Introduction to Machine Vision
Part 1

Professor Kok-Meng Lee
Georgia Institute of Technology
George W. Woodruff School of Mechanical Engineering
Atlanta, GA 30332-0405

Webpage: www.me.gatech.edu/me6406

Course Outline

- Introduction and low-level processing
  - Physics of digital images, histogram equalization, segmentation, edge detection, linear filters.
- Model-based Vision
  - Hough transform, pattern representation, matching
- Geometric methods
  - Camera model, calibration, pose estimation
- Neural network for machine vision
  - Basics, training algorithms, and applications
- Color images and selected topics
  - Physics, perception, processing and applications

Reference Textbooks:

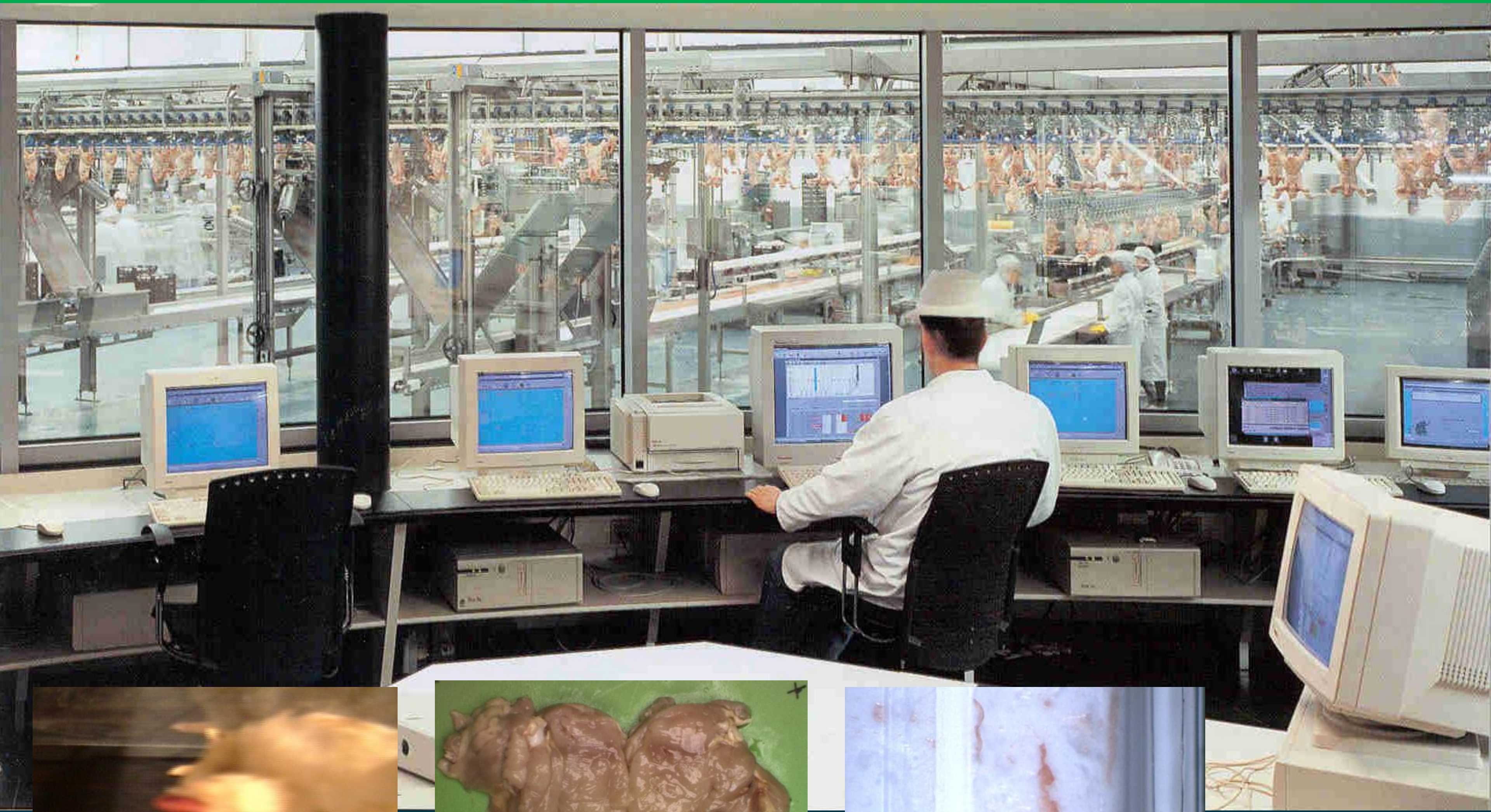
1. Ballard and Brown, Computer Vision, 1982.
2. Horn, Robot Vision, MIT Press. 1986.
3. Davies, Machine Vision: Theory, Algorithm & Practicalities, 1997.
4. Shapiro and Stockman, Machine Vision, 2001
5. Forsyth, and Ponce, Computer Vision: A Modern Approach, 2003
6. Gonzalez, Woods and Eddins, Digital Image Processing using Matlab, 2004.
7. Gonzalez and Woods, Digital Image Processing, 3rd Edition, 2008.

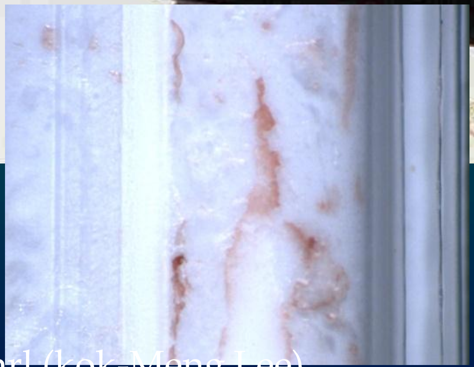
Image Interpretation Applications

- Robotics and automation
- Food processing applications
- Medical radiographic images, functional magnetic resonance imaging (fMRI), medical ultrasound
- Industrial radiographic images, security images
- Satellite images, astronomy

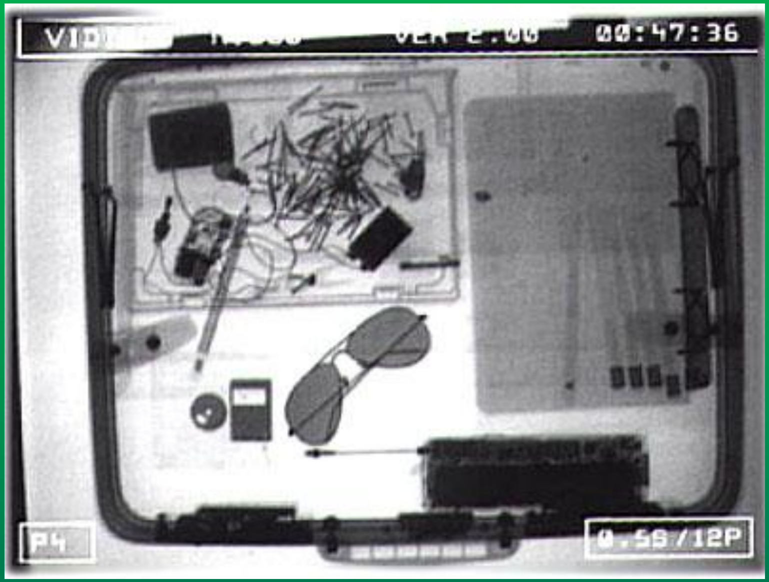
Robotics and Automation

Food processing automation





Industrial Radiographic Image



www.vidisco.com/ CabinetXrayMic80A_01.htm

Pseudo-color

Identification imaging

Finger print

Medical Radiographic Image

Right elbow

www.4umi.com/image/x-ray.jpg

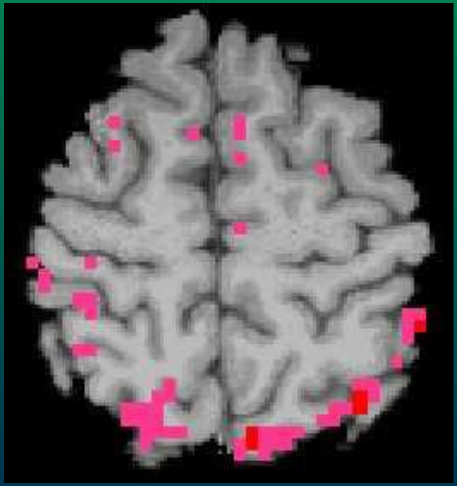
Medical Ultrasound

http://keystone.stanford.edu/~huster/photos/i/ultrasound.640.jpg

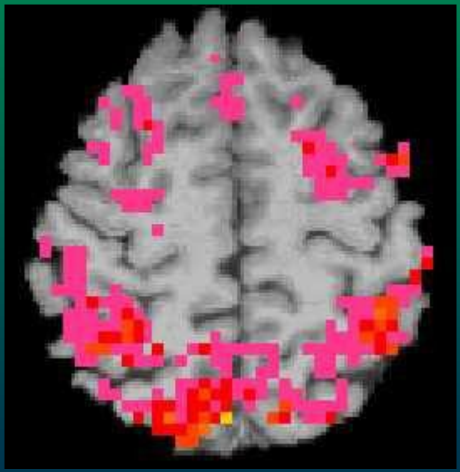
Functional MRI

Response to the spatial working memory task. Brain activation is shown in bright colors.

www.alcoholism2.com/



*A 20-year old female **drinker***



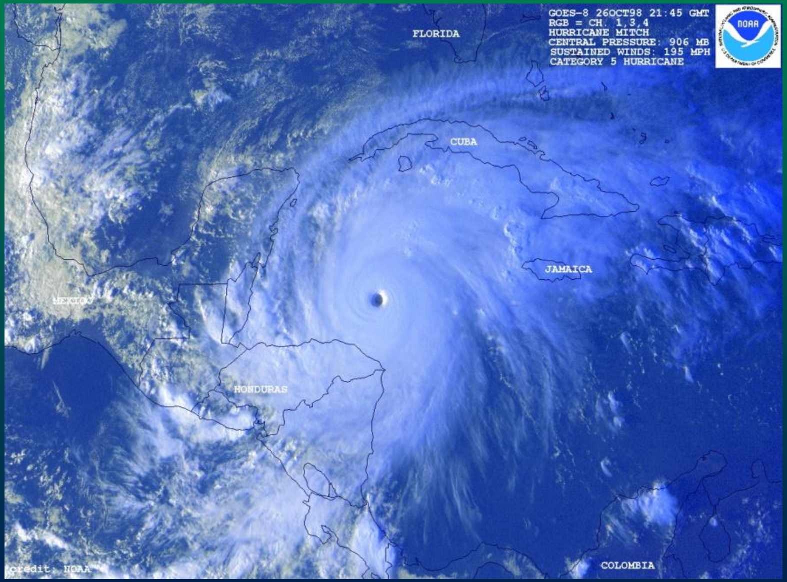
*A 20-year old female **nondrinker***

Astronomy Images

www.sdsc.edu/ sciencegroup/astronomy/

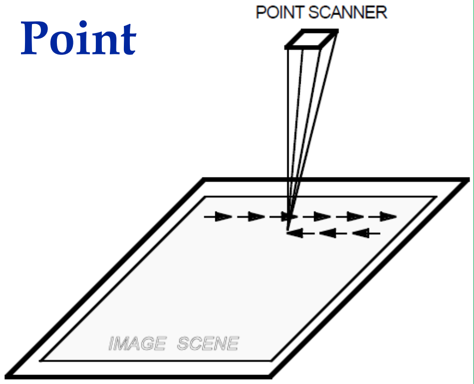
astro.martianbachelor.com/

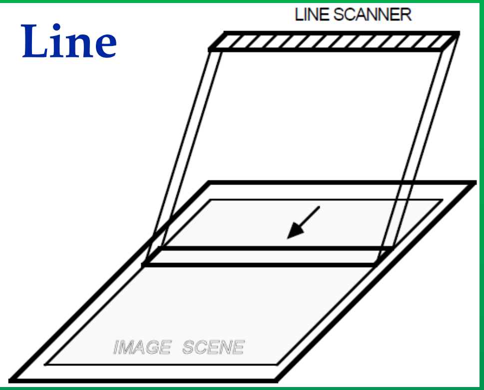
Satellite Images

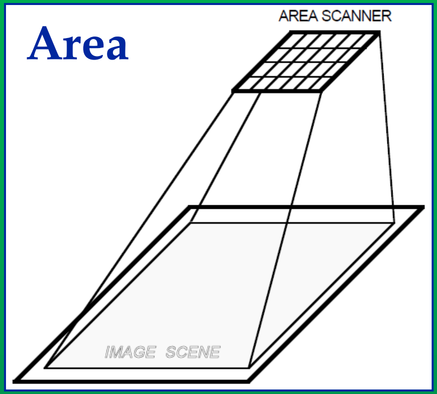


www.noaa.gov

CCD Formats



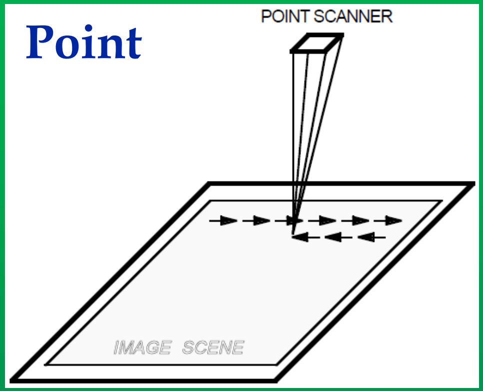


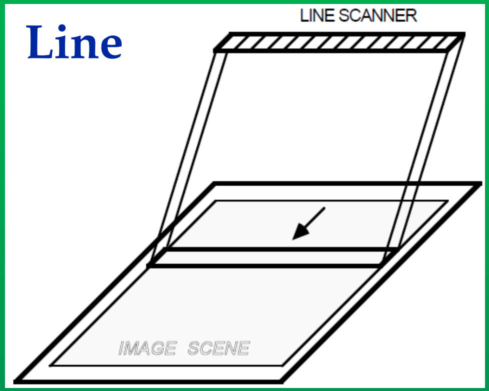


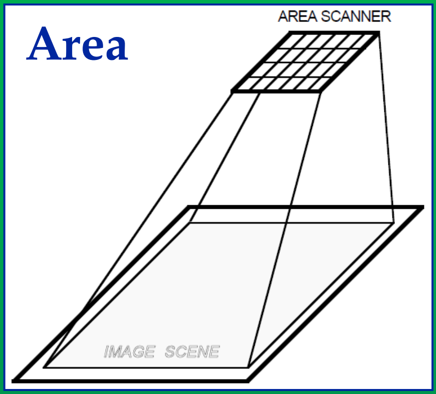
Point Scanning

- Single-cell detector, or pixel (picture element), scanned sequentially
- Advantages: High resolution, lower cost ,simplicity of the detector.
- Disadvantages :
  - Registration errors from the X,Y movement of scene or detector,
  - lower frame scanning rates (because the repeated number of exposures increases the scanning time), and
  - system complexity (because of the X,Y movement).

