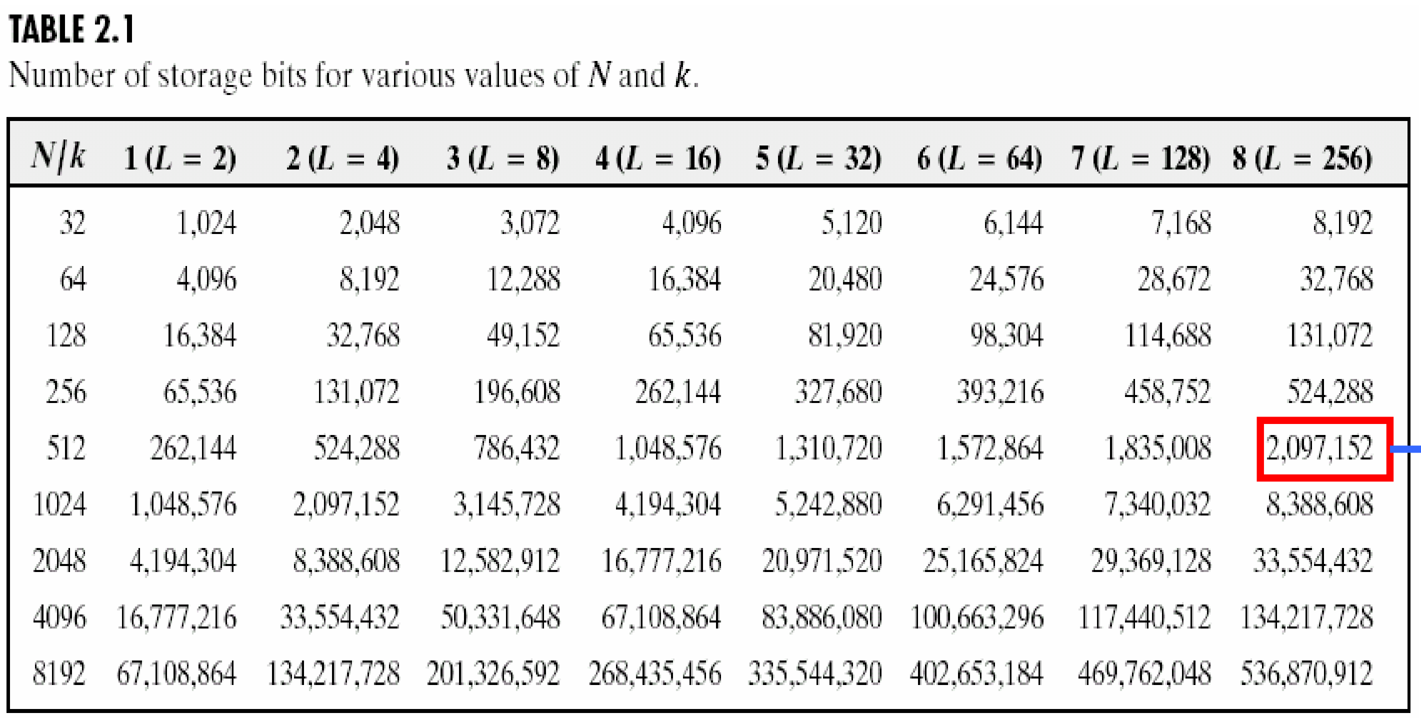


TABLE 2.1
Number of storage bits for various values of *N* and *k*.

N/k	1 (L = 2)	2 (L = 4)	3 (L = 8)	4 (L = 16)	5 (L = 32)	6 (L = 64)	7 (L = 128)	8 (L = 256)
32	1,024	2,048	3,072	4,096	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,384	32,768	49,152	65,536	81,920	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	393,216	458,752	524,288
512	262,144	524,288	786,432	1,048,576	1,310,720	1,572,864	1,835,008	2,097,152
1024	1,048,576	2,097,152	3,145,728	4,194,304	5,242,880	6,291,456	7,340,032	8,388,608
2048	4,194,304	8,388,608	12,582,912	16,777,216	20,971,520	25,165,824	29,369,128	33,554,432
4096	16,777,216	33,554,432	50,331,648	67,108,864	83,886,080	100,663,296	117,440,512	134,217,728
8192	67,108,864	134,217,728	201,326,592	268,435,456	335,544,320	402,653,184	469,762,048	536,870,912

Assume that M = N.

262,144 bytes (256KB)

◇ Representing Digital Images(4)

- How many samples and gray levels are required for a good approximation? **-> Trade-off between Quality and Computational complexity**

2.4.3 Effect of Varying Spatial Resolution(1)

- Decreasing the **spatial resolution**(Figure 2.19)
- Figure 2.19(a) : N=1024
- Figure 2.19(b) ~ (f) : N=512, 256, 128, 64, 32
- For all cases, # of gray levels = 256
- Display area shrinks according to subsampling.

◇ Effect of Varying Spatial Resolution(2)

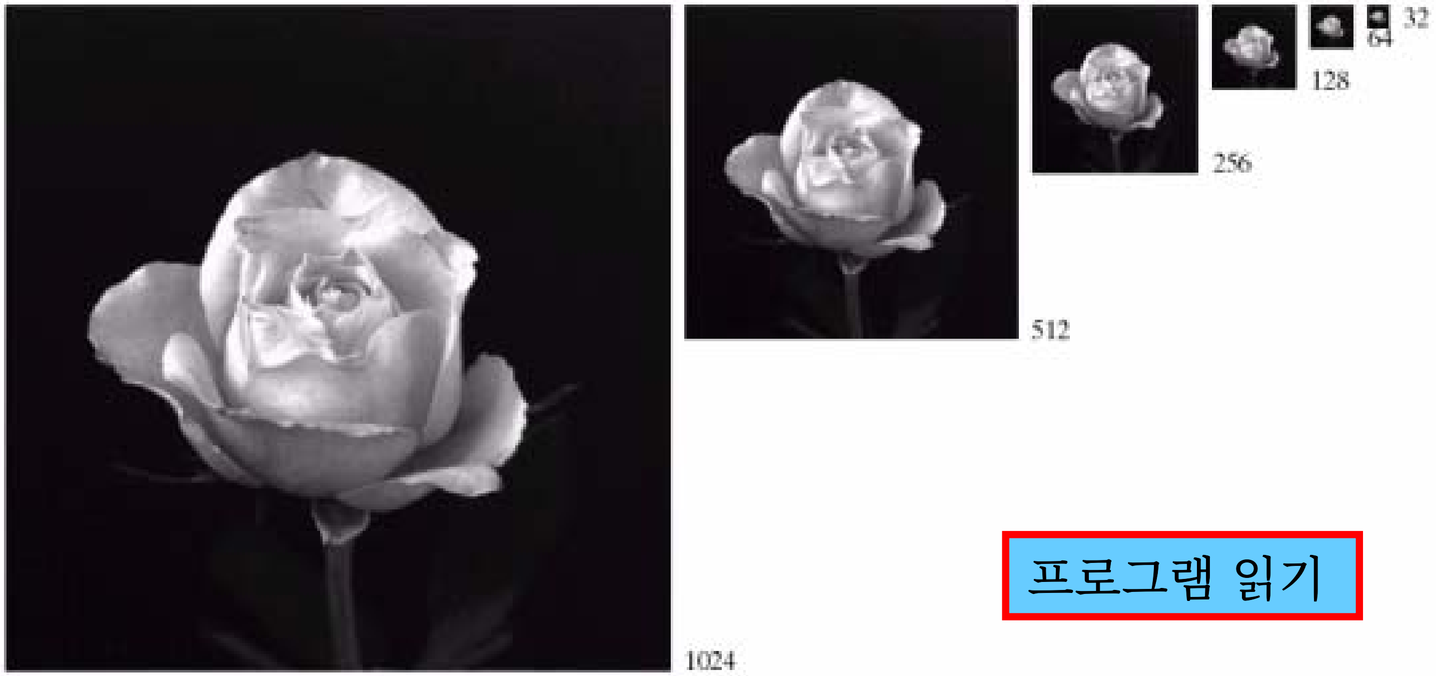


FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

프로그램 읽기

◇ Effect of Varying Spatial Resolution(3)

- Decreasing the **spatial resolution**(Figure 2.20)
- Figure 2.20(a) : N=1024
- Figure 2.20(b) ~ (f) : N=512, 256, 128, 64, 32
- For all cases, # of gray levels = 256
- Display area used for each image is the same. (1024*1024 display field)

◇ Effect of Varying Spatial Resolution(4)

