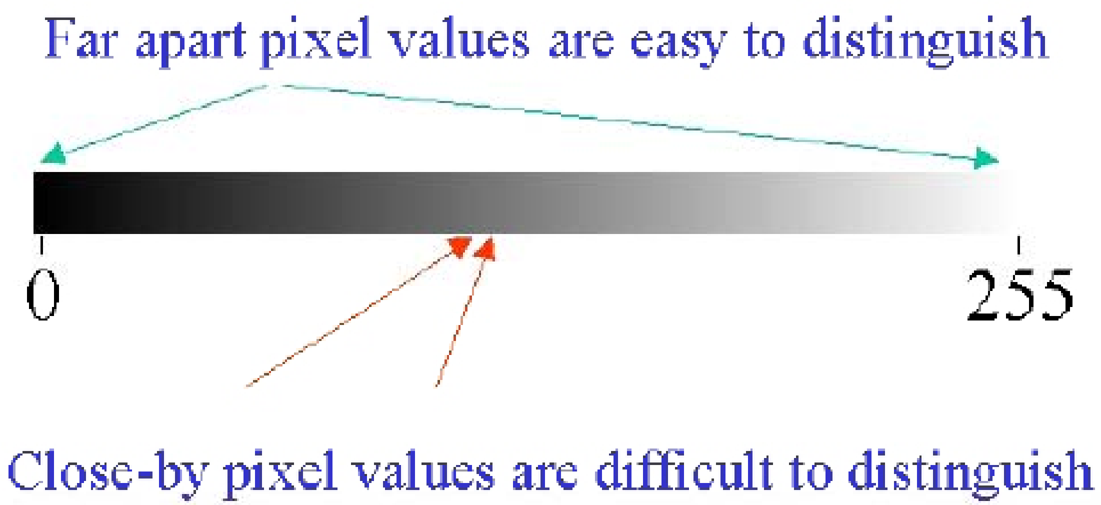
Digital Image Processing

Dr. Vrijendra Singh
IIIT Allahabad

Summary of Lecture 2

- Simple structural processing techniques like transposing, flipping and cropping.
- Simple image statistics like sample mean and sample variance.
- Histograms.
  - $h_A(l)$: number of pixels in image $\mathbf{A}$ that have the value l.
  - Histograms tell us how the values of individual pixels in an image are "distributed".
  - Two different images may have the same histogram.
- Point processing techniques.
  - $B(i,j) = g(A(i,j))$

Dynamic Range, Visibility and Contrast Enhancement


Far apart pixel values are easy to distinguish
0
255
Close-by pixel values are difficult to distinguish

- Contrast enhancing point functions we have discussed earlier expand the dynamic range occupied by certain "interesting" pixel values in the input image.
- These pixel values in the input image may be difficult to distinguish and the goal of contrast enhancement is to make them "more visible" in the output image.
- Don't forget we have a limited dynamic range $(0-255)$ at our disposal.