CCD Formats



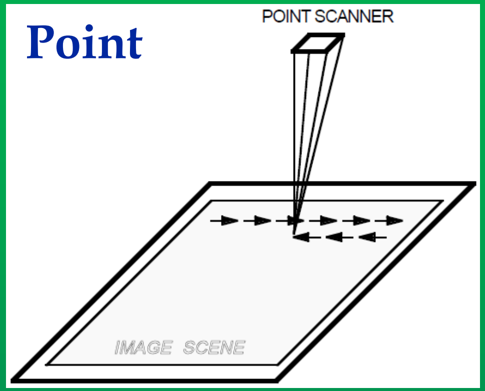


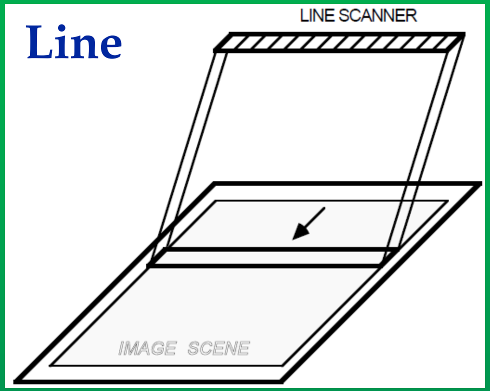


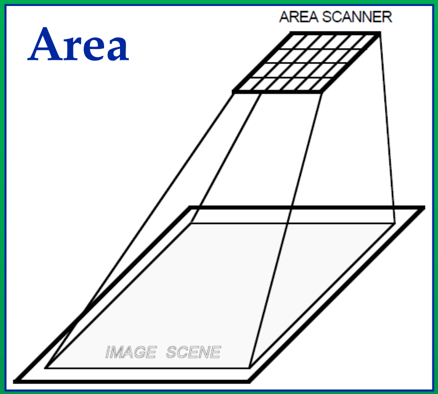
Line Scanning

- Single-cell detectors placed along a single axis, scanned by line.
- Advantages: High resolution, less sophisticated mechanics than PS.
- Disadvantages :
  - Physical length of a linear CCD scanner is limited only by the size of the silicon wafer used to make the device.
  - Resolution is limited by the pixel spacing and size in one direction.
  - Measurement accuracy at each pixel has finite non-uniformities that occasionally must be factored out by the system
  - Scan times, of the order of several seconds or minutes, are still unsuitable for many applications.

CCD Formats



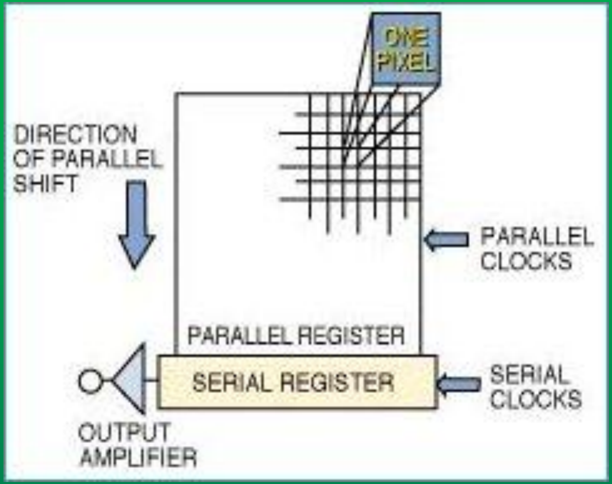




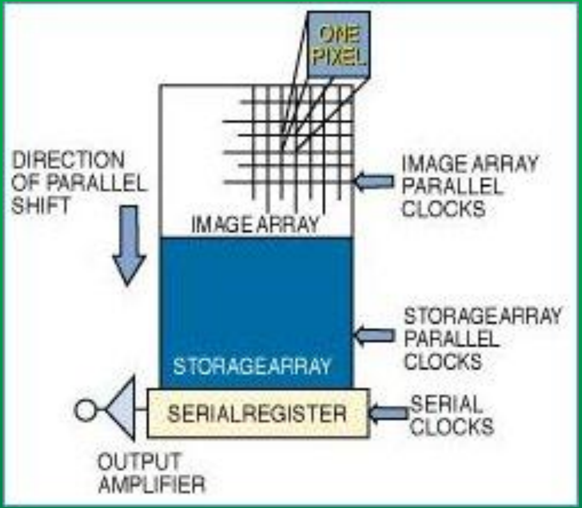
Area Scanning

- 2-D array of detectors capture the entire image with one exposure, eliminating the need for movement by detector or scene.
- Advantages:
  - Highest frame rates with the greatest registration accuracy between pixels.
  - System complexities are also kept to a minimum.
- Disadvantages:
  - Resolution is limited in two directions.
  - Higher cost and lower signal-to-noise performance.

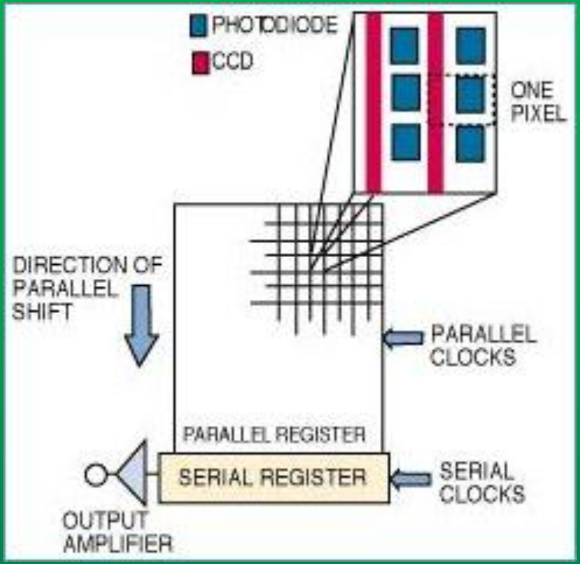
CCD Architectures



Full-Frame (FF) Devices.



Frame-Transfer (FT) Devices

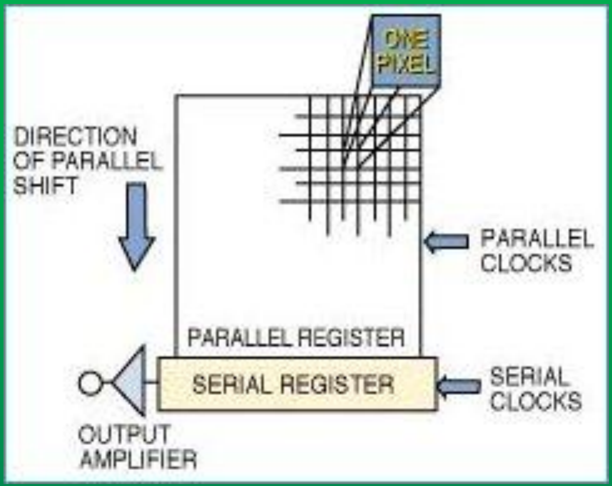


Interline (IL) Devices.

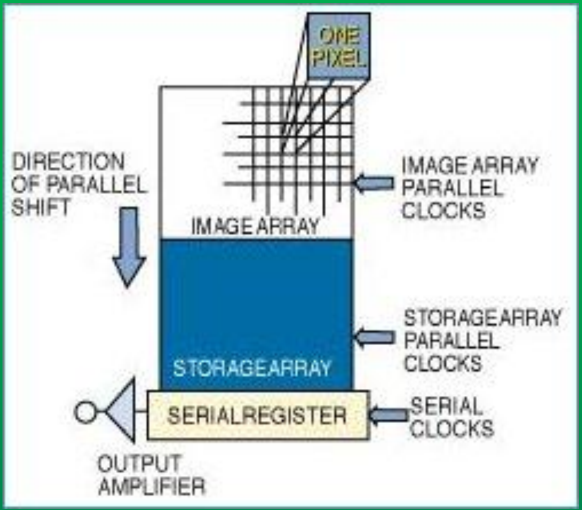
Full-Frame (FF) Devices

- Exposure is controlled by a mechanical shutter or strobe.
- The resultant charges are shifted one row at a time to the serial register. After a row is read out by the serial register, the next row is shifted to the register for readout. The process is repeated until all rows are transferred, at which point the array is ready for the next exposure.

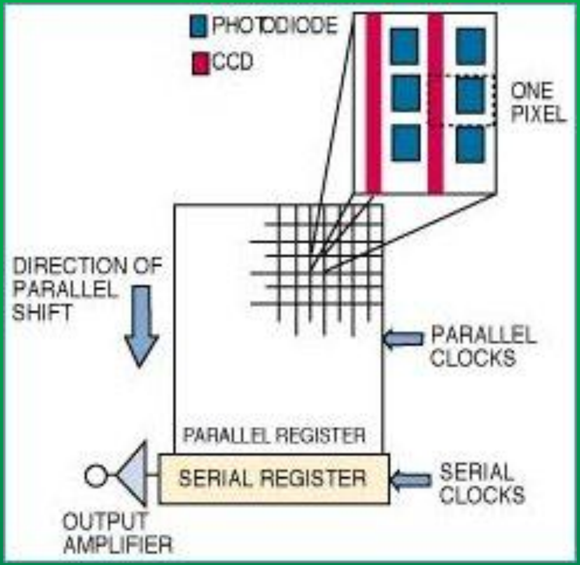
CCD Architectures



Full-Frame (FF) Devices.



Frame-Transfer (FT) Devices

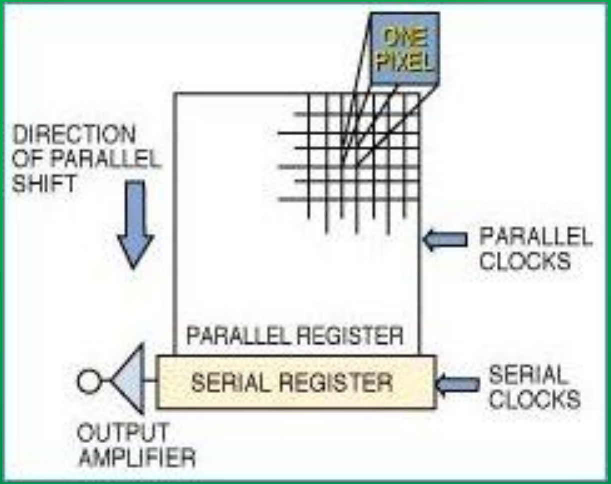


Interline (IL) Devices.

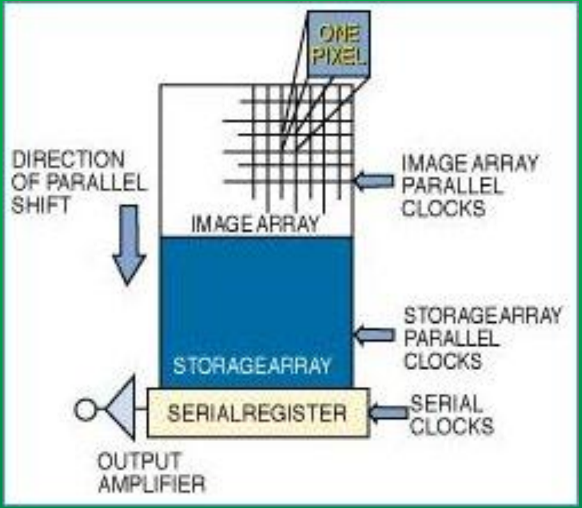
Frame-Transfer (FT) Devices

- Shutterless operation: Entire array is shifted quickly to an identically sized storage array. Same readout as in a full-frame device, but the frame rates are increased because the image can begin the next exposure while readout.
- Disadvantages:
  - Performance is compromised because integration is still occurring during the image dump to the storage array, which results in image smear.
  - Because twice the silicon area is required to implement this architecture, FT CCDs have lower resolutions and higher costs than FF CCDs.

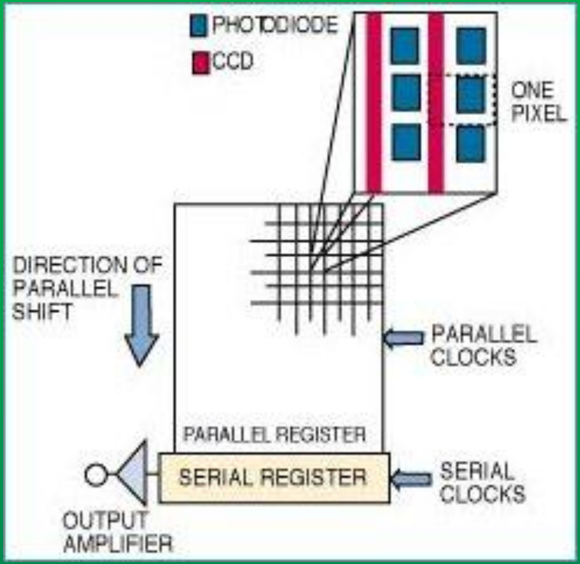
CCD Architectures



Full-Frame (FF) Devices.



Frame-Transfer (FT) Devices

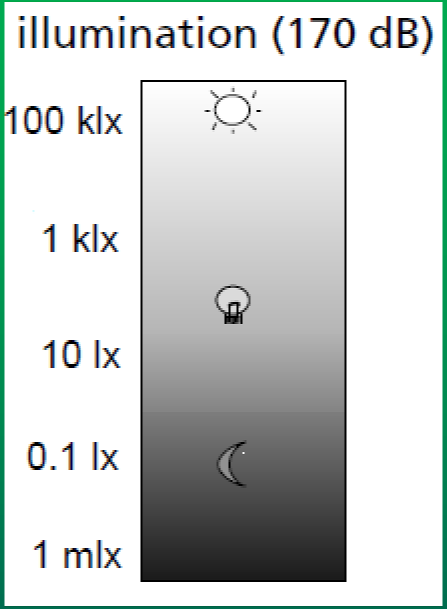


Interline (IL) Devices.