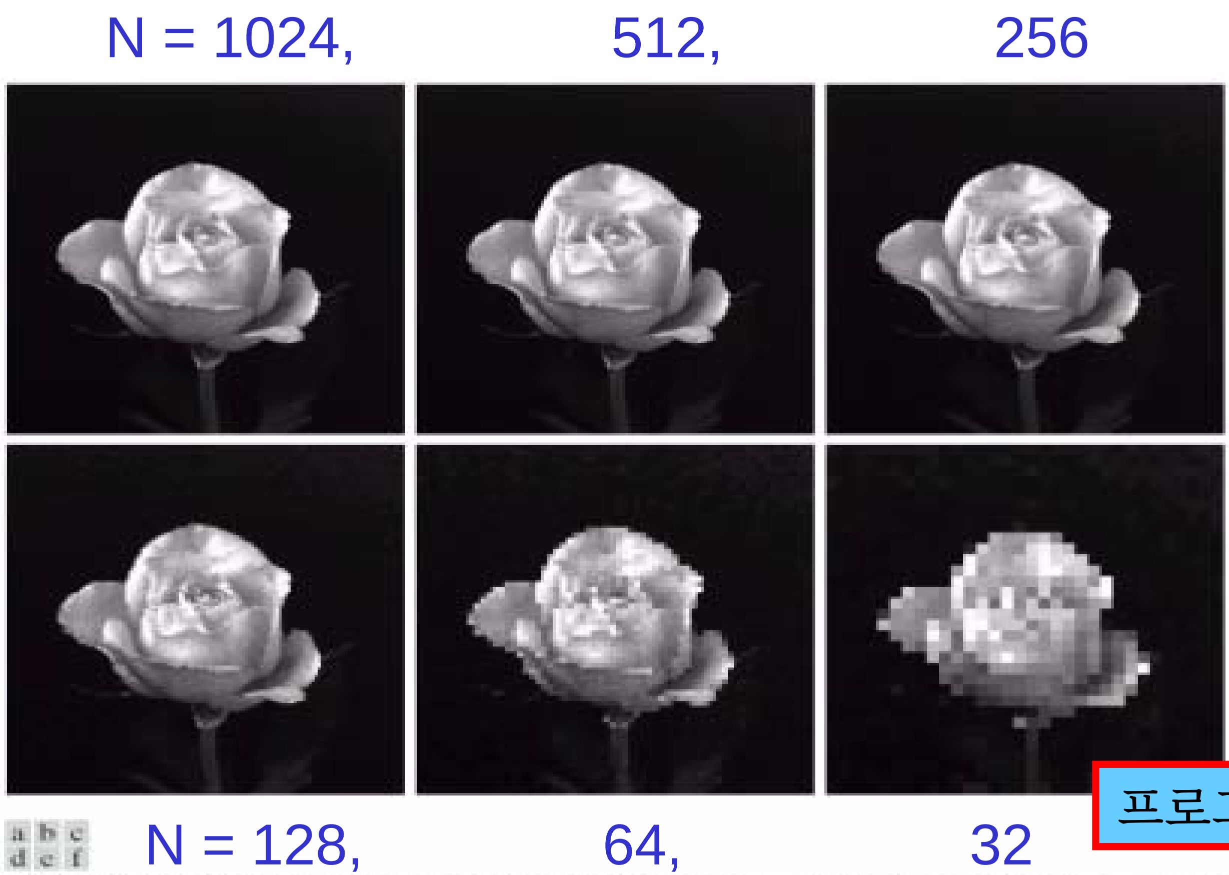


FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

프로그램 읽기

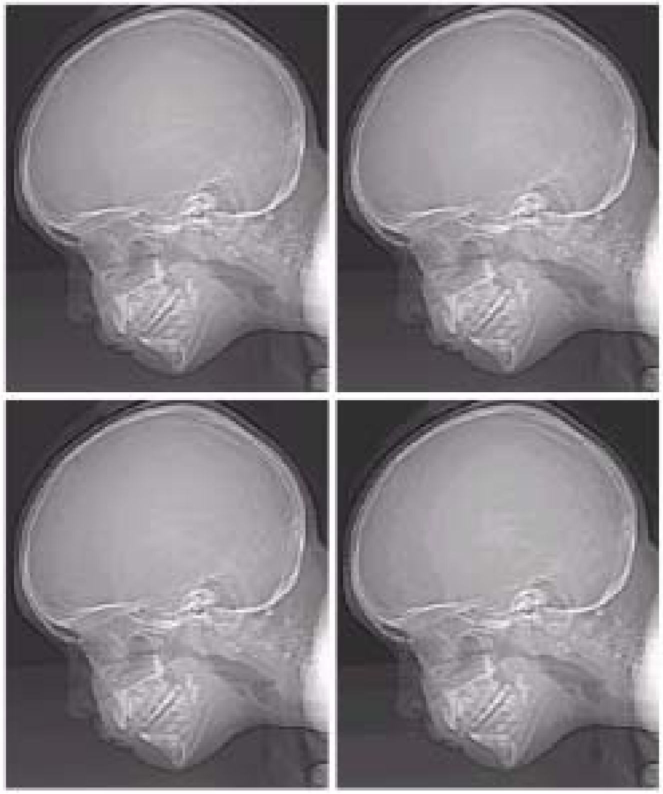
◇ Effect of Varying Gray Level Resolution(1)

- Decreasing the **gray level resolution**(Figure 2.21)
- Figure 2.21(a) : N=1024, m = 8bit image
- Figure 2.21(b) ~ (h) : m = 7 to m = 1

프로그램 실습(30분)

◇ Effect of Varying Gray Level Resolution(2)

m = 8,   m = 7



m = 6   m = 5

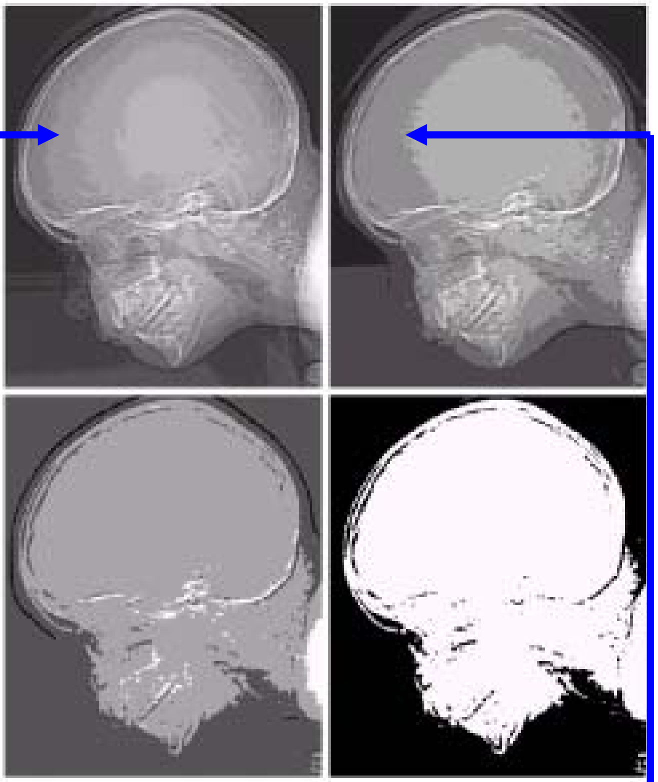
a b
c d

FIGURE 2.21 (a) 452 × 374, 256-level image. (b)–(d) Image displayed in 128, 64, and 32 gray levels, while keeping the spatial resolution constant.

e f
g h

FIGURE 2.21 (Continued) (e)–(h) Image displayed in 16, 8, 4, and 2 gray levels. (Original courtesy of Dr. David R. Pickens, Department of Radiology & Radiological Sciences, Vanderbilt University Medical Center.)

m = 4   m = 3



m = 2   m = 1

False contouring effects

◇ Effect of Varying Gray Level Resolution(3)

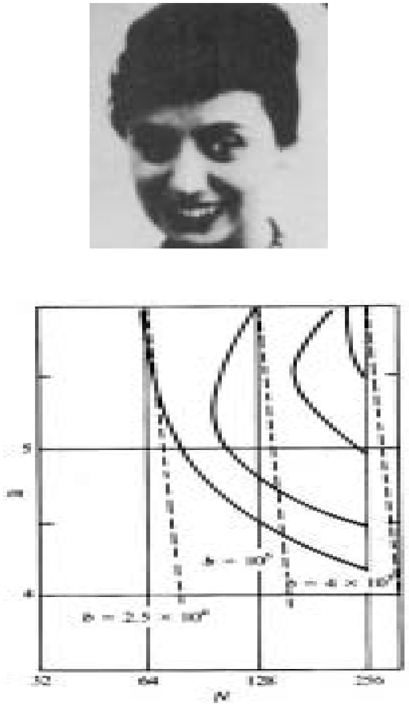
- **False contouring effect** in low gray level resolution