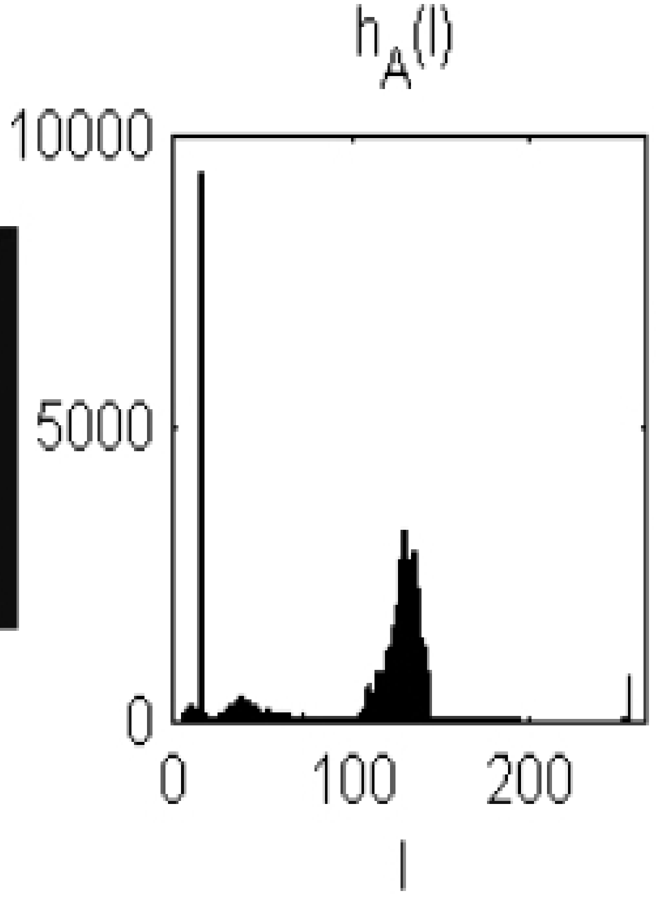
Image Segmentation

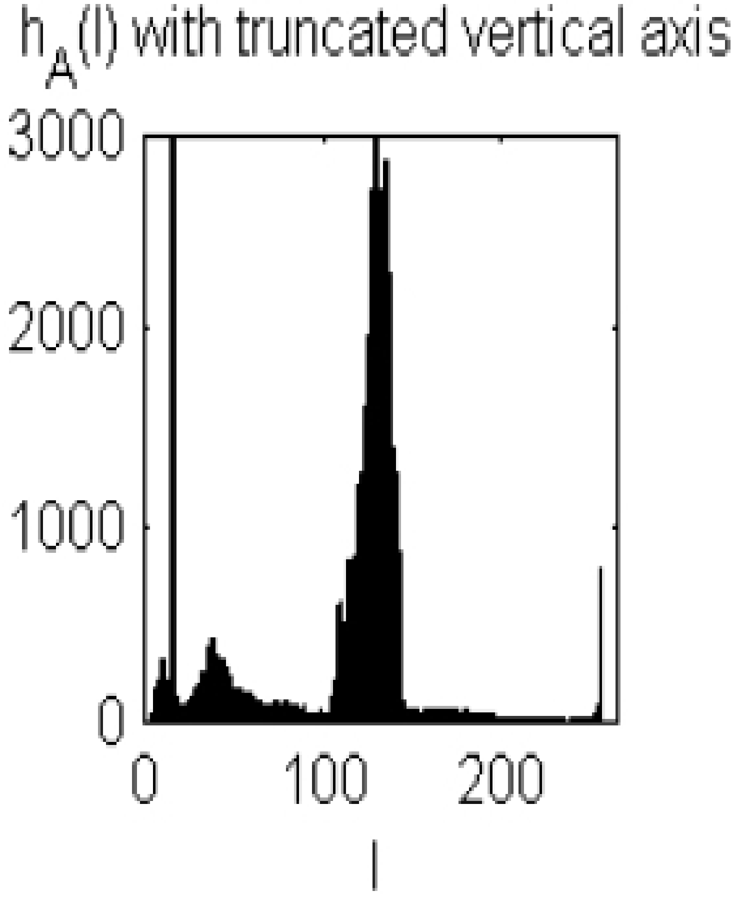
- If one views an image as depicting a scene composed of different objects, regions, etc. then segmentation is the decomposition of an image into these objects and regions by associating or "labelling" each pixel with the object that it corresponds to.
- Most humans can easily segment an image.
- Computer automated segmentation is a difficult problem, requiring sophisticated algorithms that work in tandem.
- "High level" segmentation, such as segmenting humans, cars etc., from an image is a very difficult problem. It is still considered unsolved and is actively researched.
- Based on point processing, histogram based image segmentation is a very simple algorithm that is sometimes utilized as an initial guess at the "true" segmentation of an image.