Interline (IL) Devices

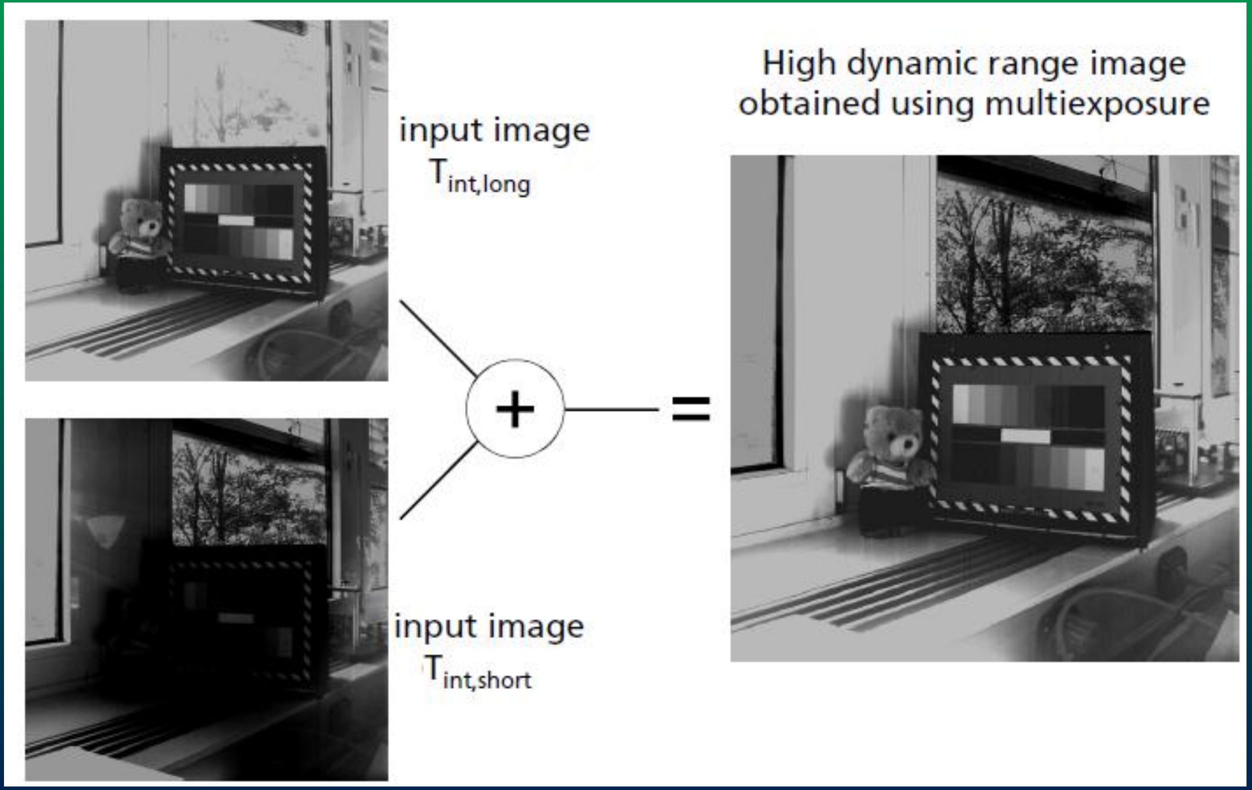
- Each pixel is composed of a photodiode and an associated light-shielded CCD. After exposure, the charges in each photodiode are immediately transferred to the adjacent CCD, where the readout process is conducted.
- Frame transfer rate is even faster than a frame-transfer device.
- Disadvantages:
  - Device complexity, higher unit costs .
  - Lower sensitivity and greater quantization, or sampling, errors because less photosensitive area (reduced aperture).
  - Suffer image lag as a consequence of charge transfer from photodiode to CCD.

Recent development



Dynamic range of solid-state image sensors varies over wide range:

- high end CCDs > *78dB*
- consumer grade CCDs 66dB
- consumer grade CMOS imagers 54dB



High Dynamic Range CMOS Image Sensor with On-Chip Programmable Region of Interest Readout

Comparison between CCD and HDRC

CCD

HDRC

Machine Vision Architectures

Questions asked — *FIVS developed* — Algorithms for RAM applications

1980 — *1990* — *2000* — *2010*

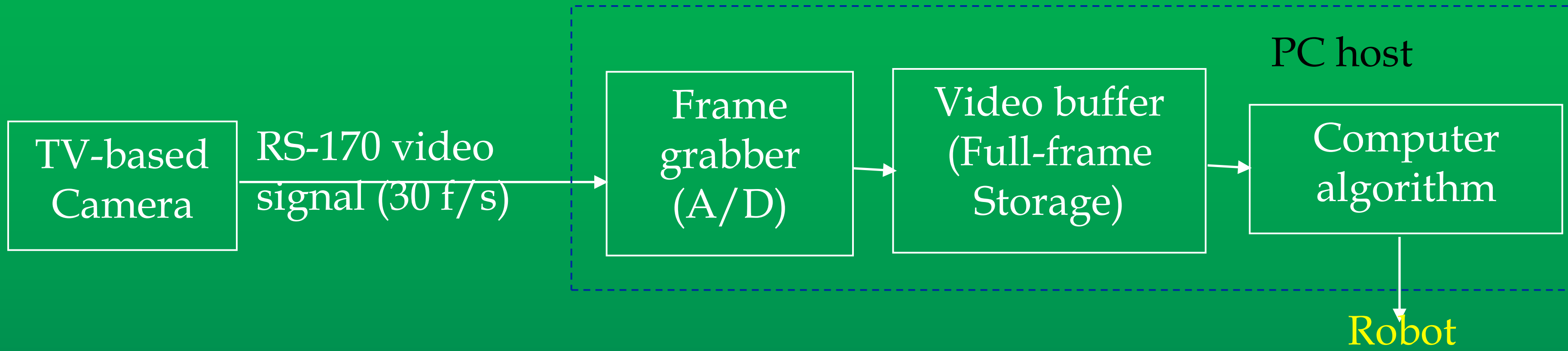
Some questions we asked in the 1980's

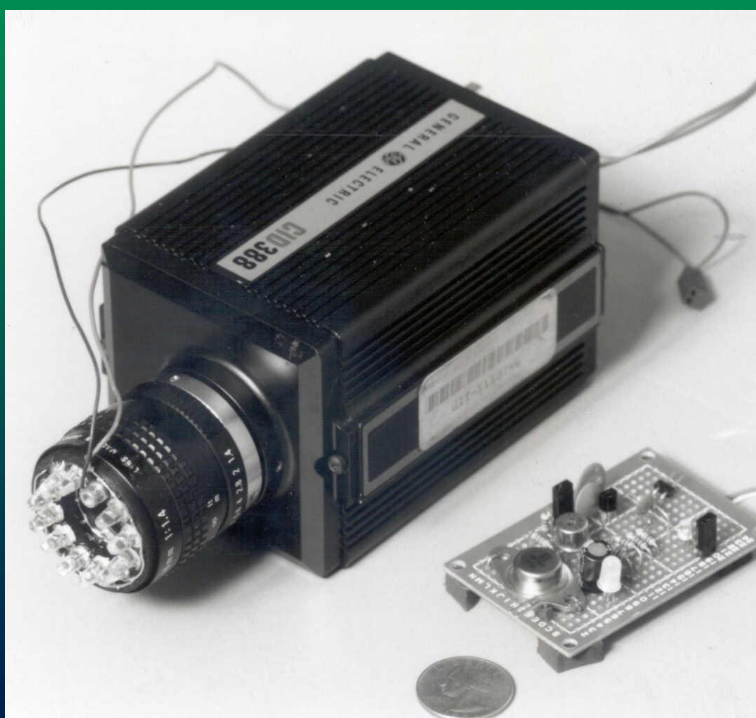
Question #1: Why a color video camera costs only US$1,500 but a gray-level machine vision system costs US$50,000?

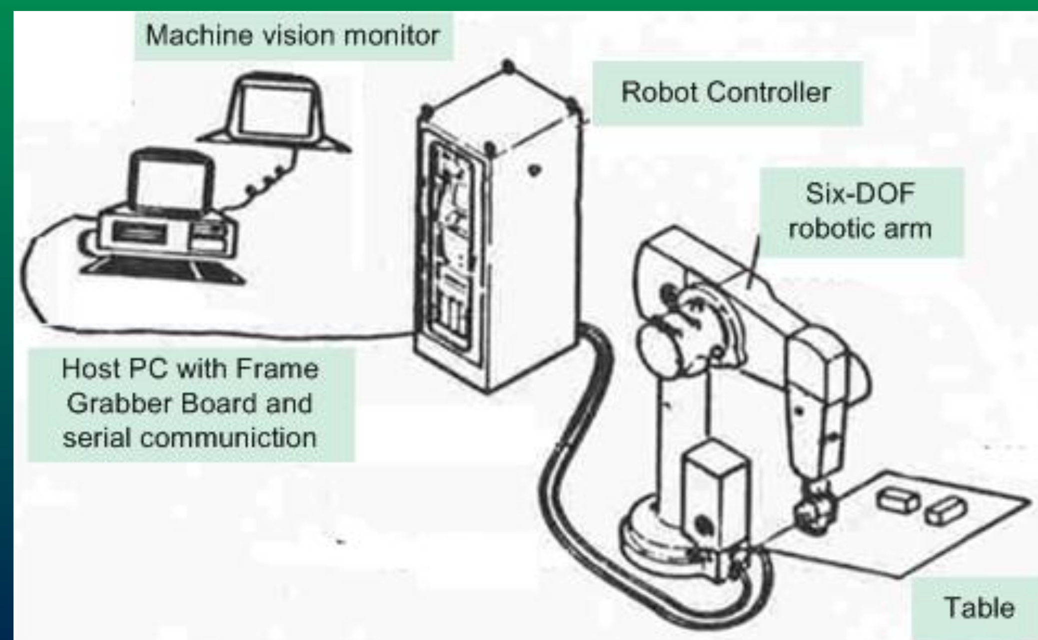
Question #2: Why do the we need 0.5 second to compute the center of a small blob?

Question #3: We have long noticed that a human visual system (HVS) functions remarkably well even in the presence of significant noises. Can we effectively emulate some or all the HVS features to improve machine vision for automating the tasks of visual inspection?

TV-Camera-based Imaging systems







Question #1: Cost

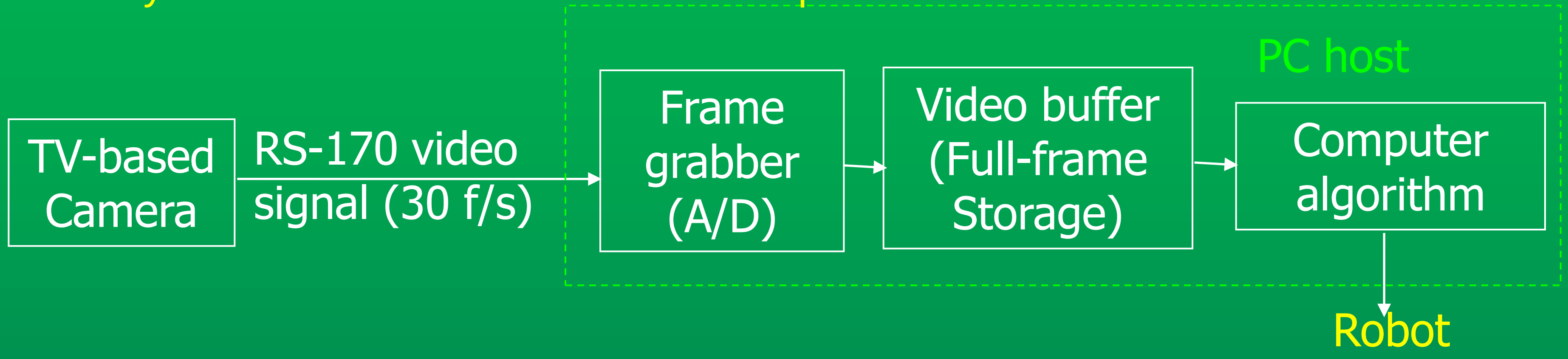
Why a color video camera costs only US$1,500 but a gray-level machine vision system costs US$50,000?

A primary difference between human visual perception and machine vision:

- A color video camera (based on standard established by the TV industry) for human vision requires only *qualitative* information.
- Machine controlled system, like a robot, requires *quantitative* information to do work.

Question #2: Processing time

Why do the we need 0.5 second to compute the center of a small blob?



RS-170 TV-standard established in the 1950's limits the image readout at a rate of 30 fps (frames per second) or 33ms.

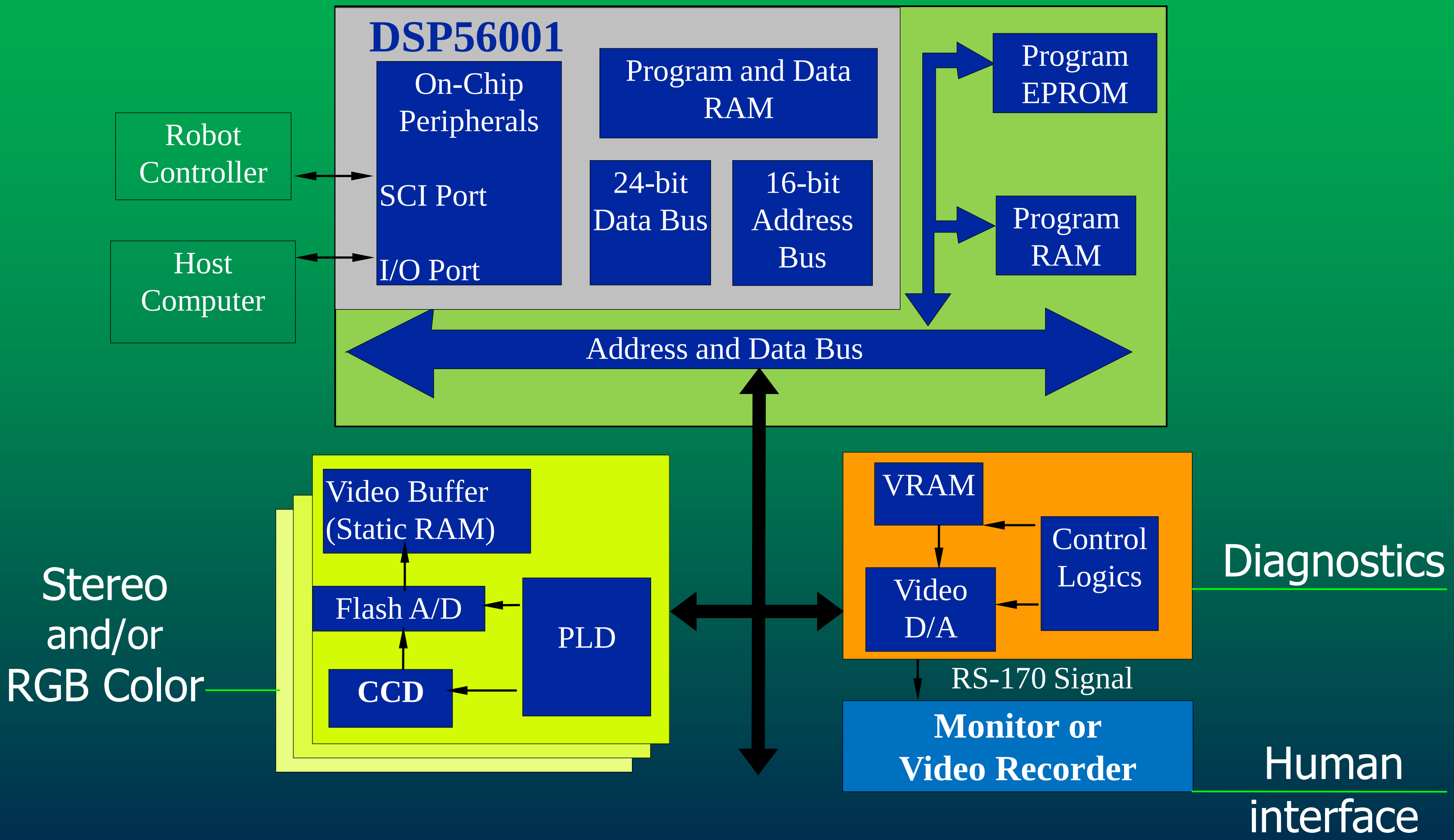
- A frame-grabber is needed to convert the RS-170 video signal into a series of n bit brightness values (gray levels) via a flash A/D converter.
- Digitized image data must be stored in a memory buffer before processing, often only a few of these carry the information on which the vision system will base a decision.