REMEDY

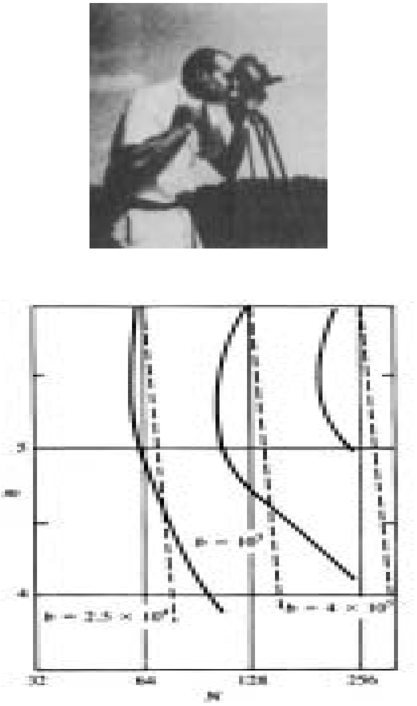
- **Nonuniform sampling** -> sparse sampling in smoothly varying area, dense sampling in abruptly varying area
- **Nonuniform quantization** -> sparse quantization in abruptly varying area, dense quantization in smoothly varying area

◇ Effect of Varying Gray Level Resolution(4)

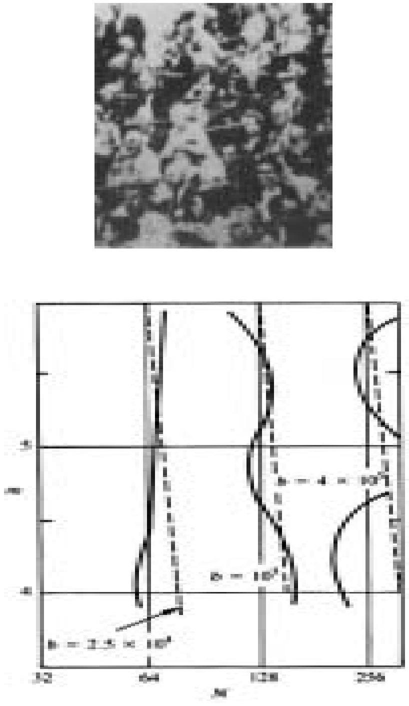
- Isopreference Curves.



(a) : little detail.



(b) : intermediate amount of detail.



(c) : a large amount of detail.

2.4.4 Aliasing and Moire Patterns

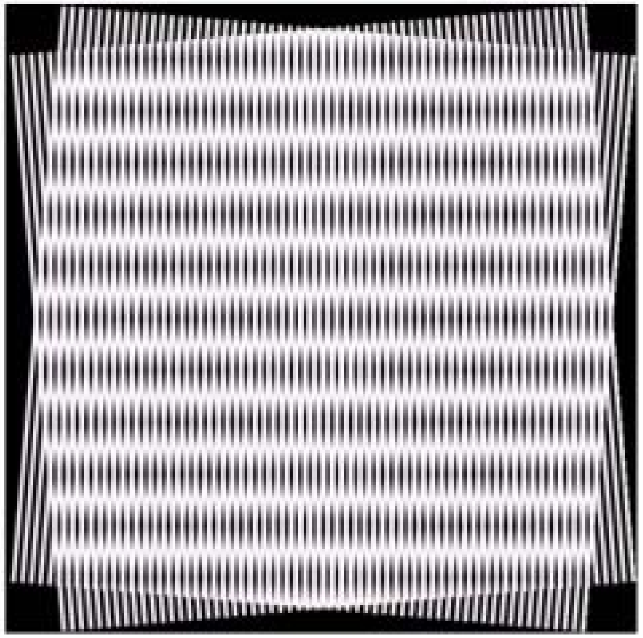


FIGURE 2.24 Illustration of the Moiré pattern effect.

2D sinusoidal(aliased) waveform, Moire pattern, is generated due to the break-up of periodicity.

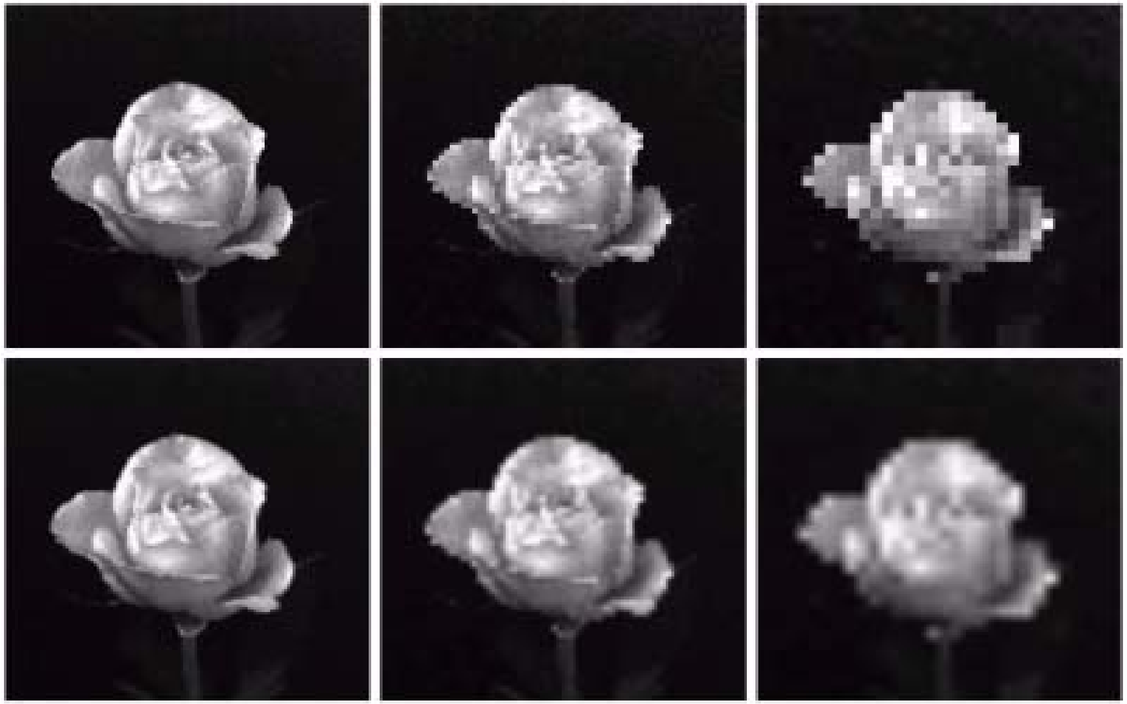
2.4.5 Zooming and Shrinking(1)

Zooming requires two steps :

1. Creation of new pixel location
   Geometric transformation

2. Assignment of new gray level to that location
   Gray level interpolation

◇ Zooming and Shrinking(2)



a b c
d e f

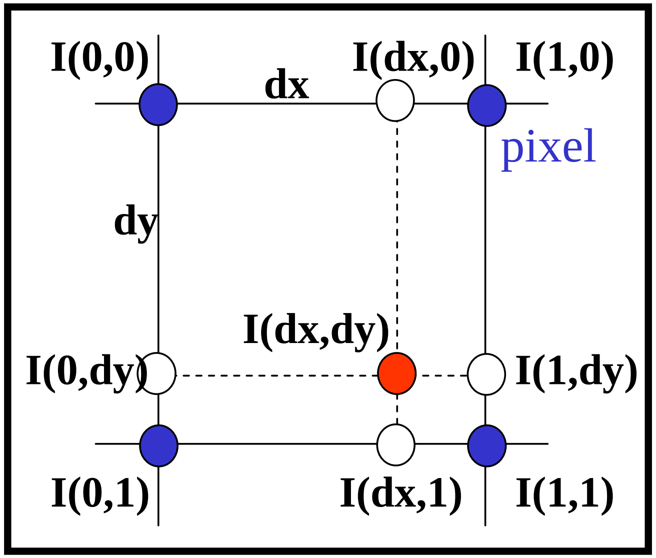
FIGURE 2.25 Top row: images zoomed from 128 × 128, 64 × 64, and 32 × 32 pixels to 1024 × 1024 pixels, using nearest neighbor gray-level interpolation. Bottom row: same sequence, but using bilinear interpolation.