Histogram based Image Segmentation

- For a given image, decompose the range of pixel values $(0, \ldots, 255)$ into "discrete" intervals $R_t = [a_t, b_t]$, $t = 1, \ldots, T$, where T is the total number of segments.
- Each R_t is typically obtained as a range of pixel values that correspond to a hill of $h_A(l)$.
- "Label" the pixels with pixel values within each R_t via a point function.
- Main Assumption: Each object is assumed to be composed of pixels with $similar$ pixel values.

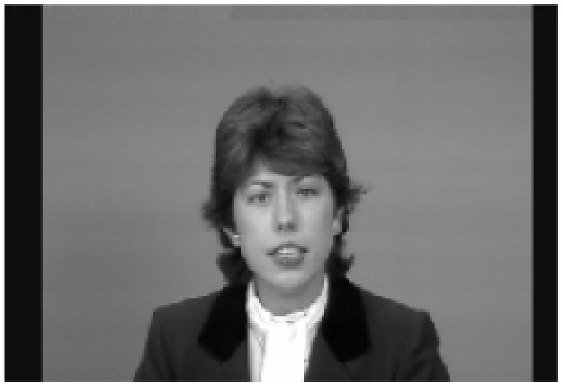
Example



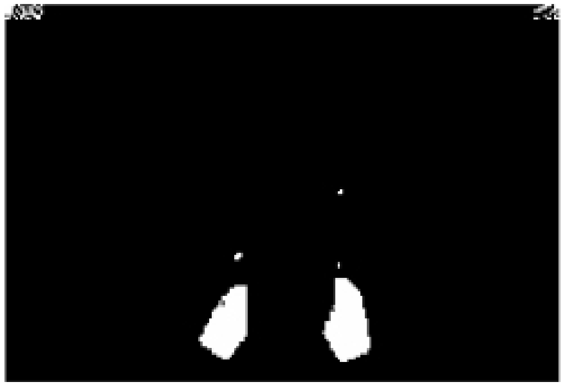


- $R_1 = [0, 14], R_2 = [15, 15], R_3 = [16, 99], R_4 = [100, 149], R_5 = [150, 220], R_6 =$

A

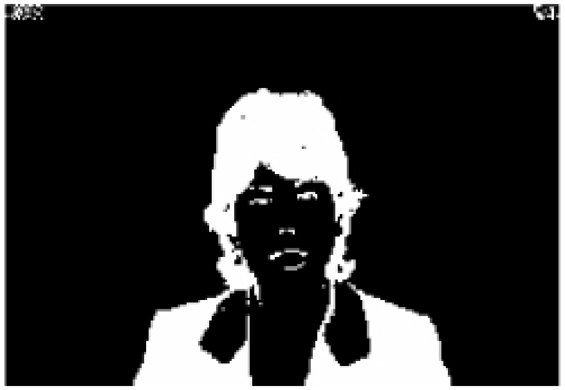


B1 (R_1=[0,14])

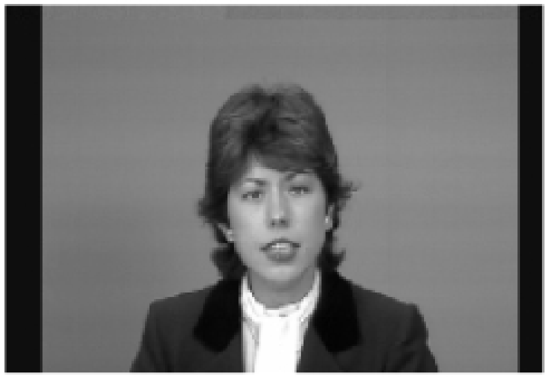


B2 (R_2=[15,15])

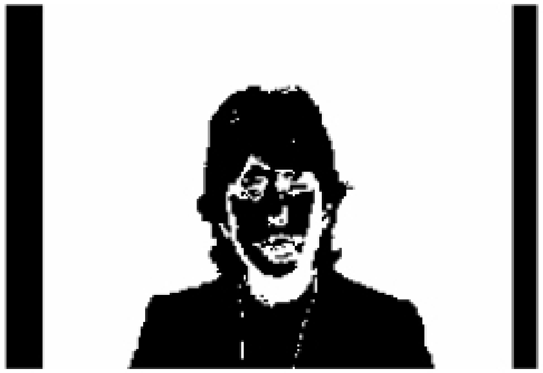
B3 (R_3=[16,99])