2. Flexible Integrated Vision System (FIVS) - An alternative vision system design

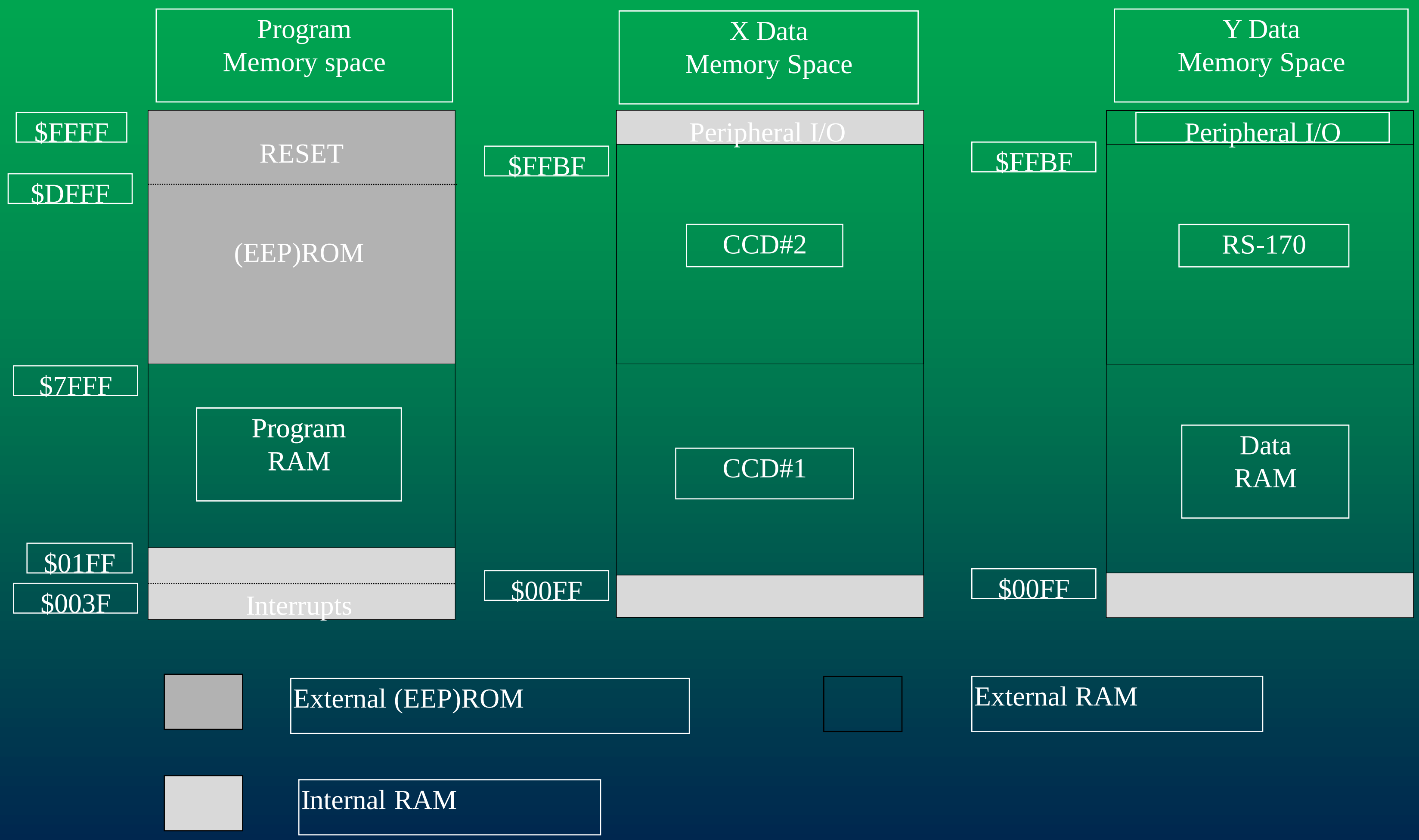
- **To overcome the problems associated with the traditional video-based vision system:**
  - Offer performance and cost advantages by integrating the imaging sensor, control, illumination, digitization, computation, and data communication in a single unit.
  - Eliminating the PC-based frame grabber, the camera is no longer restricted by the RS-170 standard and thus frame rates higher than 30 fps can be achieved.
- **Digital camera:**
  - On-board micro-processor
  - Imbedded software
  - Direct illumination control

Flexible Integrated Vision System

FIVS Architecture



Flexible Integrated Vision System *FIVS Memory map*



Prototype FIVS

developed at Georgia Tech (Lee and Blenis, 1992)

- Texas Instruments TC211 CCD (192X165)
- 25 MHz Motorola MC10319 8-bit A/D Converter.
- 20MHz Motorola DSP56001
- RS232 Serial Communication baud-rate (115.2 KB >20KB)
- Intel-386 33MHz Host PC

Mathematical representation of Digital Images

g(x,y) 

$$0 \le g \le (2^8 - 1) \text{ or } 255$$

- Matrix of numbers
- Each number represents a picture element – 'pixel'
- Pixels are parameterized by
  - x – y position
  - Intensity: n bit or 2^n gray-levels, color)
  - Time
- Binary image

$g_{th} = \text{threshold value}$

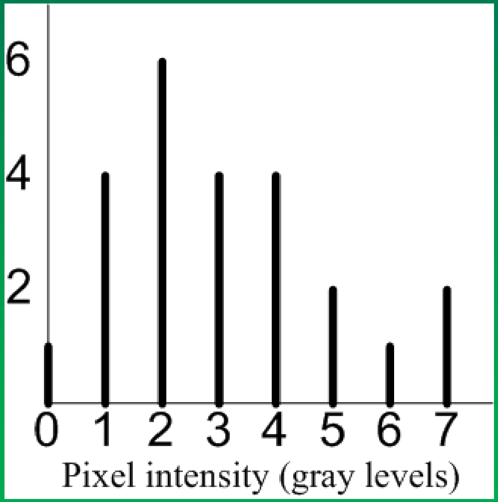
$$b(x,y) = \begin{cases} 1 & \text{if } g(x,y) \ge g_{th} \\ 0 & \text{if } g(x,y) < g_{th} \end{cases}$$

Histogram

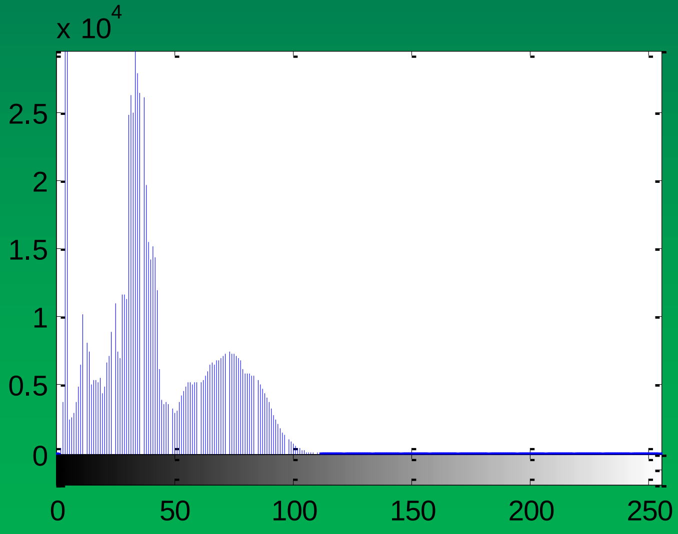
- A bar graph of the pixel intensities plotted along the x-axis and the number of occurrences for each intensity represents the y-axis.
- Provide information about the contrast and overall intensity distribution of an image.

3	4	2	5	1
2	1	6	4	2
3	5	7	1	3
4	2	5	7	1
2	5	0	3	2

Image

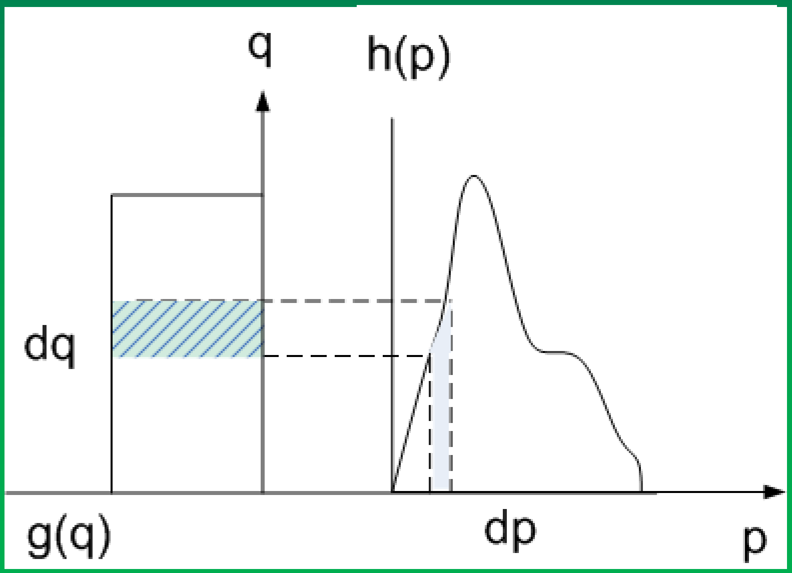


Histogram



Histogram Equalization

- The goal is to obtain a uniform histogram.
- Histogram equalization redistributes intensity distributions. If the histogram of any image has many peaks and valleys, it will still have peaks and valley after equalization, but peaks and valley will be shifted.



$$h(p)dp = g(q)dq$$

$$\text{Since } g(q) = \frac{M \times N}{2^n - 1},$$

$$q = \frac{2^n - 1}{M \times N} \int_0^p h(p)dp$$

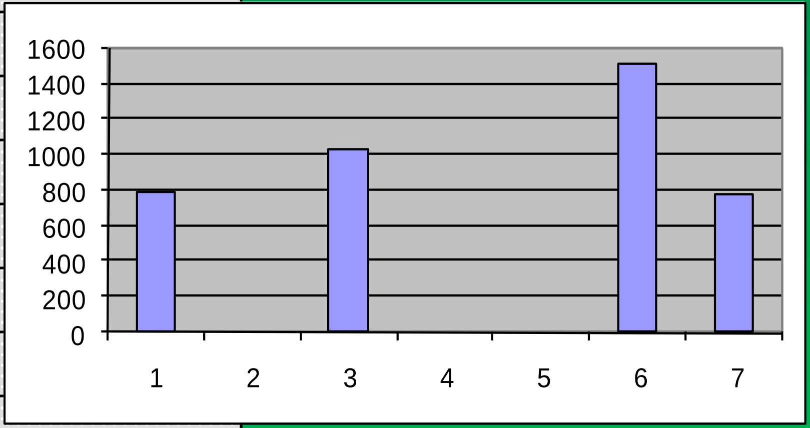
Histogram Equalization

Example: An 8-level 64x64 gray-scale image has the following histogram distribution:

Gray levels, r_k	Number of repetitions, n_k
0	790
1	1023
2	1506
3	574
4	203
5	0
6	0
7	0

$$q_k = \frac{2^n - 1}{M \times N} \sum_{p=0}^{k} h(p) = \frac{7}{64 \times 64} \sum_{p=0}^{k} h(p)$$

p	h(p)	C. hist.	q	R'd(q)	Eq. Hist.
0	790	790	1.4	1	790
1	1023	1813	3.1	3	1023
2	1506	3319	5.7	6	1506
3	574	3893	6.7	7	777
4	203	4096	7.0	7	
5	0	4096	7.0	7	
6	0	4096	7.0	7	
7	0	4096	7.0	7	

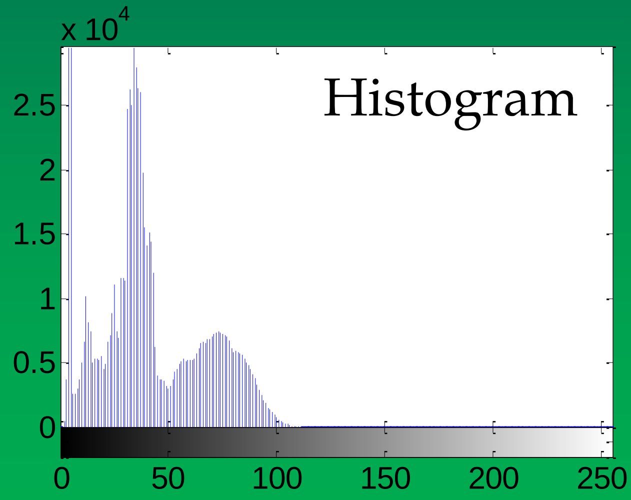


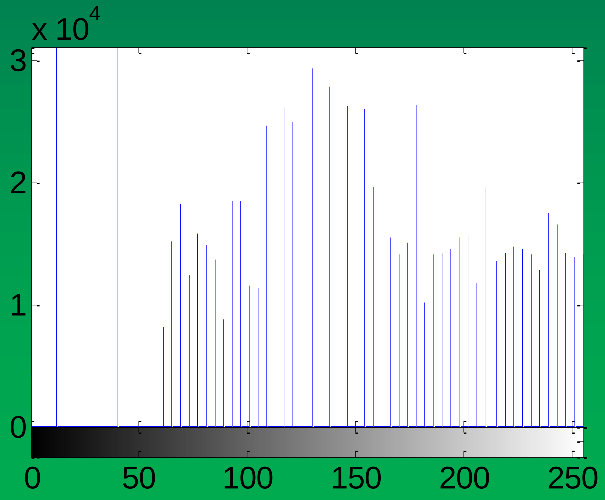
Histogram Equalization

OPERATION:
1. Compute histogram
2. Calculate cumulative histogram
3. Transform input image to output image.