Zooming with nearest neighbor interpolation

Zooming with bilinear interpolation

프로그램 읽기

Bilinear Interpolation (p65)



- $I(dx,0) = (I(1,0) - I(0,0)) \times dx + I(0,0)$

  $= (1-dx) \times I(0,0) + (dx) \times I(1,0)$
- $I(dx,1) = (1-dx) \times I(0,1) + (dx) \times I(1,1)$
- $I(0,dy) = (1-dy) \times I(0,0) + (dy) \times I(0,1)$
- $I(1,dy) = (1-dy) \times I(1,0) + (dy) \times I(1,1)$

I(dx,0), I(dx,1), I(0,dy), I(1,dy)를 대입

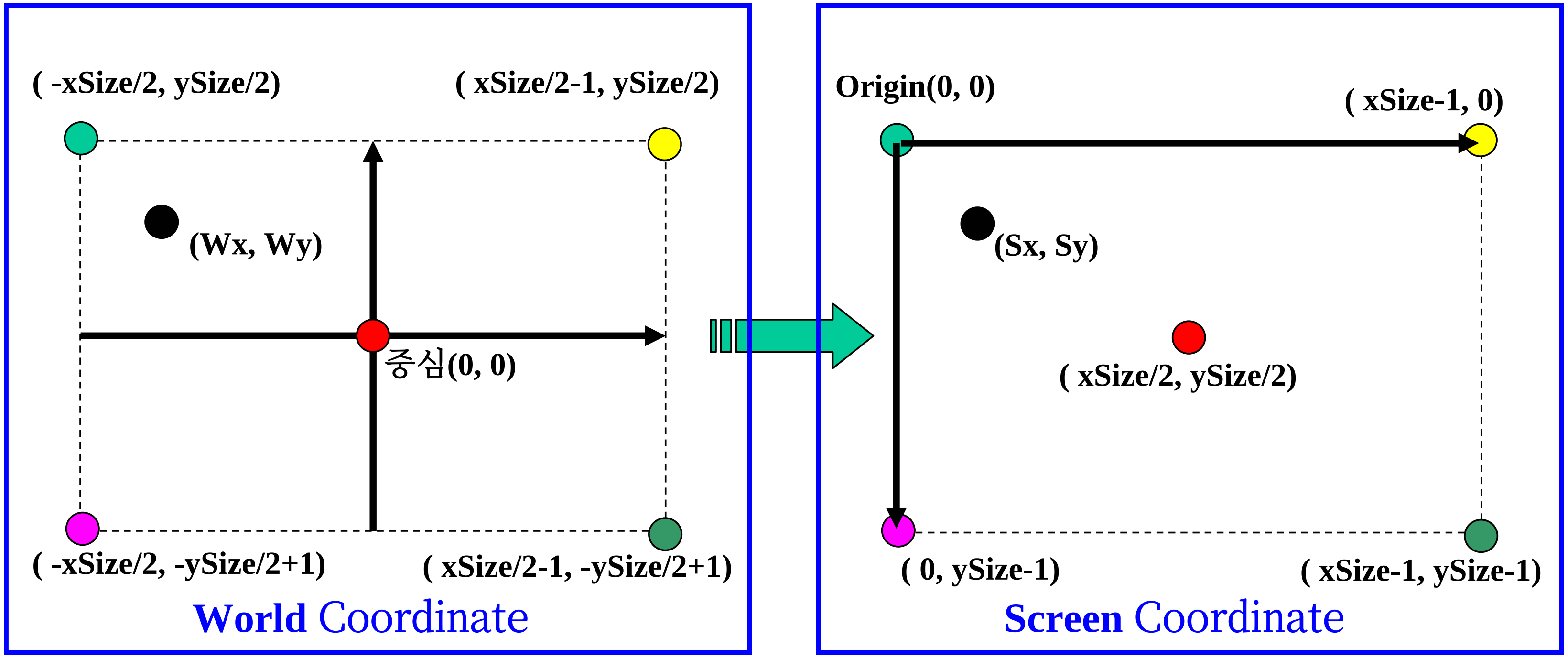
- $I(dx,dy) = \{ (1-dx) \times I(0,dy) + (dx) \times I(1,dy)$

  $+ (1-dy) \times I(dx,0) + (dy) \times I(dx,1) \}/2$

  $= (1-dx) \times (1-dy) \times I(0,0) + (dx) \times (1-dy) \times I(1,0)$

  $+ (1-dx) \times (dy) \times I(0,1) + (dx) \times (dy) \times I(1,1)$

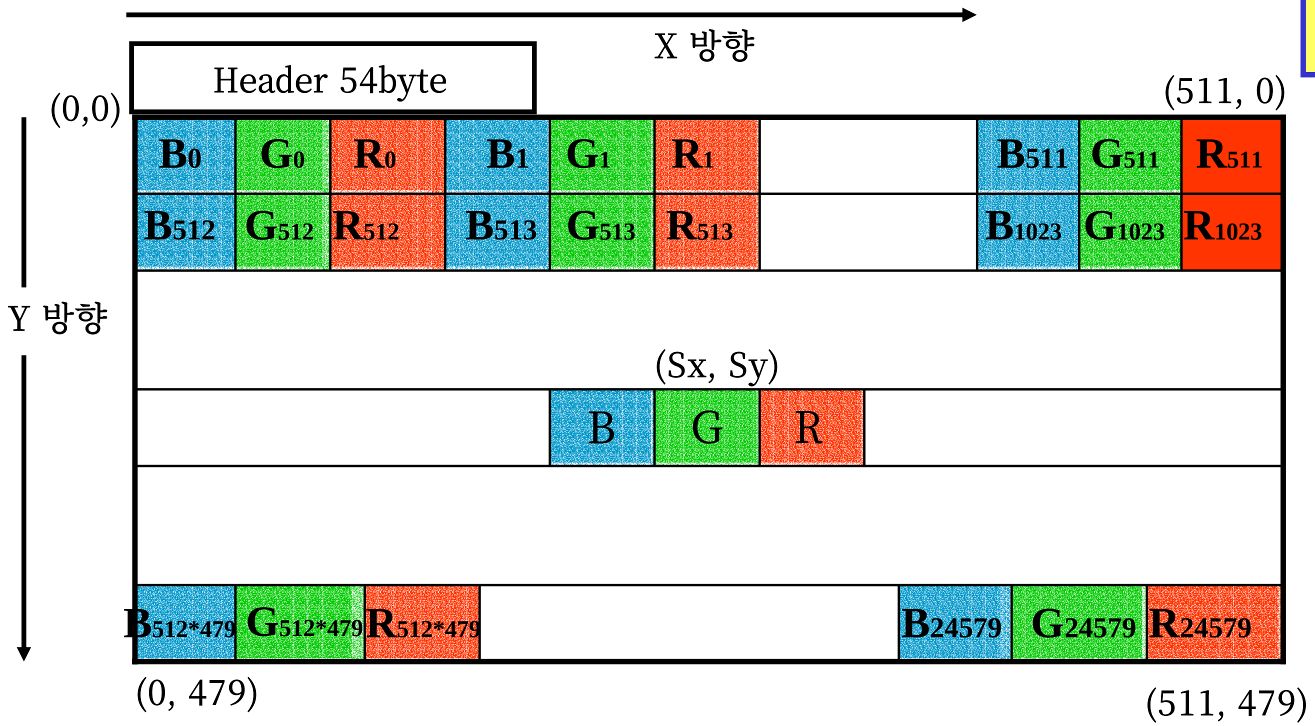
World Coordinates와 Screen Coordinates



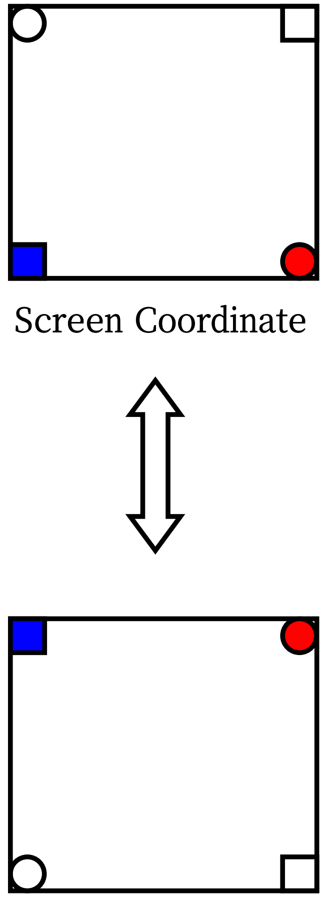
$Sx = Wx + xSize/2, \quad Sy = -Wy + ySize/2$

window size : xSize, ySize

The BMP Structure


X 방향
Header 54byte
(0,0)
(511, 0)
Y 방향
B0 G0 R0 B1 G1 R1 B511 G511 R511
B512 G512 R512 B513 G513 R513 B1023 G1023 R1023
(Sx, Sy)
B G R
B512*479 G512*479 R512*479 B24579 G24579 R24579
(0, 479)
(511, 479)