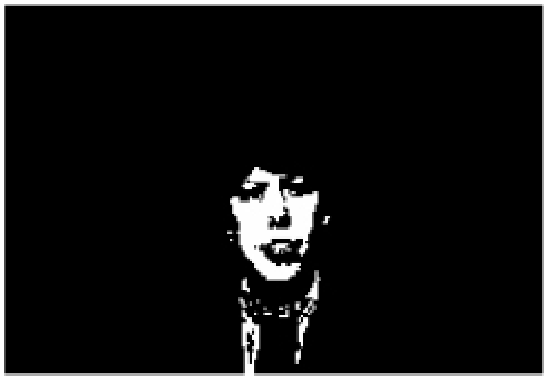
A



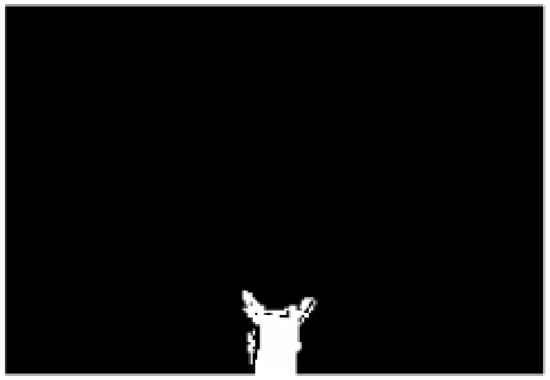
B4 (R_4=[100,149])



B5 (R_5=[150,220])

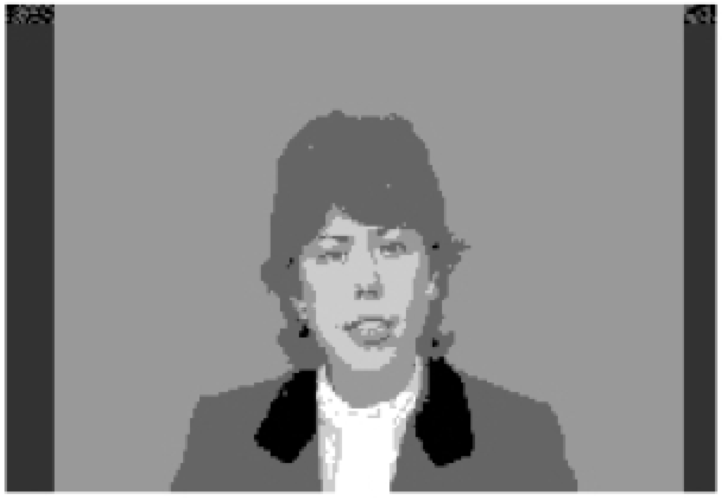


B6 (R_6=[221,255])



A

B (Histogram Segmented)



- Compute the sample mean of each segment ($>> m1 = sum(sum(B1.*A))/sum(sum(B1))$, etc.).
- $C = m1 \times B1 + m2 \times B2 + m3 \times B3 + m4 \times B4 + m5 \times B5 + m6 \times B6.$ $B(i,j) = g_s^C(C(i,j))$.