Original

Histogram Equalized





Templates (masks) and Convolution

A sliding window, called the template (or mask), centers on each pixel in an input image and generates new output pixels computed by multiplying each pixel value in the neighborhood with the corresponding weight in the convolution mask and summing these products.

$$T * I(X,Y) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} T(i,j) I(X+i, Y+j)$$

Alternatively,

X_1	X_2	X_3
X_4	X_5	X_6
X_5	X_8	X_9

Image

w_1	w_2	w_3
w_4	w_5	w_6
w_5	w_8	w_9

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_9 \end{bmatrix}; \quad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_9 \end{bmatrix}$$

$$y(i,j) = \mathbf{w}^{\mathrm{T}}\mathbf{x} = \sum_{k=1}^{9} w_k x_k$$

Example masks:

Line mask

$$\begin{pmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{pmatrix} \quad \begin{pmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Averaging or smoothing

$$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Point mask

$$\begin{pmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{pmatrix} \quad \begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix}$$

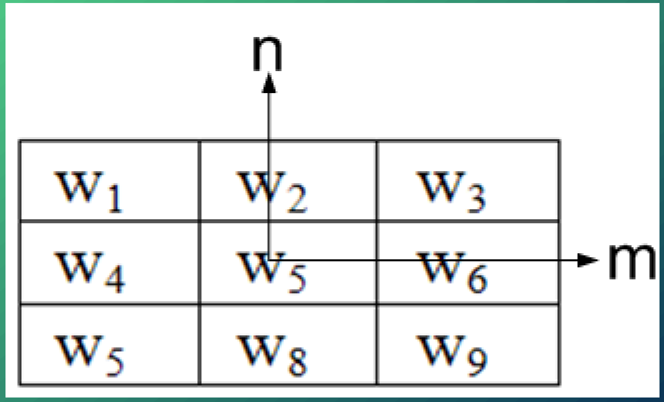
Example: Derive a 5x5 Gaussian filter with σ equal to 1 pixel.

$$G_\sigma(m,n) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left[\frac{-(m^2+n^2)}{2\sigma^2}\right]$$

$$G_\sigma(0,0) = \frac{1}{\sqrt{2\pi\sigma}}$$

$$G_\sigma(\pm 1,0) = G_\sigma(0,\pm 1) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left[\frac{-1}{2\sigma^2}\right]$$

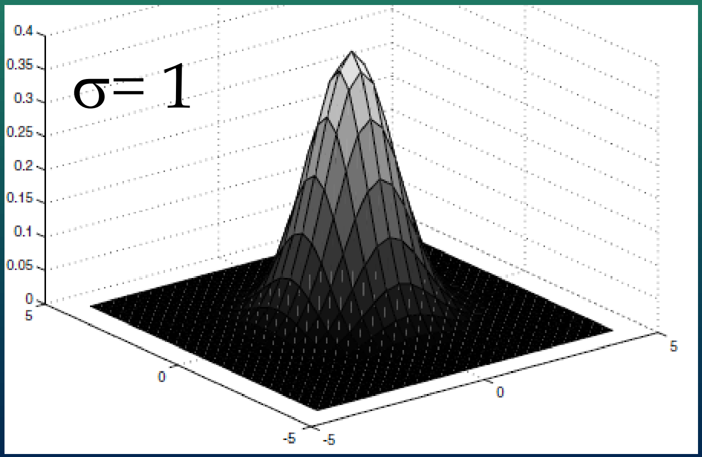
$$G_\sigma(\pm 1,\pm 1) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left[\frac{-1}{\sigma^2}\right]$$

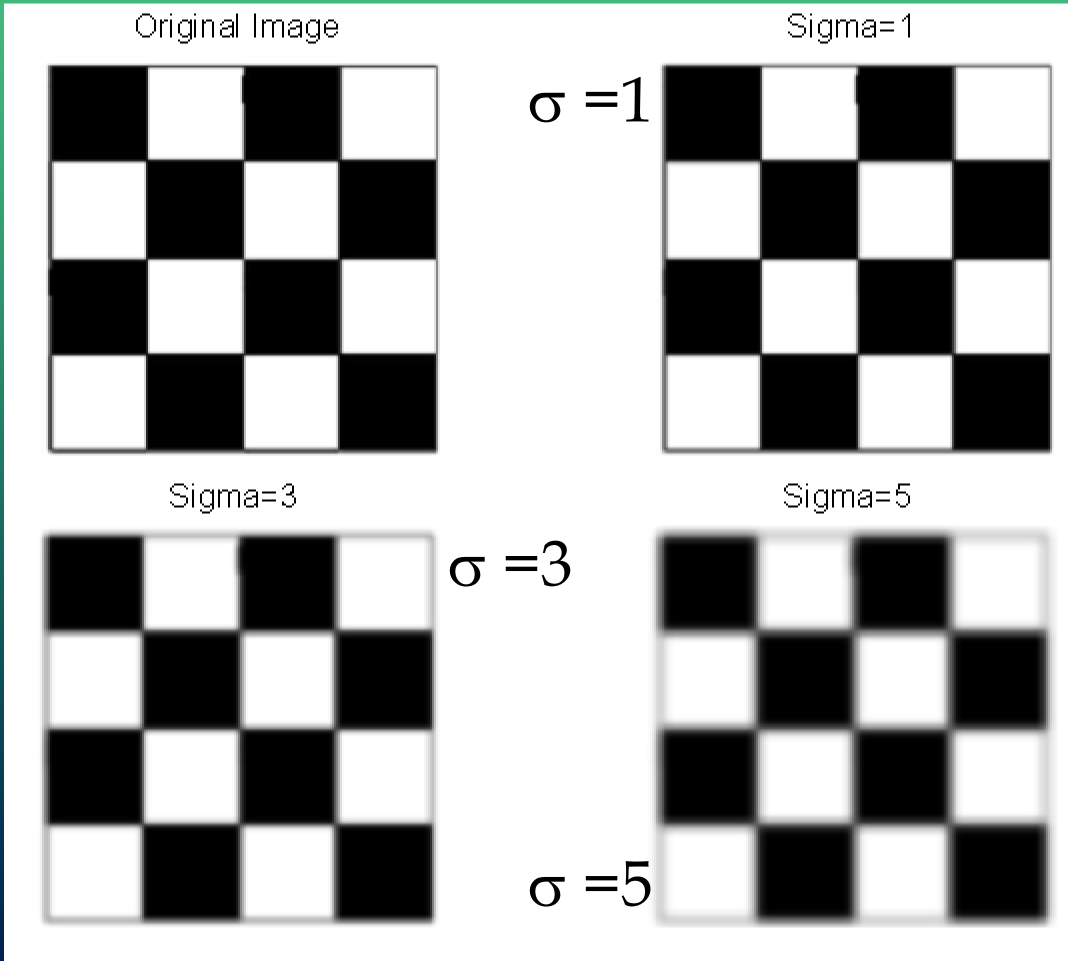


Example: 5x5 Gaussian filter

$$G_\sigma(m,n) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left[\frac{-(m^2+n^2)}{2\sigma^2} \right]$$

0.0029	0.0131	0.0215	0.0131	0.0029
0.0131	0.0585	0.0965	0.0585	0.0131
0.0215	0.0965	0.1592	0.0965	0.0215
0.0131	0.0585	0.0965	0.0585	0.0131
0.0029	0.0131	0.0215	0.0131	0.0029





CCD's are prone to many types of disturbing factors:

- ***Dynamic range.*** The maximum number of electrons that can be held by a storage element is limited to (typically) $4x10^5$. Therefore, above a certain illumination level the storage elements are saturated.
- ***Blooming.*** When part of the image plane is illuminated above the saturation level, the area of saturation increases due to diffusion of the excess-charge.
- ***Smear.*** When the charges are shifted vertically the integration of radiant flux continues. Therefore a bright spot makes a vertical stripe on the image. This effect is called smear.
- ***Dark current noise***. Each storage element may have a random number of electrons due the thermal fluctuations (recombination and generations of electron-hole pairs). Typically the standard deviation of these fluctuations is about 50 electrons.
- ***Fixed-pattern noise***. Impurities in the crystals of the CCD may produce electron-hole pairs. The standard deviation of the number of produced electrons varies from storage element of storage element. But, typically, this deviation is about 700.
- ***Reset noise.*** When resetting the charge in a capacitor there will be a fluctuating number of electrons that remains due to thermal activities. The standard deviation of this number is inversely with the square root of the capacitance. Resetting occurs at each clock cycle at the input of the output-circuitry. Typically the standard deviation is about 100 electrons.
- ***Thermal noise in the output amplifier.*** Even an optimized amplifier design cannot prevent a thermal noise contribution. This noise level lies at around 100 electrons.
- ***Quantum noise.*** The number of electrons in a charge packet obeys the Poisson distribution. This implies that the standard deviation due to the discrete nature of the charge is proportional to the square root of the mean. Consequently, without illumination this standard deviation is almost zero, but at saturation level, the standard deviation becomes about 700 electrons.

Concept of edge:

First order derivative for edge detection

g_m = x_difference(m,n)
= value(m, n) – value(m+1, n)

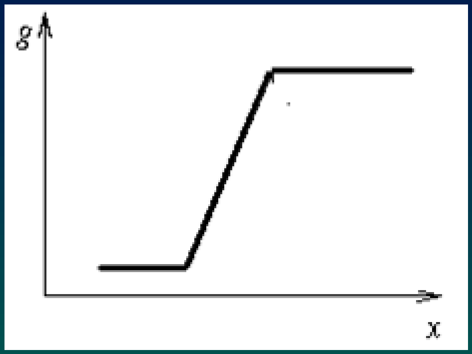
g_n = y_difference(m,n)
= value(m, n) – value(m, n+1)

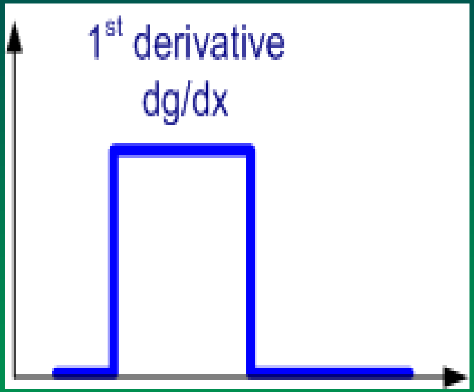
Magnitude:

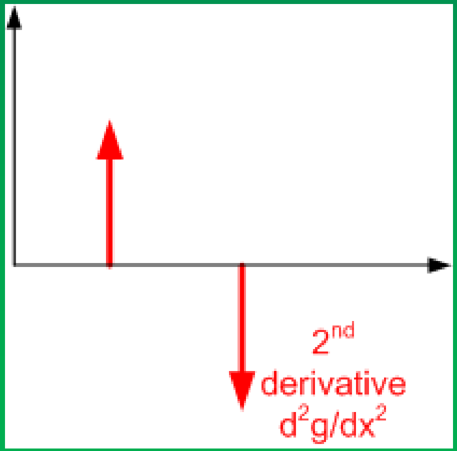
$$\left|g(m,n)\right| = \sqrt{g_m^2(m,n) + g_n^2(m,n)}$$

Direction:

$$\angle g(m,n) = \tan^{-1}\frac{g_n(m,n)}{g_m(m,n)}$$







Edge detection mask

Sobel 3 x 3 masks (most commonly used)

$$H_x = \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad H_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Magnitude:

$$|g(m,n)| = \sqrt{g_m^2(m,n) + g_n^2(m,n)}$$

Direction (measured from x axis):

$$\angle g(m,n) = \tan^{-1} \frac{g_n(m,n)}{g_m(m,n)}$$

Example image:

y

x

3	4	2	5	1
2	1	6	4	2
3	5	7	1	3
4	2	5	7	1
2	5	1	3	2

Calculate both *magnitude* and *direction* of the gradient using Sobel operator at the pixel p(2,2).

Example:

Calculate both *magnitude* and *direction* of the gradient using Sobel operator at the pixel p(2,2).

3	4	2	5	1
2	1	6	4	2
3	5	7	1	3
4	2	5	7	1
2	5	1	3	2

$$H_x = \begin{bmatrix} w_{x1} & w_{x2} & w_{x3} \\ w_{x4} & w_{x5} & w_{x6} \\ w_{x7} & w_{x8} & w_{x9} \end{bmatrix} \quad H_x = \begin{bmatrix} w_{x1} & w_{x2} & w_{x3} \\ w_{x4} & w_{x5} & w_{x6} \\ w_{x7} & w_{x8} & w_{x9} \end{bmatrix} \quad Z = \begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix}$$

$$g_y(m,n) = \sum_{i=1}^{9} w_{ni} z_i$$

$$g_x(m,n) = \sum_{i=1}^{9} w_{mi} z_i$$

Magnitude:

$$|g(m,n)| = \sqrt{g_m^2(m,n) + g_n^2(m,n)}$$

Direction:

$$\angle g(m,n) = \tan^{-1} \frac{g_n(m,n)}{g_m(m,n)}$$

Example (cont.)

$$H_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad H_x = \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$Z = \begin{bmatrix} 3 & 4 & 2 \\ 2 & 1 & 6 \\ 3 & 5 & 7 \end{bmatrix}$$

$$g_m = 3 + 2(5) + 7 - 3 - 2(4) - 2 = 7$$

$$g_n = 2 + 2(6) + 7 - 3 - 2(2) - 3 = 11$$

Magnitude:

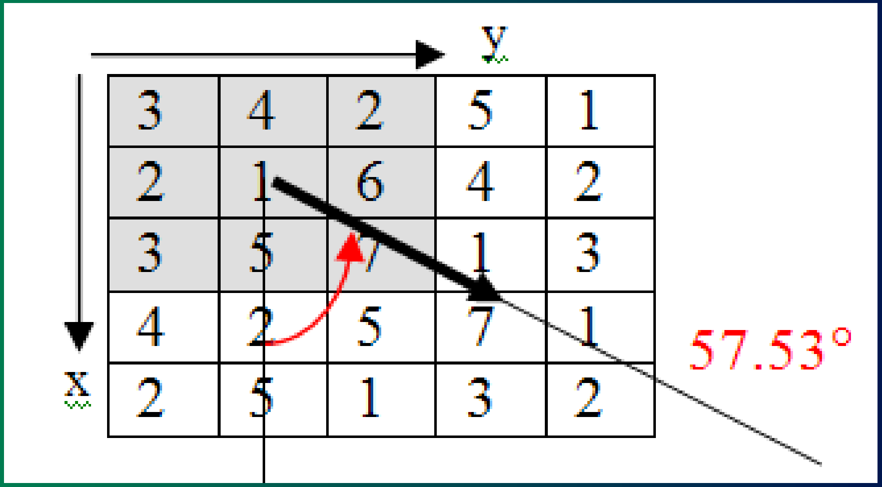
$$|g(m,n)| = \sqrt{g_m^2 + g_n^2} = 13$$

Direction:

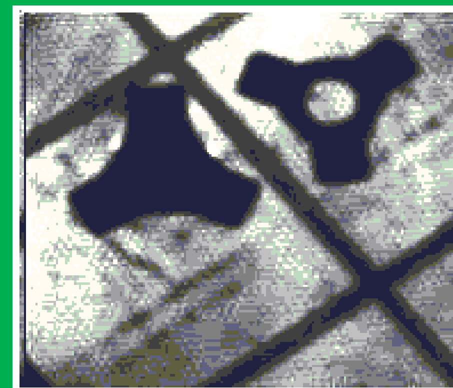
$$\angle g(m,n) = \tan^{-1}\frac{g_n}{g_m} = 57.53°$$

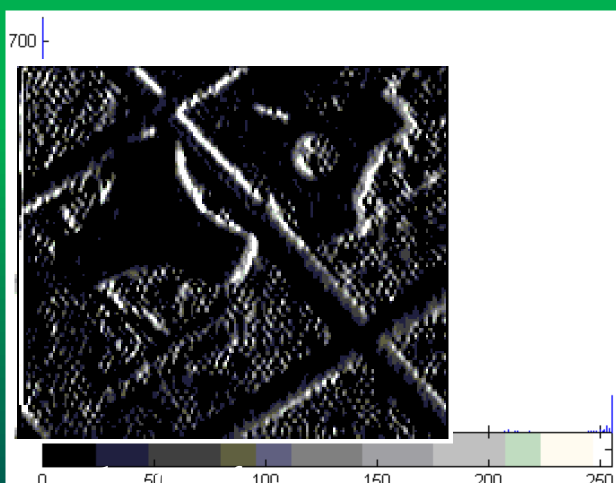
Notes:

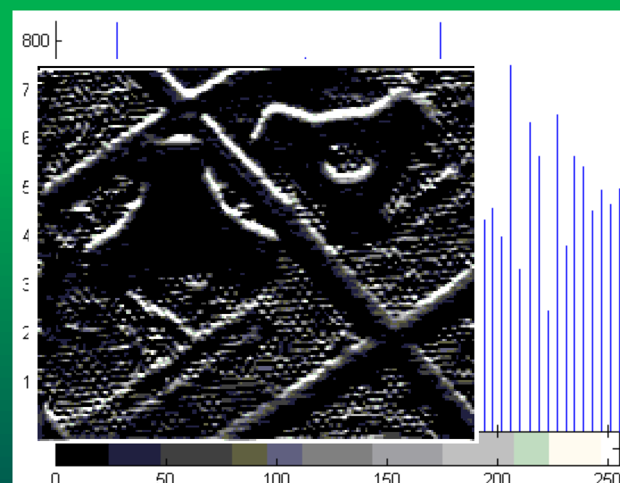
- The direction is measure from the positive x-axis toward the positive y-axis.
- It points towards the brighter region i.e. between 6 and 7 on the lower right of the P(2,2).