Screen의 위치와 bmp 메모리의 위치관계



▶ xSize = 512, ySize = 480의 경우

▶ 메모리 위치 계산

B = *(bmp+((ySize – 1-Sy)* xSize + Sx)*3)
G = *(bmp+((ySize – 1-Sy)* xSize + Sx)*3+1)
R = *(bmp+((ySize – 1-Sy)* xSize + Sx)*3+2)

● BMP 파일 구조

▶ 파일구조

Header (54 byte)	Data (xSize * ySize * 3 byte)

▶ 헤더 구조

File header (14 byte)	File information header (40 byte)

▶ 데이터

•헤더를 제외한 나머지가 bmp 파일의 데이터

•영상의 크기: 가로= **xSize**, 세로= **ySize** → 데이터의 크기 = **xSize** * **ySize** *3 (byte)

▶ 하나의 **bmp** 파일의 크기 : **54(byte) + xSize * ySize *3 (byte)**

(프로그램 코딩 시 주의)

▶ BITMAPFILEHEADER(Win32에서 file header 가 들어있는 구조체)

bfType (2 byte) :”BM” bmp 파일이라는 표시

bfSize (4 byte) : 이미지의 크기 *window의 크기를 (xsize, ysize)라 한다면 (xsize * ysize *3 + 54)

bfReserved1(2 byte) : 사용하지 않음 (0)

bfReserved2(2 byte) : 사용하지 않음 (0)

bfOffBits(4 byte) : 이미지 데이터가 들어있는 포인터(54)

▶ BITMAPINFOHEADER(Win32에서 file information header가 들어있는 구조체)

biSize (4 byte) : BITMAPINFOHEADER 의 크기 (40)

biWidth (4 byte) : 영상의 가로크기 (xsize)

biHeight (4 byte) : 영상의 세로크기 (ysize)

biplanes (2 byte) : 플레인 수(항상 1)

biBitCount (2 byte) : 비트카운터 (한 픽셀의 정보가 들어 있는 비트 수) (24)

biCompression (4 byte) : 압축유무(압축유: 1, 압축무: 0)

biSizeImage (4 byte) : 이미지의 크기 (xsize * ysize * 3)

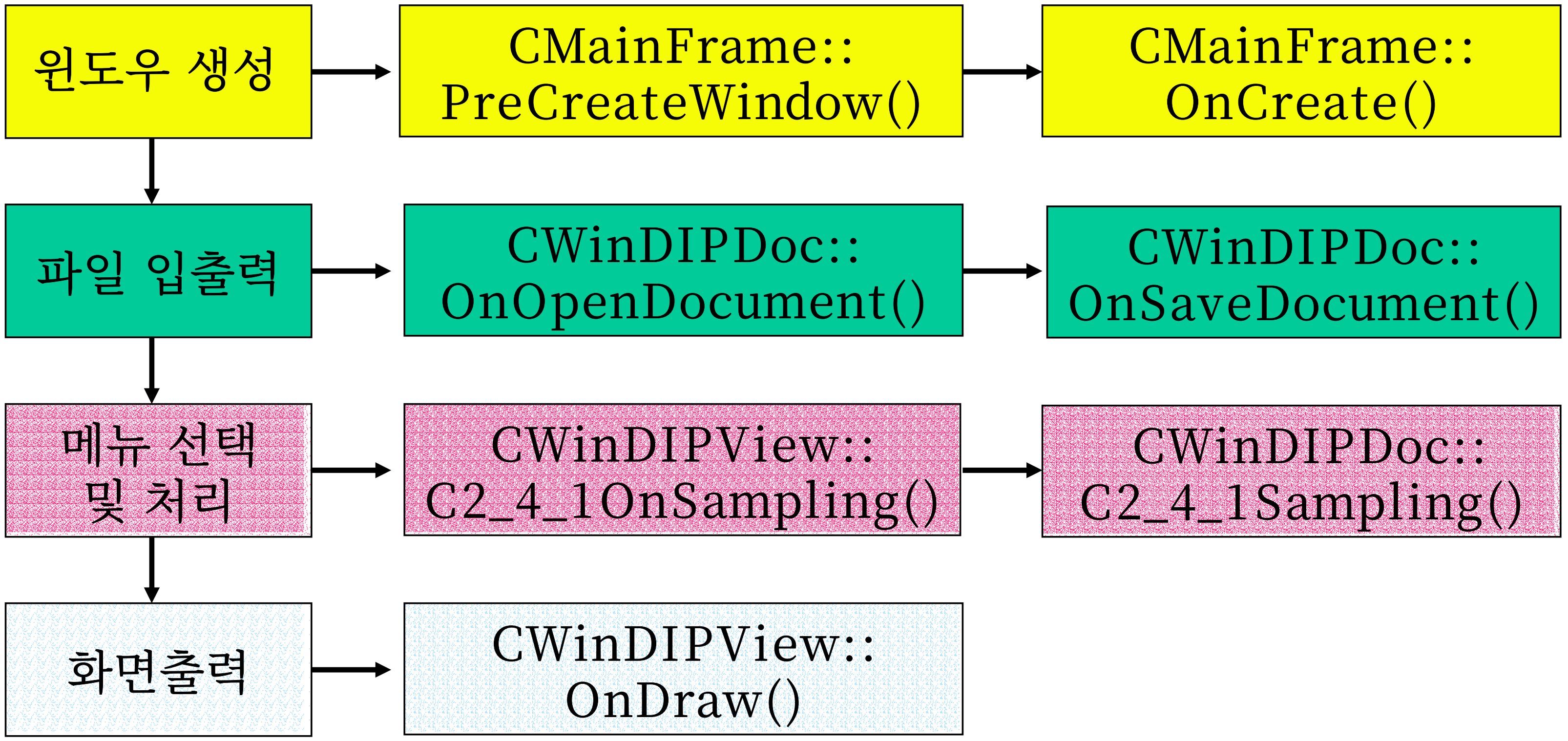
biXPelsPerMeter (4 byte) : 미터 당 가로픽셀 (일반적으로 0)

biYPelsPerMeter (4 byte) : 미터 당 세로픽셀 (일반적으로 0)

biClrUsed (4 byte) : 컬러 사용 유무 (0이면 비트맵은 모든 색상을 사용함)

biClrImprtant (4 byte) : 중요하게 사용하는 색(0이면 출력시 모든 색상이 다 사용됨)

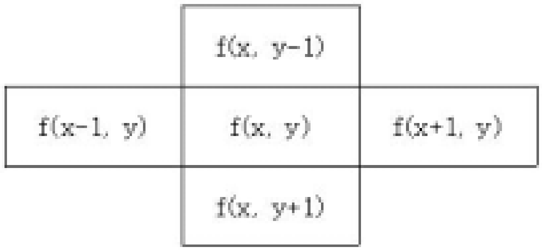
영상처리를 위한 MFC의 구조

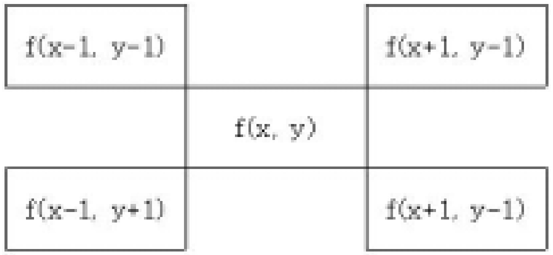

윈도우 생성
CMainFrame::PreCreateWindow()
CMainFrame::OnCreate()
파일 입출력
CWinDIPDoc::OnOpenDocument()
CWinDIPDoc::OnSaveDocument()
메뉴 선택 및 처리
CWinDIPView::C2_4_1OnSampling()
CWinDIPDoc::C2_4_1Sampling()
화면출력
CWinDIPView::OnDraw()

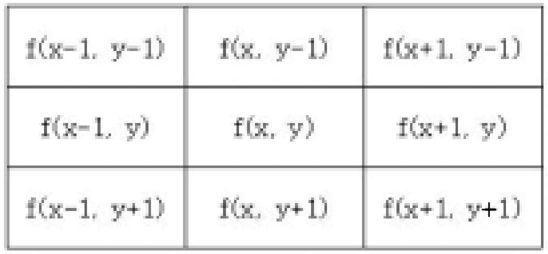
2.5 Some Basic Relationships between Pixels

2.5.1 Neighbors of a pixel (picture element)

4-neighbors = $N_4(p)$ **diagonal neighbors = $N_D(p)$** **8-neighbors = $N_8(p)$**







2.5.2 Adjacency, Connectivity, Regions, Boundaries(1)

- **Connectivity**
  - V : set of gray-level values used to define connectivity
  - **4-connectivity** : two pixels p & q with values from V are 4-connected if q is in the set $N_4(p)$
  - **8-connectivity** : two pixels p & q with values from V are 8-connected if q is in the set $N_8(p)$
  - **m-connectivity** :

    two pixels p and q with values form V are m-connected

    if q is in $N_4(p)$, or q is in $N_D(p)$ and the set $N_4(p) \cap N_4(q)$ is empty

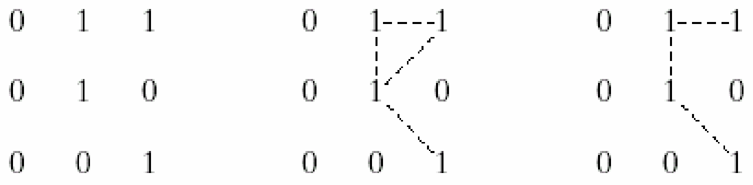
    Connectivity = Adjacency

◇ Adjacency, Connectivity, Regions, Boundaries(2)

- m-connectivity

	a	
b	c	

- a and b are not m-connected.
- If c is not there, then a and b are m-connected



a b c Arrange 8-adjacency m-adjacency

FIGURE 2.26 (a) Arrangement of pixels; (b) pixels that are 8-adjacent (shown dashed) to the center pixel; (c) *m*-adjacency.