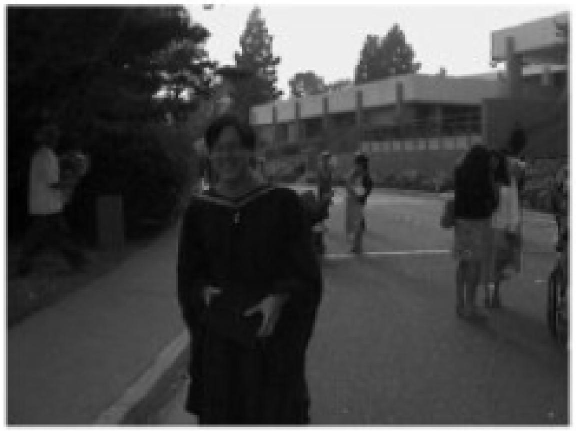
Limitations

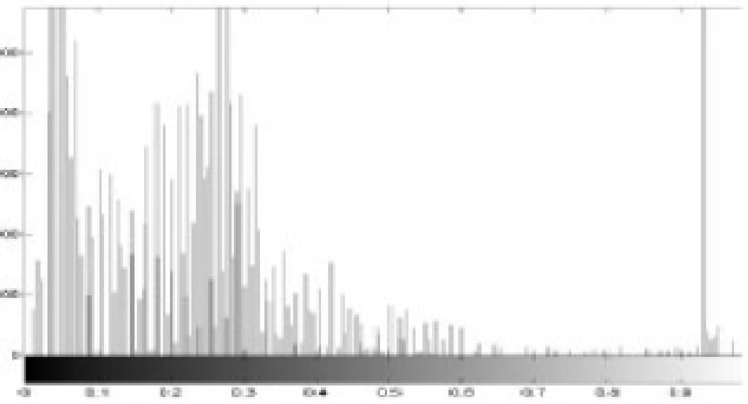
- Histogram based segmentation operates on each image pixel independently. As mentioned earlier, the main assumption is that objects must be composed of pixels with similar pixel values.
- This independent processing ignores a second important property: Pixels within an object should be *spatially* connected. For example, B3, B4, B5 group spatially disconnected objects/regions into the same segment.
- In practice, one would use histogram based segmentation in tandem with other algorithms that make sure that computed objects/regions are spatially connected.

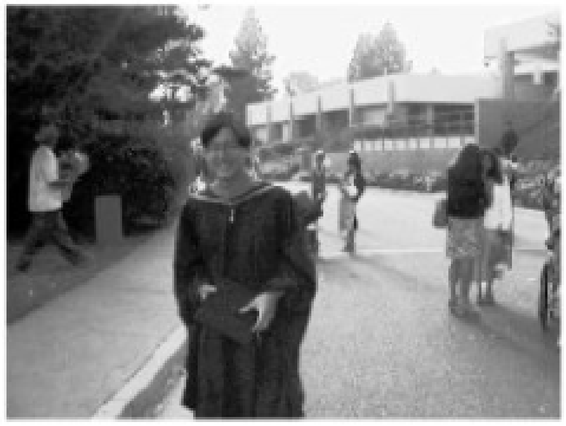
Histogram Equalization

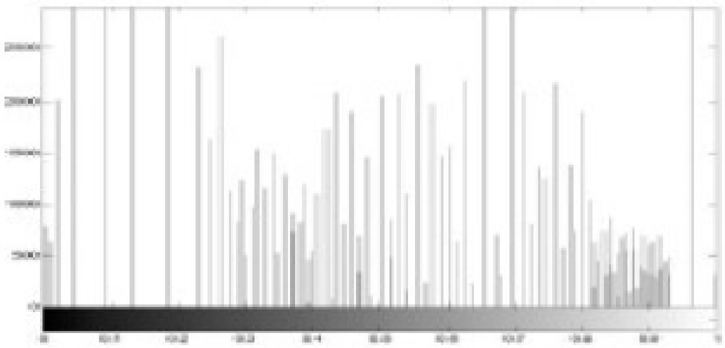
- For a given image $\mathbf{A}$, we will now design a special point function $g^e_A(l)$ which is called the histogram equalizing point function for $\mathbf{A}$.
- If $B(i,j) = g^e_A(A(i,j))$, then our aim is to make $h_B(l)$ as uniform/flat as possible *irrespective* of $h_A(l)$!
- Histogram equalization will help us:
  - Stretch/Compress an image such that:
    - Pixel values that occur frequently in $\mathbf{A}$ occupy a bigger dynamic range in $\mathbf{B}$, i.e., get stretched and become more visible.
    - Pixel values that occur infrequently in $\mathbf{A}$ occupy a smaller dynamic range in $\mathbf{B}$, i.e., get compressed and become less visible.