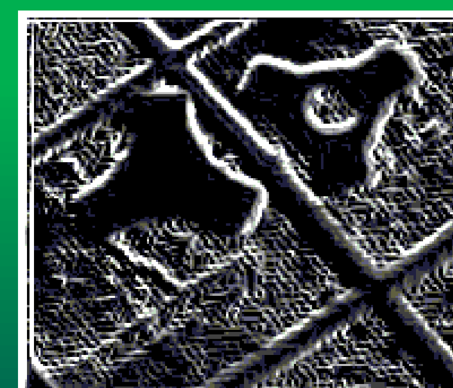


Edge detection example (Sobel)









Plot of g_m

Original Image

Plot of g_n

Histogram Equalization

Plot of g

Other 1st order edge detection masks

Roberts operator

$$H_x = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad H_y = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

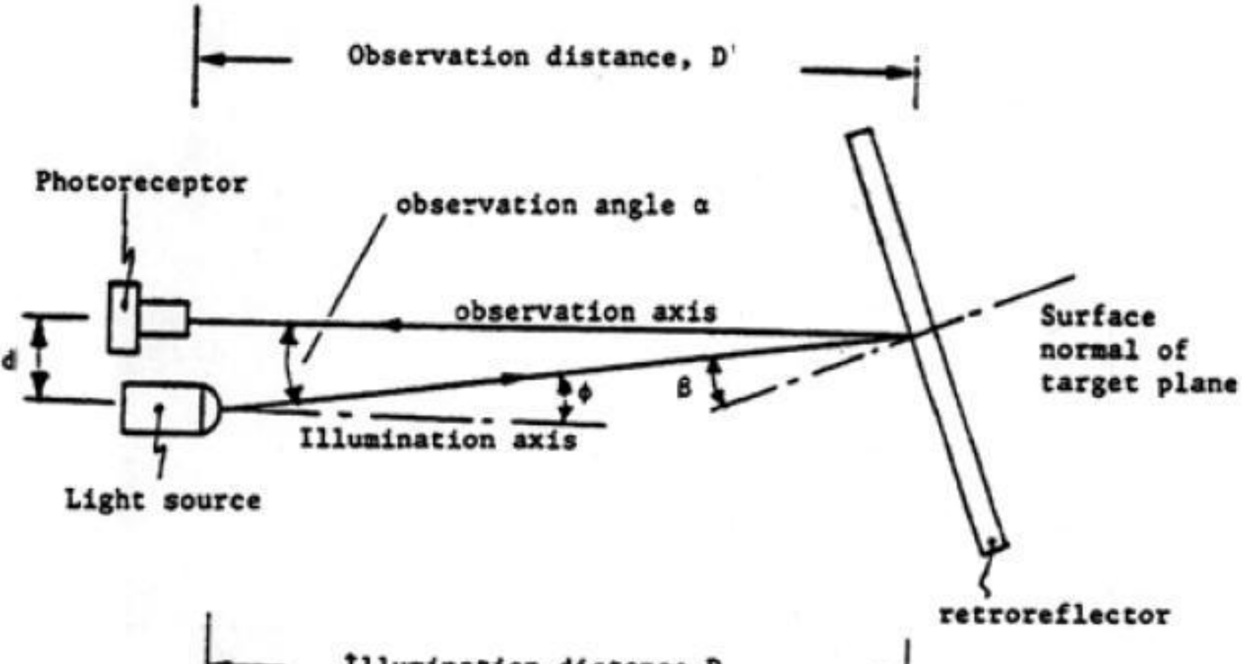
Prewit operator

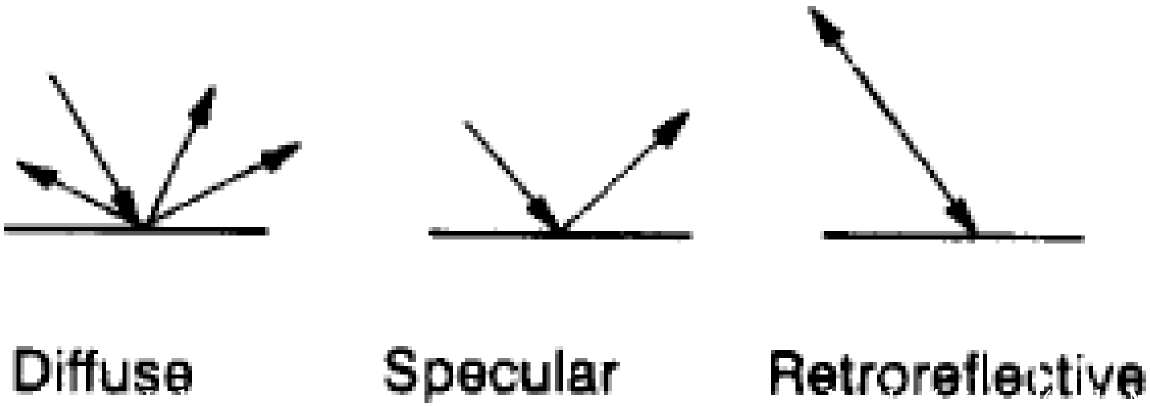
$$H_x = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad H_y = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

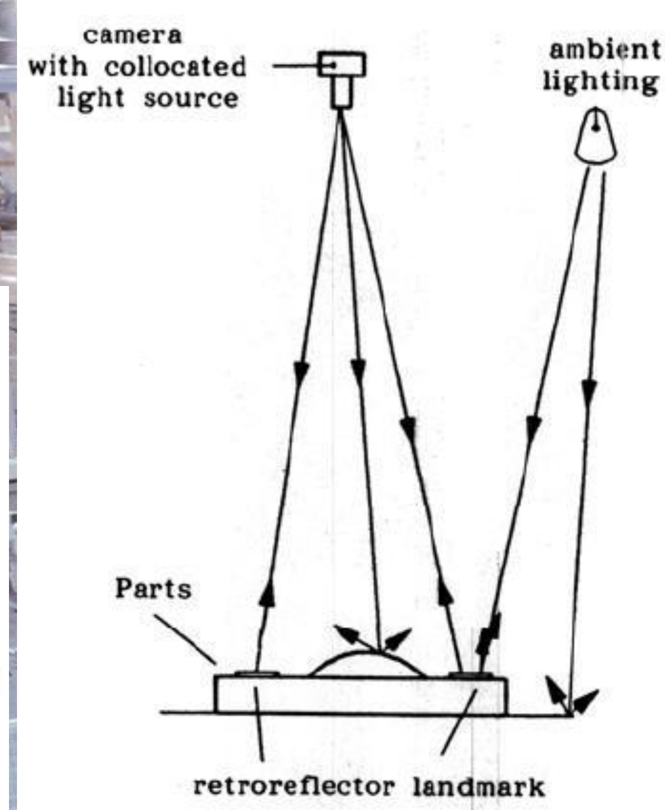
Frei-Chen operator

$$H_x = \begin{bmatrix} 0 & 0 & -1 \\ \sqrt{2} & 0 & \sqrt{2} \\ 0 & 0 & -1 \end{bmatrix} \quad H_y = \begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$$