◇ Adjacency, Connectivity, Regions, Boundaries(3)

- **Connected components**

For pixels **p** and **q** $\in$ image subset **S**,

If a path from **p** to **q**, then **p** is connected to **q**.

For any **p** in **S**,

the set of pixels connected to **p** is a connected component.

◇ Adjacency, Connectivity, Regions, Boundaries(4)

- Labeling of Connected Components
  - 4-connectivity
    - First Pass

```
If p =1 then
   if r=t=0, then
         assign new label to p
   else if  one of r,t=1, then
         assign that label to p
   else if r=1 and t=1, then
         assign any label to p  and record
         that labels for r,t are equivalent
```

	r	
t	p	

- Second Pass : relabel equivalent sets

◇ Adjacency, Connectivity, Regions, Boundaries(5)

- Labeling of Connected Components
  - 8-connectivity
    - First Pass

```
If p =1 then
   if q=r=s=t=0, then
        assign new label to p
   if  one of q,r,s,t=1, then
        assign that label to p
    else if ≥2  of q,r,s,t =1, then
        assign any label to p  and
        record as  equivalent
```

q	r	s
t	p	

    - Second Pass : relabel equivalent sets

프로그램 숙제

2.5.3 Distance Measures(1)

- Distance measures
  - Euclidean distance between p=(x,y) and q =(s,t)
    - $D_e(p, q) = [(x - s)^2 + (y - t)^2]^{1/2}$
  - D_4 distance(city-block distance) between p and q
    - $D_4(p, q) = |x - s| + |y - t|$
  - D_8 distance(chessboard distance) between p and q
    - $D_8(p, q) = \max(|x - s|, |y - t|)$

◇ Distance Measures(2)

- Distance measures
  - D_4 distance(city-block distance) between p and q

$$
\begin{array}{ccccc}
 & & 2 & & \\
 & 2 & 1 & 2 & \\
2 & 1 & 0 & 1 & 2 \\
 & 2 & 1 & 2 & \\
 & & 2 & &
\end{array}
\qquad\qquad
\begin{array}{ccccc}
2 & 2 & 2 & 2 & 2 \\
2 & 1 & 1 & 1 & 2 \\
2 & 1 & 0 & 1 & 2 \\
2 & 1 & 1 & 1 & 2 \\
2 & 2 & 2 & 2 & 2
\end{array}
$$

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

제2장 숙제

■ 숙제 샘플에 다음 기능을 갖는 프로그램을 설계한다.

1. 영상의 중앙을 중심으로 임의의 배수로 확대한다. 배수는 슬라이더로 정한다.
2. 마우스를 이용하여 영상을 임의의 크기로 잘라내어 저장한다.
3. 역치(Thresold)를 슬라이더로 정하여 영상을 이치화한다.
4. 8-connectivity를 이용하여 Labeling한다.

■ 프로그램 제출방법

1. 폴더명은 제2장(학생이름,학년)으로 한다.
2. 다른 사람의 프로그램을 복사한 경우는, 보여준 사람과 복사한 사람 모두 숙제 점수만큼 마이너스 점수를 준다.
3. 프로그램 방법을 교수 또는 다른 학생에게 질문을 할 수는 있지만, 프로그램 코드는 자기가 직접 작성한다
4. 실행 가능한 프로그램에서 Debug 폴더는 삭제하고 제출한다.
5. 제출기한 : 2주후

2.5.4 Image Operations on a Pixel Basis

- Image operations
  - Arithmetic operations
  - Logic operations

2.6 Linear and Nonlinear Operations

$$\boldsymbol{H}(a\boldsymbol{f} + b\boldsymbol{g}) = a\boldsymbol{H}(\boldsymbol{f}) + b\boldsymbol{H}(\boldsymbol{g})$$

f, g : two images

a, b : two scalar values

H : opeartor

H is **linear** if the above relation is satisfied.
Otherwise ***H*** is **nonlinear**.

Taking the average of two images is a **linear operation**.
Taking the absolute difference of two images is a **nonlinear operation**.

◇ Imaging Geometry (1)

- Some basic transformations
  - Translation
    - $x^* = X + X_0$
    - $Y^* = Y + Y_0$
    - $Z^* = Z + Z_0$

$$\Rightarrow \underbrace{\begin{bmatrix} X^* \\ Y^* \\ Z^* \\ 1 \end{bmatrix}}_{v^*} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{T} \underbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}_{v}$$

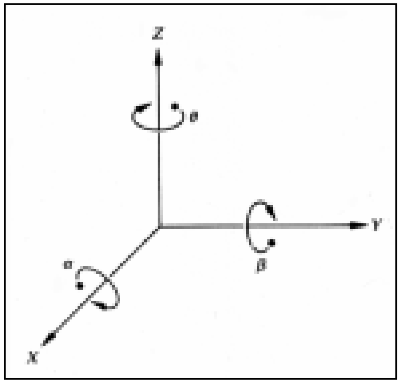
$$\boxed{v^* = T \quad v}$$

  - Scaling

$$S = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◇ Imaging Geometry (2)

- Some basic transformations
  - Rotation

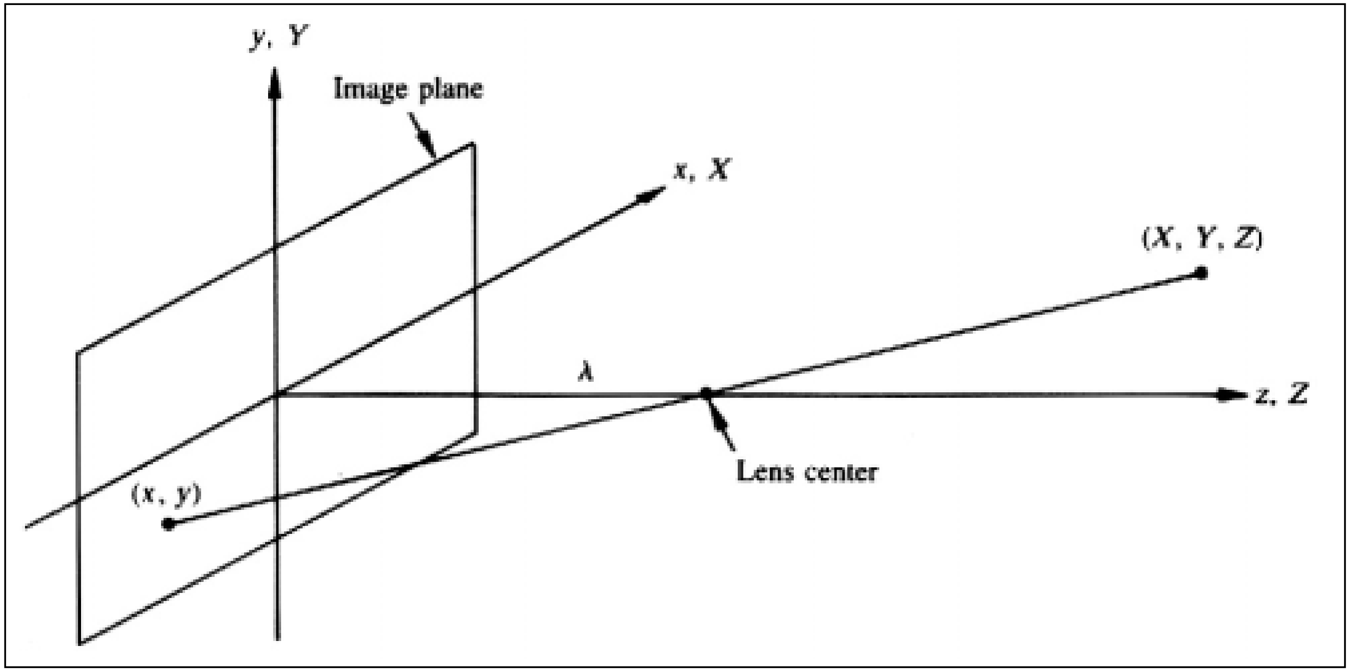


$$R_\theta = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_\alpha = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_\beta = \begin{bmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◇ Perspective Transformations (1)

- A perspective transformation projects 3-D points onto a plane



◇ Perspective Transformations (2)

- (X, Y, Z)
  - World coordinate system of 3D scene
- (x, y, z)
  - Camera coordinate system
- λ
  - Focal length of the lens

$$\frac{x}{\lambda} = -\frac{X}{Z-\lambda} = \frac{X}{\lambda - Z} \longrightarrow x = \frac{\lambda X}{\lambda - Z}$$

$$\frac{y}{\lambda} = -\frac{Y}{Z-\lambda} = \frac{Y}{\lambda - Z} \longrightarrow y = \frac{\lambda Y}{\lambda - Z}$$

- These equations are nonlinear.

◇ Perspective Transformations (3)

- Nonlinear form -> Linear form (using homogeneous coordinates)
  - Cartesian -> Homogeneous

$$w = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \qquad w_h = \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix}$$

(k : arbitrary nonzero constant.)

  - Conversion of homogeneous into Cartesian
    - Dividing the first three homogeneous coordinates by the fourth.

◇ Perspective Transformations (4)

- If we define the perspective transformation matrix as

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{-1}{\lambda} & 1 \end{bmatrix}$$

- The product pw_h yields a vector denoted C_h

$$c_h = Pw_h$$

(C_h are the camera coordinates in homogeneous form)

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{-1}{\lambda} & 1 \end{bmatrix} \begin{bmatrix} kX \\ kY \\ kZ \\ k \end{bmatrix} = \begin{bmatrix} kX \\ kY \\ kZ \\ \frac{-kZ}{\lambda} + k \end{bmatrix}$$

Perspective Transformations (5)

- Cartesian coordinates of any point in the camera coordinate system are given in vector form by

$$\mathbf{c} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{\lambda X}{\lambda - Z} \\ \frac{\lambda Y}{\lambda - Z} \\ \frac{\lambda Z}{\lambda - Z} \end{bmatrix}$$

◇ Perspective Transformations (6)

- The inverse perspective transformation

$$\mathbf{w}_h = \mathbf{P}^{-1}\mathbf{c}_h \quad , \quad p^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/\lambda & 1 \end{bmatrix}$$

- Suppose that an image point has coordinates(X_0, Y_0 , 0), where the 0 in the z location simply indicates that the image plane is located at z = 0. Then,

$$C_h = \begin{bmatrix} kX_0 \\ kY_0 \\ 0 \\ k \end{bmatrix} \quad \text{and} \quad W_h = \begin{bmatrix} kX_0 \\ kY_0 \\ 0 \\ k \end{bmatrix}$$

Perspective Transformations (7)

- This results in $$w = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ 0 \end{bmatrix}$$

- Why always Z=0?

◇ Perspective Transformations (8)

- Formulate the inverse perspective transformation by using the z component of C_h as a free variable instead of 0.

$$\mathbf{c}_h = \begin{bmatrix} kx_0 \\ ky_0 \\ kz \\ k \end{bmatrix} \quad \text{…(2.5-32)} \qquad \mathbf{w}_h = \begin{bmatrix} kx_0 \\ ky_0 \\ kz \\ \frac{kz}{\lambda} + k \end{bmatrix} \quad \text{…(2.5-33)}$$

◇ Perspective Transformations (9)

- From Eq.2.5-33, we have as Cartesian coordinates

$$w = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{\lambda x_0}{\lambda + z} \\ \frac{\lambda y_0}{\lambda + z} \\ \frac{\lambda z}{\lambda + z} \end{bmatrix} \Rightarrow \begin{aligned} X &= \frac{\lambda x_0}{\lambda + z} \\ Y &= \frac{\lambda y_0}{\lambda + z} \\ Z &= \frac{\lambda z}{\lambda + z} \end{aligned}$$

◇ Perspective Transformations (10)

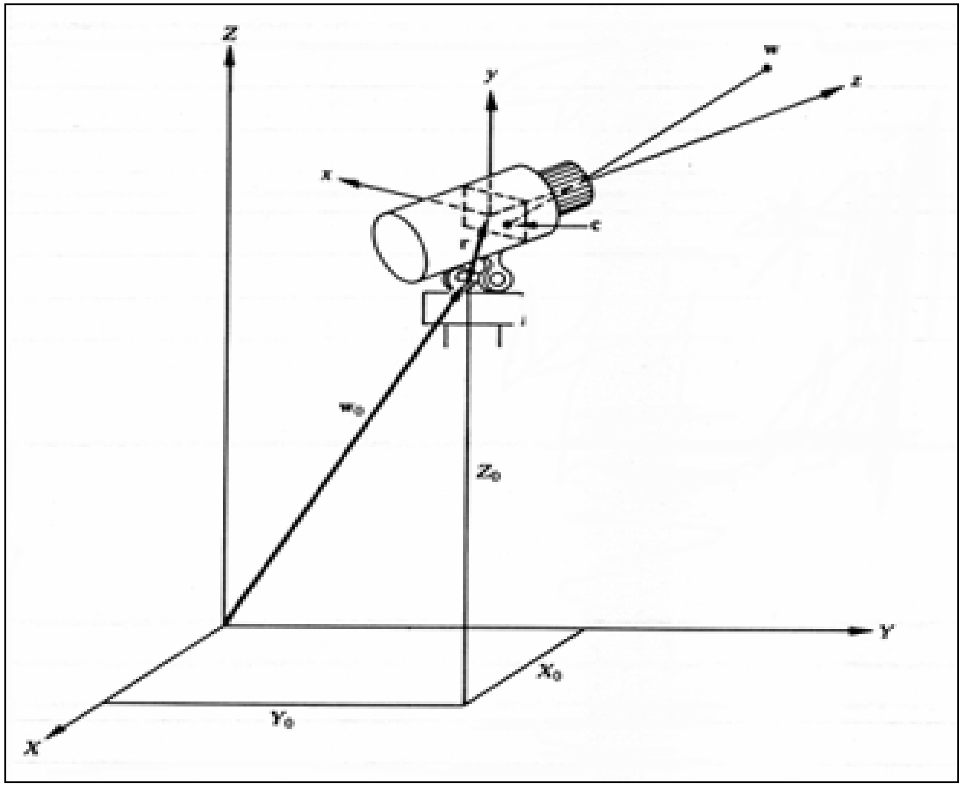
- Solving for z in terms of Z in the last equation.
- Substituting in the first two expressions yields.

$$X = \frac{x_0}{\lambda}(\lambda - Z) \quad , \quad Y = \frac{y_0}{\lambda}(\lambda - Z)$$

- Recovering a 3-D point from its image by means of the inverse perspective transformation requires knowledge of at least one of the world coordinates of the point.

◇ Camera Model(1)

- So far we assumed that world and camera coordinate systems are aligned.
- Now we generalize this.
- World coordinate system(X,Y,Z) used to locate both the camera and 3-D point (w)



◇ Camera Model(2)

- We try to bring camera and world coordinate systems into alignment by applying a set of transformations.
- **(1) Translation of the origin of the world coordinate system to the location of the gimbal center** is accomplished by using the transformation matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & -X_0 \\ 0 & 1 & 0 & -Y_0 \\ 0 & 0 & 1 & -Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◇ Camera Model(3)

- GW_h is new coordinate system after the transformation.
- **(2) pan of the x axis, (3) tilt of the z axis.**

$$R = R_\alpha R_\theta = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta\cos\alpha & \cos\theta\cos\alpha & \sin\alpha & 0 \\ \sin\theta\sin\alpha & -\cos\theta\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- **(4)Displacement of the image plane with respect to the gimbal center.**

$$C = \begin{bmatrix} 1 & 0 & 0 & -r_1 \\ 0 & 1 & 0 & -r_2 \\ 0 & 0 & 1 & -r_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◇ Camera Model(4)

- Thus applying to W_h the series of transformations $CRGW_h$ brings the world and camera coordinate systems into coincidence.