Industrial example

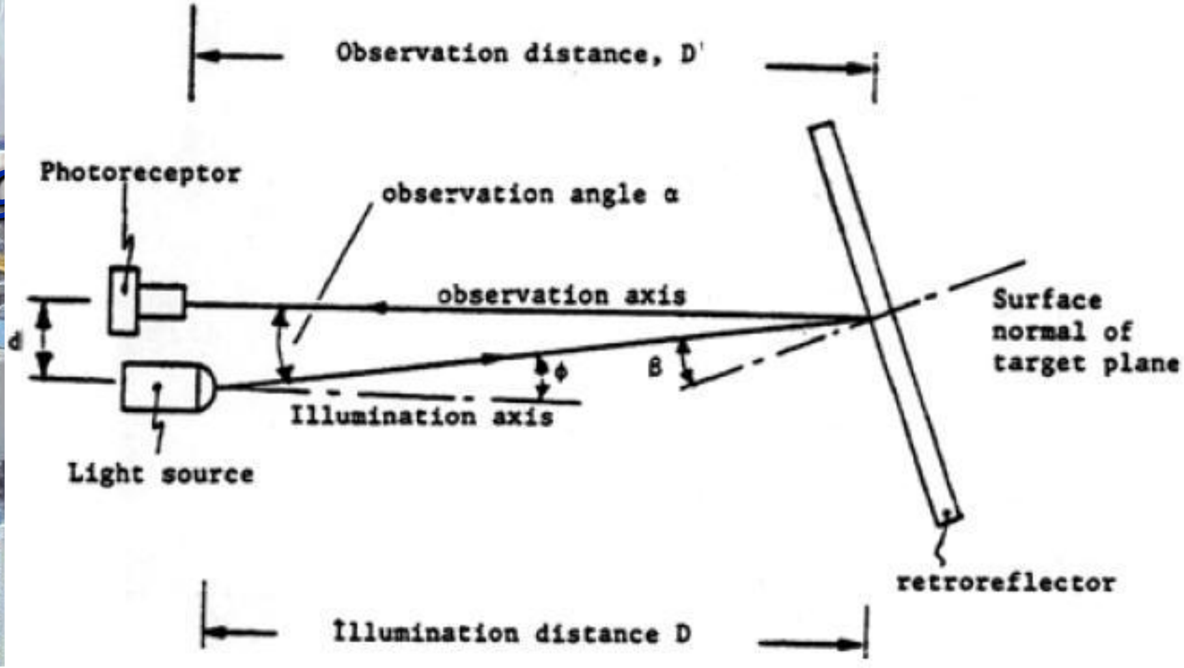


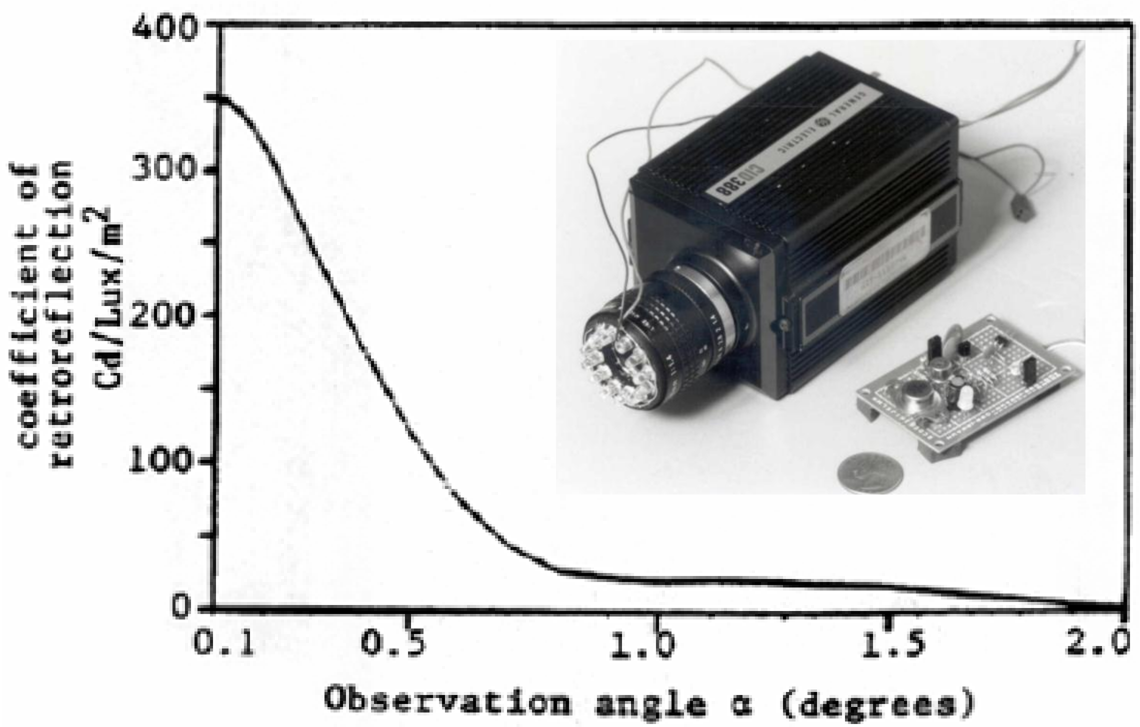




An example

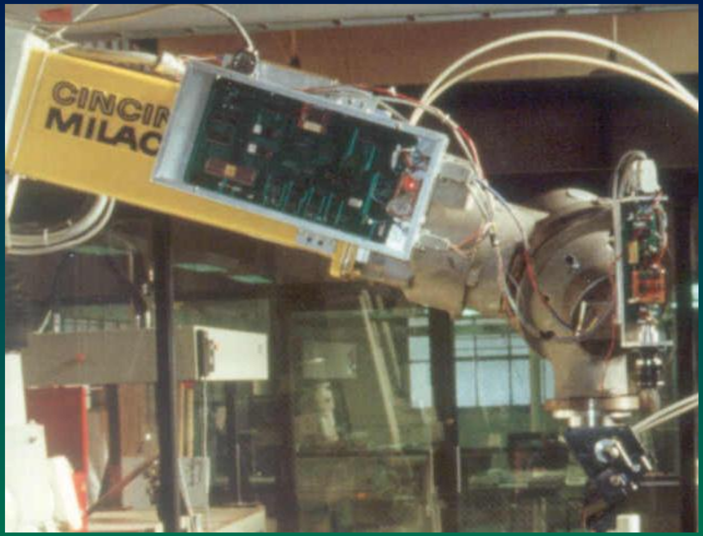
Retroreflective sensin

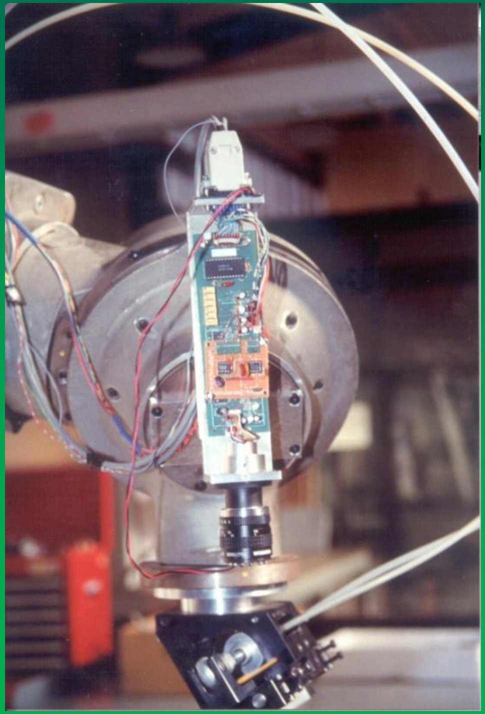


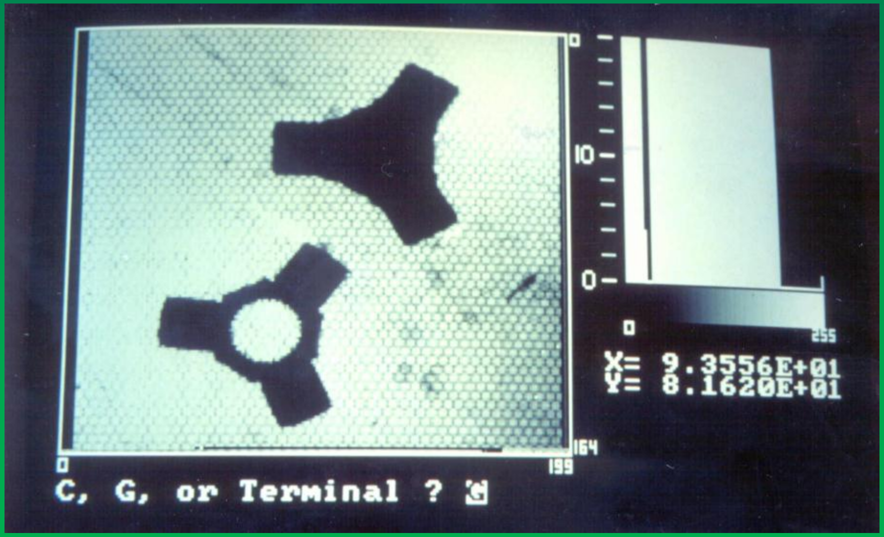


An example

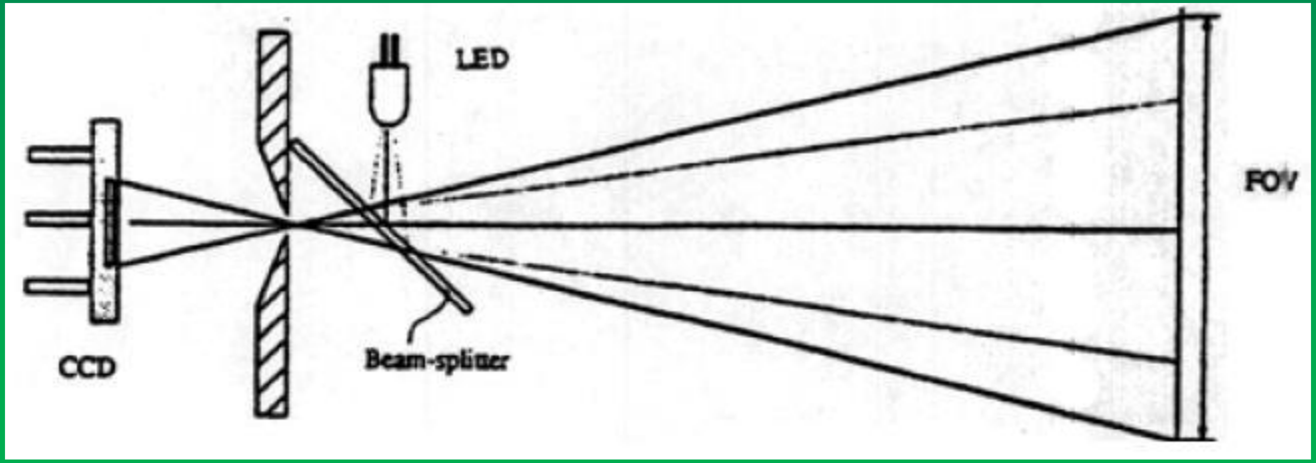
Retroreflective sensing

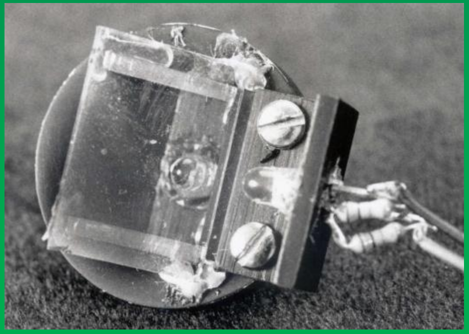






An example
Retroreflective sensing

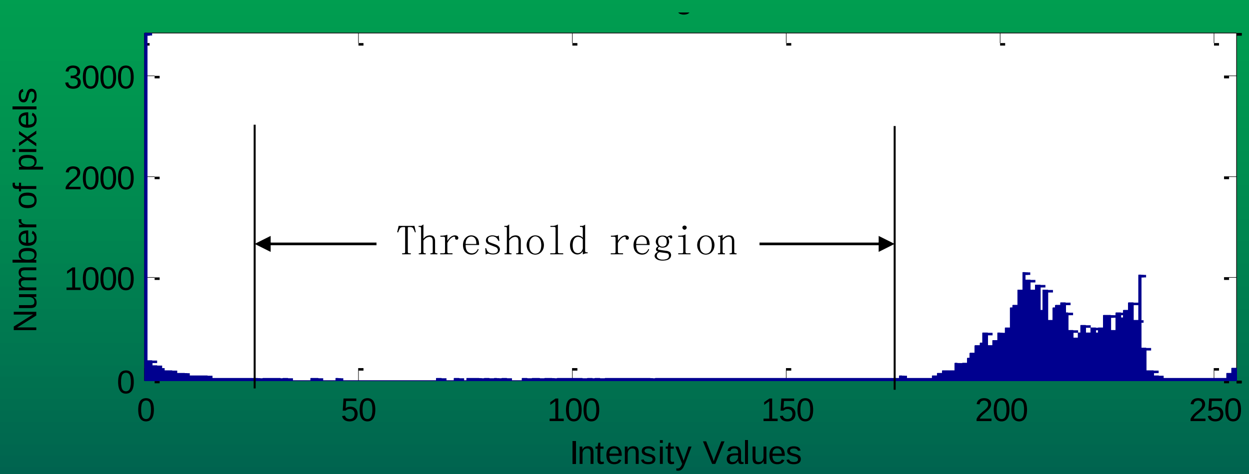


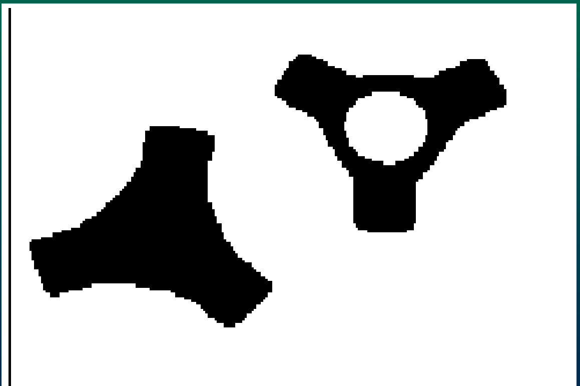


Binary image

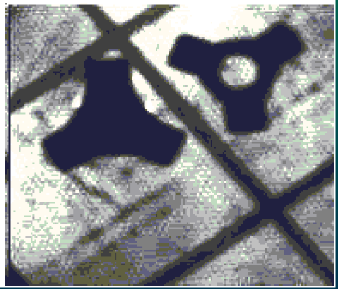
$$b(x,y)=\begin{cases}1 & \text{if } g(x,y)\ge g_{th}\\ 0 & \text{if } g(x,y)< g_{th}\end{cases}$$

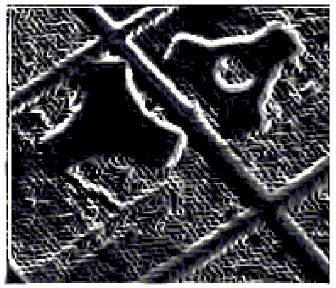

retroreflective





g_{th} = threshold = 57


Histogram Equalized


Gradient of image


Gradient of binary image

MATLAB IMAGE PROCESSING TOOLBOX

A collection of functions built on MatLab's computing environment. The functions support

- Imported and export images,
- image display,
- geometric operations,
- neighborhood and block operations,
- filtering and filter design,
- analysis and enhancement,
- binary image operations,
- region of interest operations, and
- image transform

Image Types in Matlab Toolbox

Matlab supports four basic types of images:

- Indexed images (typical color images)
- Intensity images (gray-scale images)
- Binary images (black and white)
- RGB images (standard way of representing color data)

Displaying Images

Image type	**Command**
Indexed images	imshow(X,map)
Intensity images	imshow(I, 64)
Binary images	imshow(BW, 2); 0(black), 1(white)
	imshow(~BW, 2); 1(black), 0(white)
RGB images	imshow(R, G, B)

Indexed Images

Typical color images require two matrices,:

- a colormap and
- an image matrix

The *colormap* is an ordered set of values that represent the colors in the image.

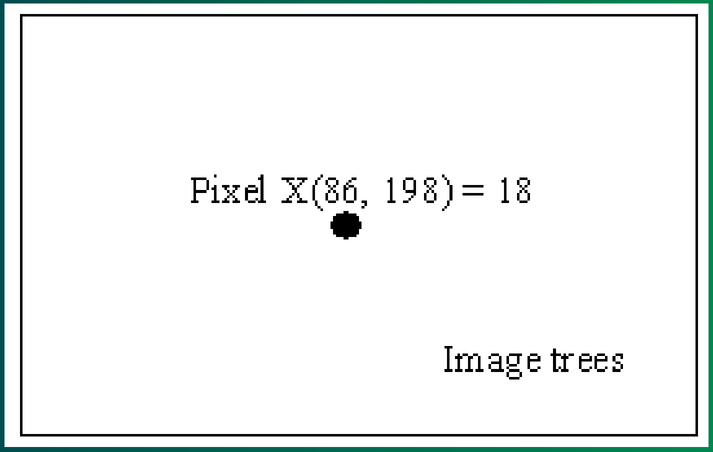
- The size of the colormap is nx3 for an image containing n colors.
- Each row of the colormap matrix is a 1-by-3 red, green, blue (RGB) color vector.

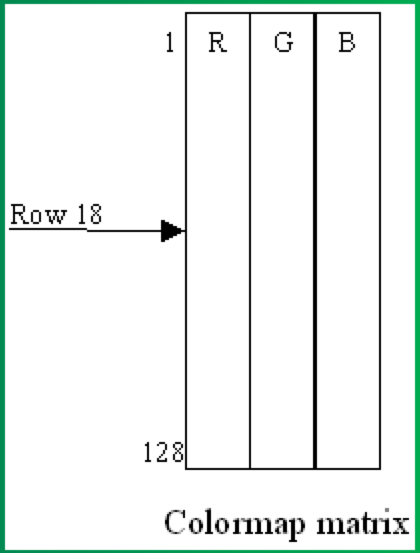
For each image pixel, the image matrix contains a corresponding index in the colormap. When Matlab displays an indexed image, it uses the values in the image matrix to look up the desired color in the colormap.

Indexed Image Example 1

load trees % **load** the image data file trees.mat (mat-general matrix extension).
imshow (X, map) % **imshow** command displays the image on the screen.

X(86, 198)

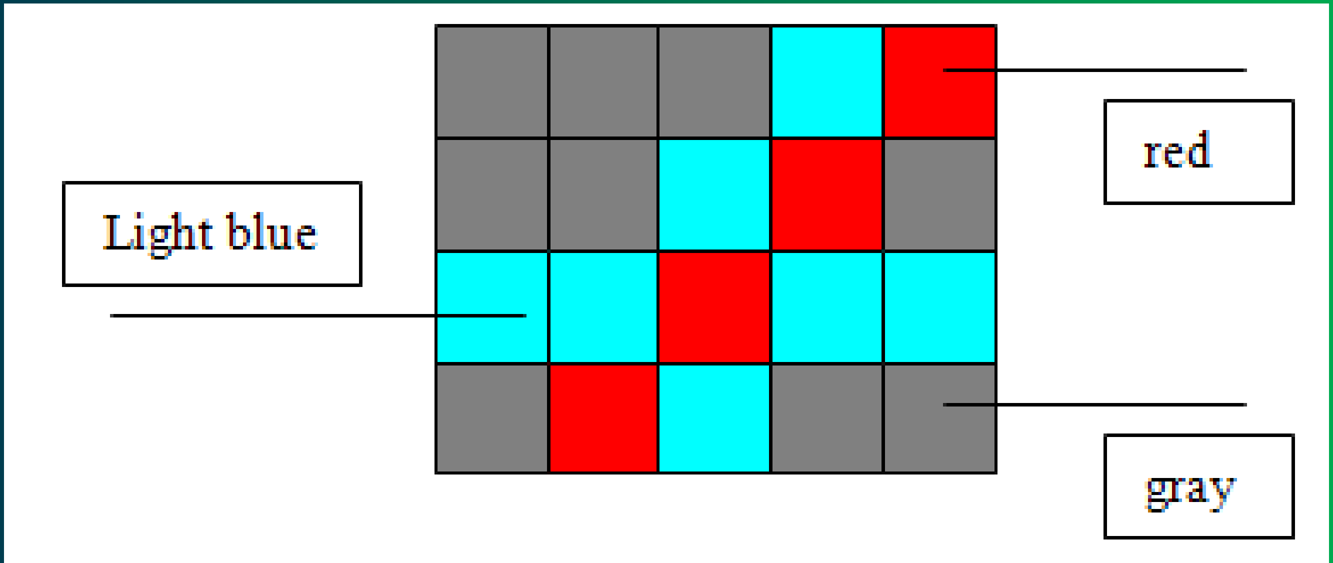




Intensity Image

```
X = [1 1 2 1 3
     1 1 2 3 1
     2 2 3 2 2
     1 3 2 1 1 ];
map = [.4  .4  .4;  0 .6  1;  1 0 0]
```

	R	G	B
1	0.4	0.4	0.4
2	0	0.6	1
3	1	0	0



Intensity Images

Store an intensity image as a single matrix, which contains double precision values ranging from 0.0 to 1.0, with each element of the matrix corresponding to an image pixel.

```
load trees
I =  ind2gray(X, map); % convert indexed
                         image to gray scale.
imshow (I, 64)
```

To convert an index image to gray scale,
ind2gray (X+1, gray(256)) % zero-based colormap.
ind2gray (X, gray(256)) % colormap beginning with index 1
ind2gray (X+1, map) or ind2gray (X, ,map) % if image comes with its own colormap, say map

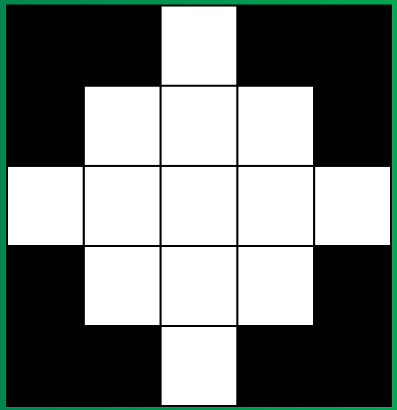
Binary Images

A binary (black and white) image is a special kind of intensity image. Binary contain only two levels, 0(black) or 1(white).

```
load trees
I =  ind2gray(X, map);
BW = edge(I);           % command edge detects change in intensity
imshow(~BW, 2)          % ~ invert black to white and vice verse.
```

The following command creates a 5x5 bnary image:

```
BW = [0 0 1 0 0
      0 1 1 1 0
      1 1 1 1 1
      0 1 1 1 0
      0 0 1 0 0 ];
Imshow (BW, 2)
```



RGB Images

In Matlab, the red, green, and blue components of an RGB image reside in three separate intensity matrices, each having the same row and column dimensions as the original RGB image.