$$c_h = PCRGW_h$$

$$x=\lambda\frac{(X-X_0)\cos\theta+(Y-Y_0)\sin\theta-r_1}{-(X-X_0)\sin\theta\sin\alpha+(Y-Y_0)\cos\theta\sin\alpha-(Z-Z_0)\cos\alpha+r_3+\lambda}$$

$$y=\lambda\frac{-(X-X_0)\sin\theta\cos\alpha+(Y-Y_0)\cos\theta\cos\alpha+(Z-Z_0)\sin\alpha-r_2}{-(X-X_0)\sin\theta\sin\alpha+(Y-Y_0)\cos\theta\sin\alpha-(Z-Z_0)\cos\alpha+r_3+\lambda}$$

◇ Camera Calibration(1)

- Camera calibration
  - The computational procedure used to obtain the camera parameters using known points.
- $C_h = AW_h$,A : All the camera parameters.

$$\begin{bmatrix} c_{h1} \\ c_{h2} \\ c_{h3} \\ c_{h4} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \text{…..(2.5-44)}$$

◇ Camera Calibration(2)

- Camera coordinates in Cartesian form

$$x = c_{h1} / c_{h4} \;, \quad y = c_{h2} / c_{h4}$$

- Substituting $c_{h1} = xc_{h4}$, $c_{h2} = yc_{h4}$ in Eq.(2.5-44) , expanding the matrix product yields (C_{h3}: ignored)

$$\begin{aligned} xc_{h4} &= a_{11}X + a_{12}Y + a_{13}Z + a_{14} \\ yc_{h4} &= a_{21}X + a_{22}Y + a_{23}Z + a_{24} \\ c_{h4} &= a_{41}X + a_{42}Y + a_{43}Z + a_{44} \end{aligned}$$

◇ Camera Calibration(3)

- Substitution of C_{h4} yields 2 equations with 12 unknowns

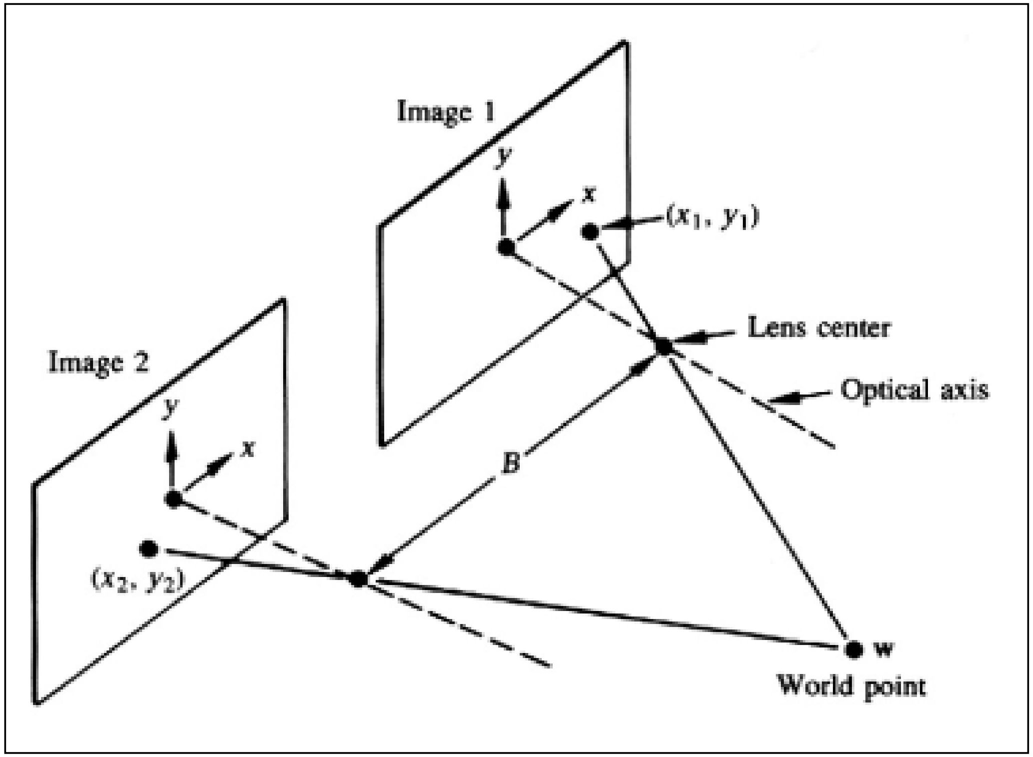
$$a_{11}X + a_{12}Y + a_{13}Z - a_{41}xX - a_{42}xY - a_{43}xZ - a_{44}x + a_{14} = 0 \quad \ldots(2.5\text{-}48)$$

$$a_{21}X + a_{22}Y + a_{23}Z - a_{41}yX - a_{42}yY - a_{43}yZ - a_{44}y + a_{24} = 0 \quad \ldots(2.5\text{-}49)$$

- The Calibration procedure
  - (1) obtain m>=6 world points with known coordinates(Xi,Yi,Zi)
  - (2) project these points with the camera in a given position to obtain the corresponding image points(xi,yi)
  - (3) use these results in Eqs.(2.5-48)&(2.5-49) to solve for the unknown coefficients. **-> first linear, then nonlinear problem solving**

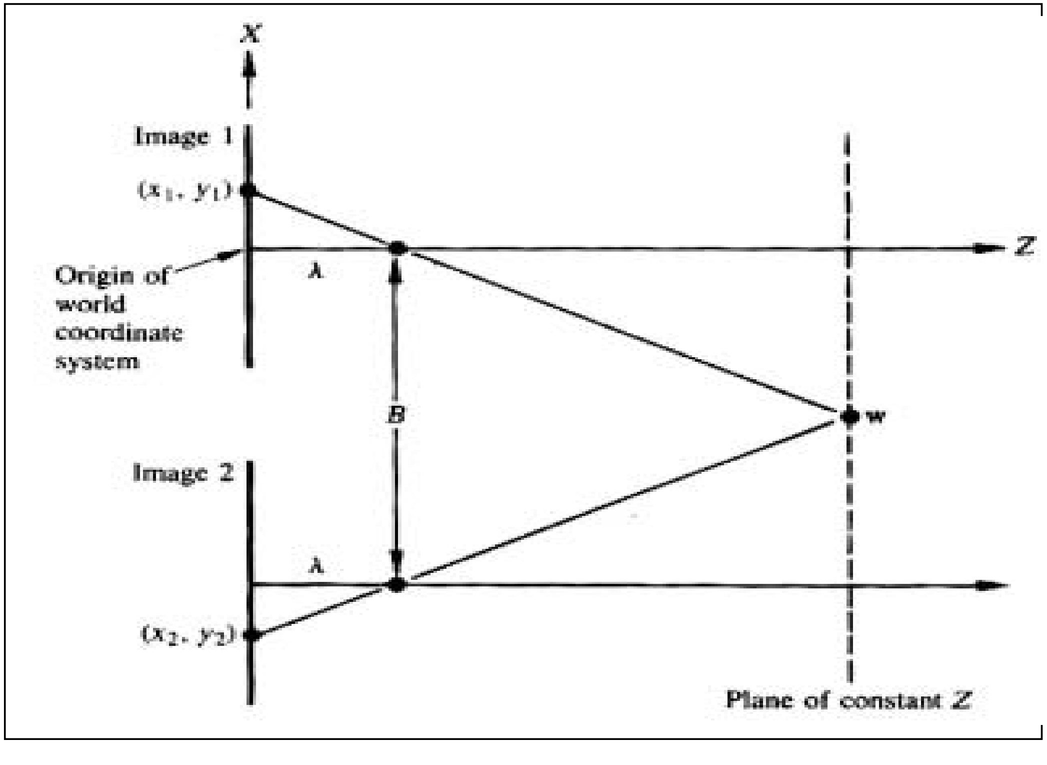
◇ Stereo Imaging (1)

- Model of the stereo imaging process.



◇ Stereo Imaging (2)

- Top view of the stereo imaging process



◇ Stereo Imaging (3)

$$X_1 = \frac{x_1}{\lambda}(\lambda - Z_1) \quad , \quad X_2 = \frac{x_2}{\lambda}(\lambda - Z_2)$$

since $X_2 = X_1 + B$ & $Z_2 = Z_1 = Z$

$$X_1 = \frac{x_1}{\lambda}(\lambda - Z) \quad , \quad X_1 + B = \frac{x_2}{\lambda}(\lambda - Z)$$

$$Z = \lambda - \frac{\lambda B}{x_2 - x_1} = \lambda\left(1 - \frac{B}{x_2 - x_1}\right)$$

◇ Stereo Imaging (4)

- Problem in stereo imaging
  - **Correspondence problem**

    find 2 corresponding image points from the same 3-D point
  - Solution

    $\rightarrow$ Use **epipolar line constraint** (2 corresponding image points have the same y coordinate)