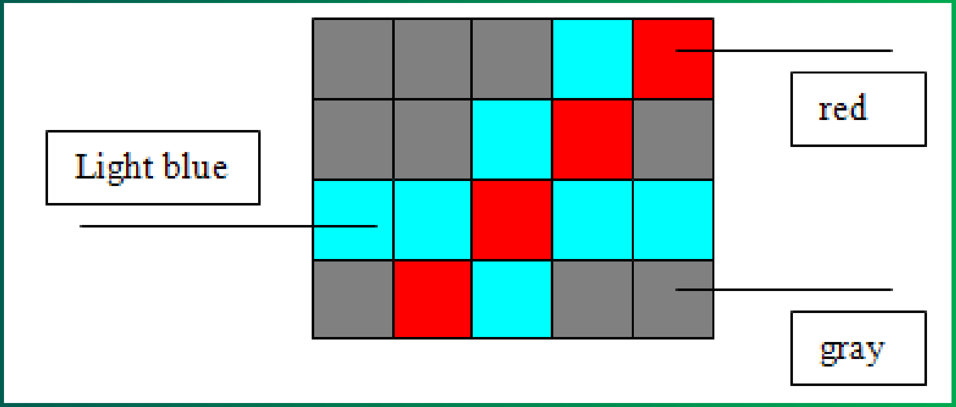
The intensities of corresponding pixels from each matrix combine to create the actual pixel color at a given location.

The following statements create an RGB image on the right:

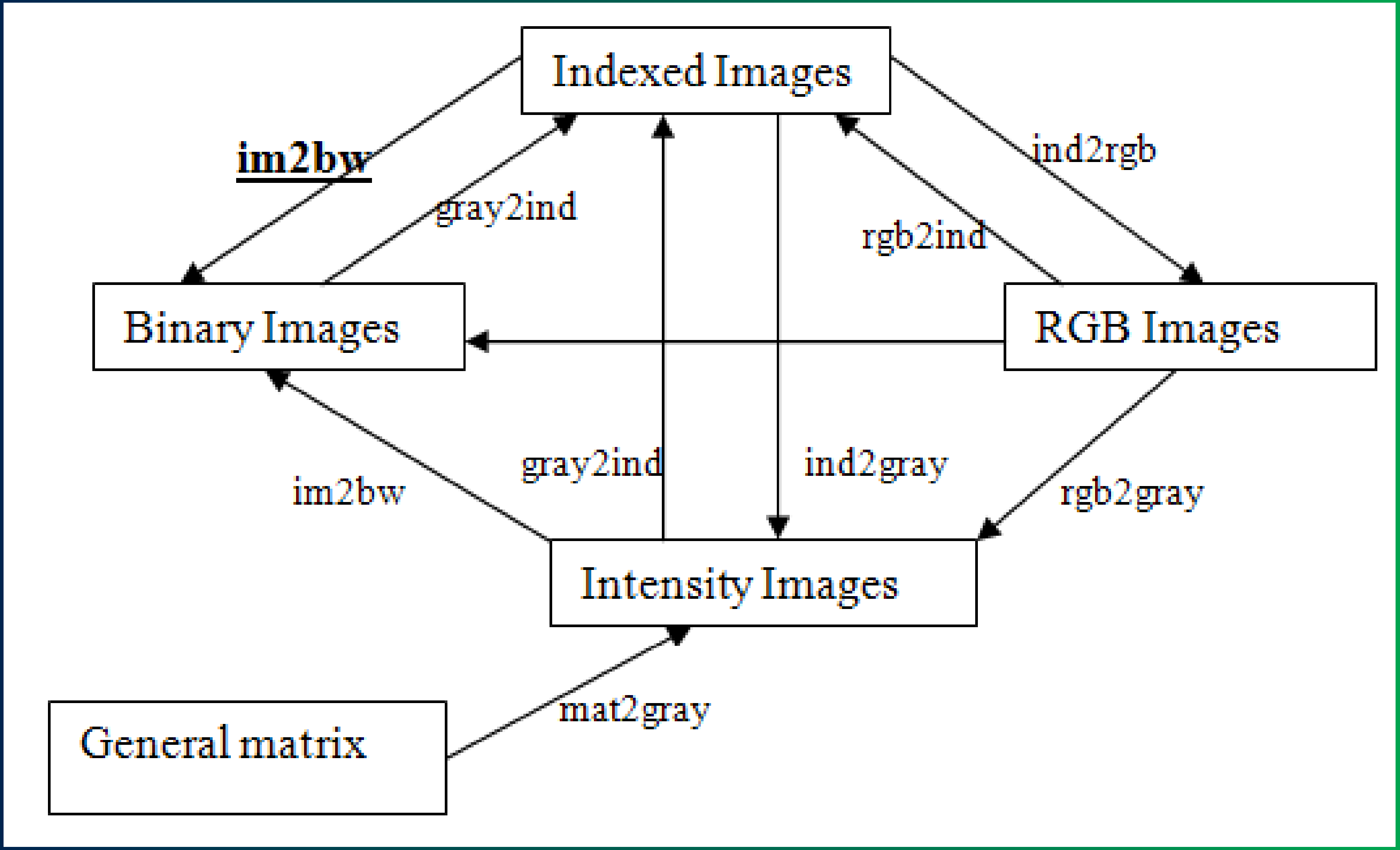
```
R = [ .4 .4 0 .4 1
      .4 .4 0 1 .4
      0 0 1 0 0
      .4 1 0 .4 .4 ];

G = [ .4 .4 .6 .4 0
      .4 .4 .6 0 .4
      .6 .6 0 .6 .6
      .4 0 .6 .4 .4 ];

B = [ .4 .4 1 .4 0
      .4 .4 1 0 .4
      1 1 0 1 1
      .4 0 1 .4 .4 ];
```



Changing Image Types



Importing and Exporting Images

MATLAB *imread* function can read these graphics file.

To write image data from MATLAB to a file, use the *imwrite* function.

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`

† `GIF` is supported by `imread`, but not by `imwrite`.

<u>Example 3</u>

```
I=imread('trees.tif');        % import the image saturn.tif
imshow(I)                     %display a grayscale intensity image, default [0 255]
imwrite(I, 'test.jpg');       % save the image into a new file test.jpg
```

Other examples

Example 4

[X,map]=imread('trees.tif'); % read the trees in tiff format to the memory (matlab workspace) as index image

imshow(X, map); % once the image is on the MATLAB workspace, it can be shown on screen.

Example 5

Y = imresize(X, 2); % double the number of pixels in image X.

Y = imresize (X, [100 150]); % resize the image to 100-by-150.

Example 6

I= imread ('ic.tif');

J = imrotate(I, 35, 'bilinear'); % rotate then 35 degrees using bilinea interpolation.

Imshow(I)

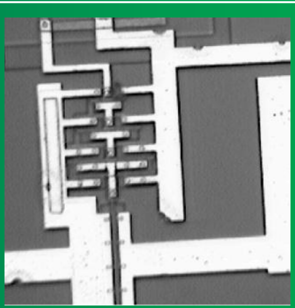
figure, imshow(J); % create new figure

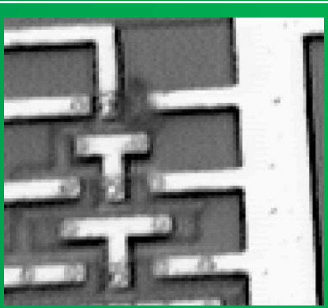
Example 7

I = imread('ic.tif');

I2 = imcrop(I,[60 40 100 90]);

imshow(I), figure, imshow(I2)





Other examples

Example: help bwlabel

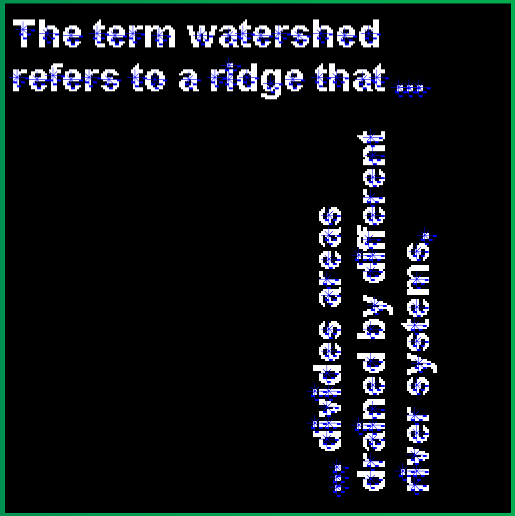
```
BW = logical([1 1 1 0 0 0 0 0
              1 1 1 0 1 1 0 0
              1 1 1 0 1 1 0 0
              1 1 1 0 0 0 1 0
              1 1 1 0 0 0 1 0
              1 1 1 0 0 0 1 0
              1 1 1 0 0 1 1 0
              1 1 1 0 0 0 0 0]);
[L, num] = bwlabel(BW,4)
```

```
L =
1   1   1   0   0   0   0   0
1   1   1   0   2   2   0   0
1   1   1   0   2   2   0   0
1   1   1   0   0   0   3   0
1   1   1   0   0   0   3   0
1   1   1   0   0   0   3   0
1   1   1   0   0   3   3   0
1   1   1   0   0   0   0   0
num = 3
```

Example: help regionprops

Label the connected pixel components in the text.png image, compute their centroids, and superimpose the centroid locations on the image.

```
bw = imread('text.png');
L = bwlabel(bw);
s  = regionprops(L, 'centroid');
centroids = cat(1, s.Centroid);
imshow(bw)
hold on
plot(centroids(:,1), centroids(:,2), 'b*')
hold off
```

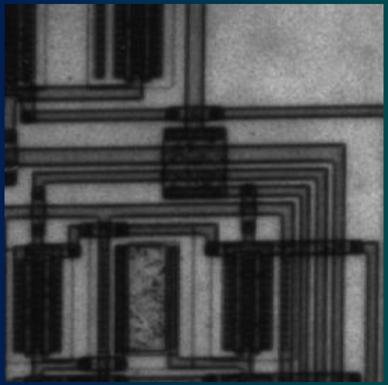


Other examples

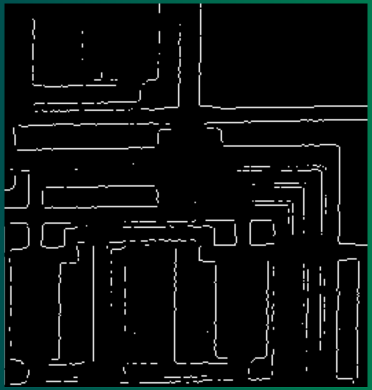
Example help edge

Find the edges of the circuit.tif image using the Prewitt and Canny methods:

```
I = imread('circuit.tif');
    BW1 = edge(I,'prewitt');
    BW2 = edge(I,'canny');
    figure, imshow(BW1)
    figure, imshow(BW2)
```



Image



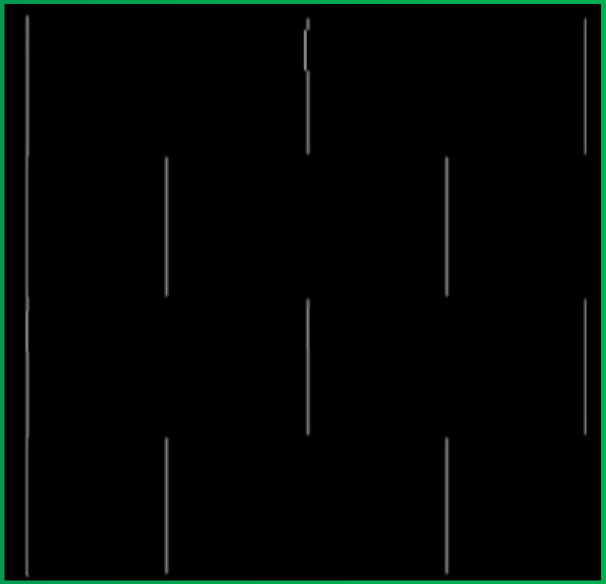
prewitt

Canny

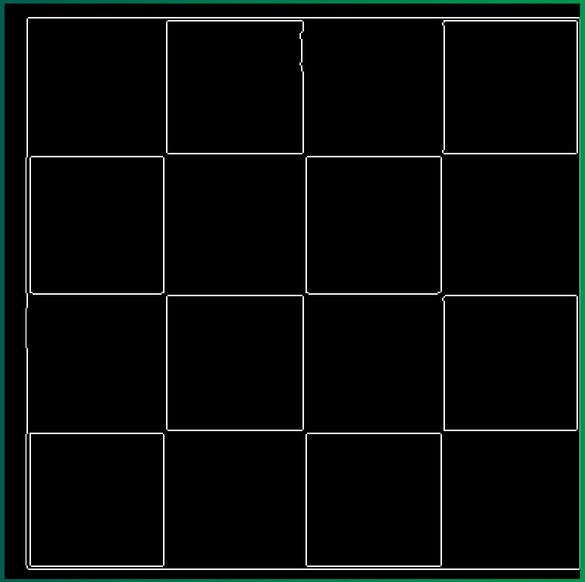
Other examples

[BW,thresh,gv,gh] = edge(I,'sobel',...) returns vertical and horizontal edge responses to Sobel gradient operators.

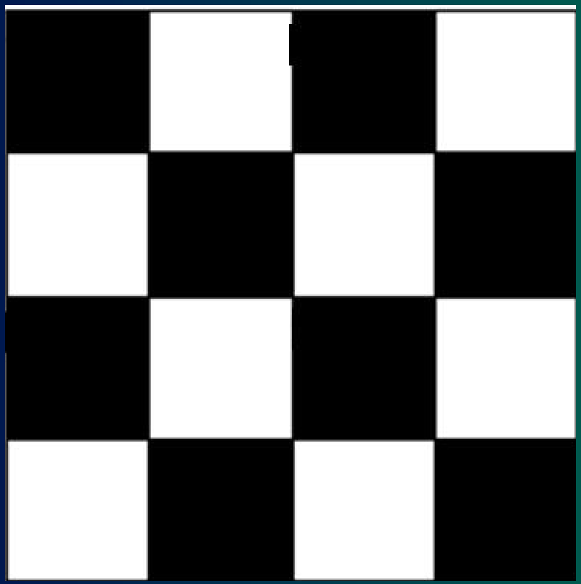
```
I=imread('checker.jpg');
 J=rgb2gray(I);
[BW,thresh,gv,gh]= EDGE(J,'sobel');
imshow(gv)
imshow(gh)
imshow(BW)
```



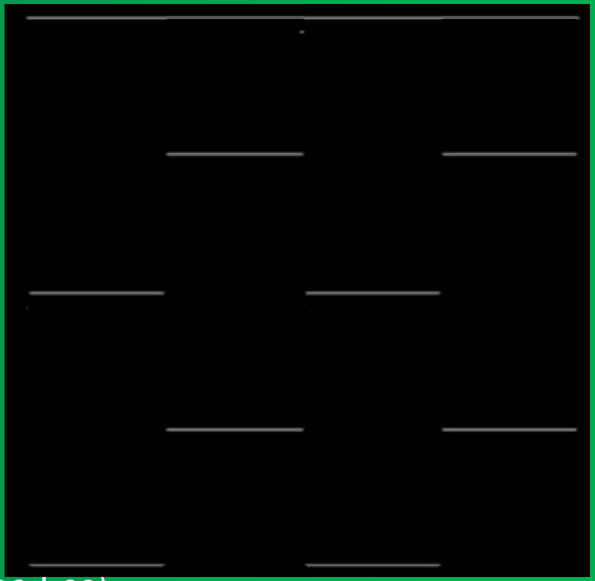
g_v



BW (Sobel edge)



Image



g_h