Histogram Equalization: Example









Histogram Equalization

Suppose our image has L different grey levels $0, 1, 2, \ldots L-1$, and that grey level i occurs n_i times in the image. Suppose also that the total number of pixels in the image is n (so that $n_0 + n_1 + n_2 + \cdots + n_{L-1} = n$. To transform the grey levels to obtain a better contrasted image, we change grey level i to

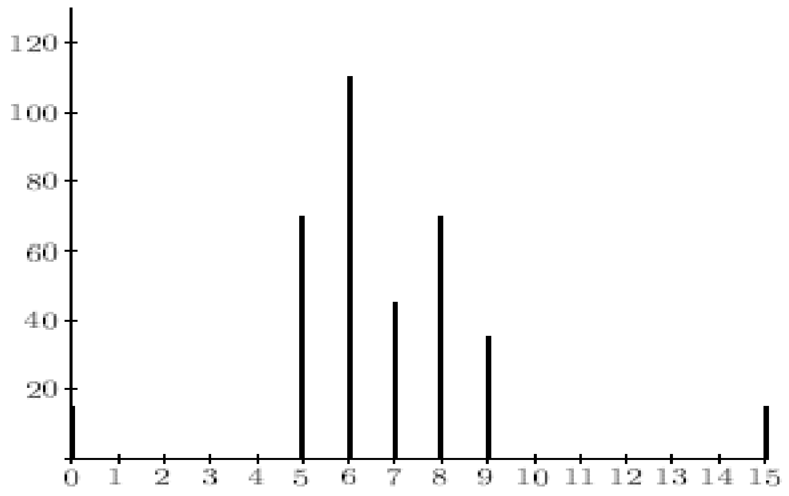
$$(n_0 + n_1 + n_2 + \ldots + n_i) \times (L-1)/n$$

- Round the output value to the nearest integer

Example

- Suppose we have an image with the histogram as shown in the figure below and associated with a table of the numbers n_i of grey values (n = 360):

Grey level i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n_i	15	0	0	0	0	0	0	0	0	70	110	45	80	40	0	0



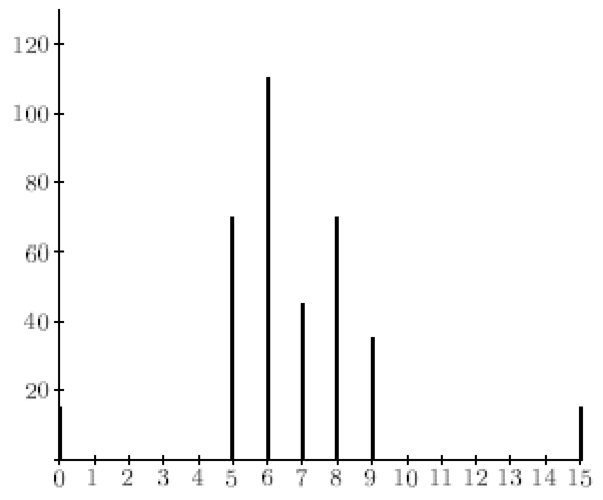
Grey level i	n_i	Σn_i	$(1/24)\Sigma n_i$	Rounded value
0	15	15	0.63	1
1	0	15	0.63	1
2	0	15	0.63	1
3	0	15	0.63	1
4	0	15	0.63	1
5	0	15	0.63	1
6	0	15	0.63	1
7	0	15	0.63	1
8	0	15	0.63	1
9	70	85	3.65	4
10	110	195	8.13	8
11	45	240	10	10
12	80	320	13.33	13
13	40	360	15	15
14	0	360	15	15
15	0	360	15	15

We now have the following transformation of grey values, obtained by reading off the first and last columns in the above table:

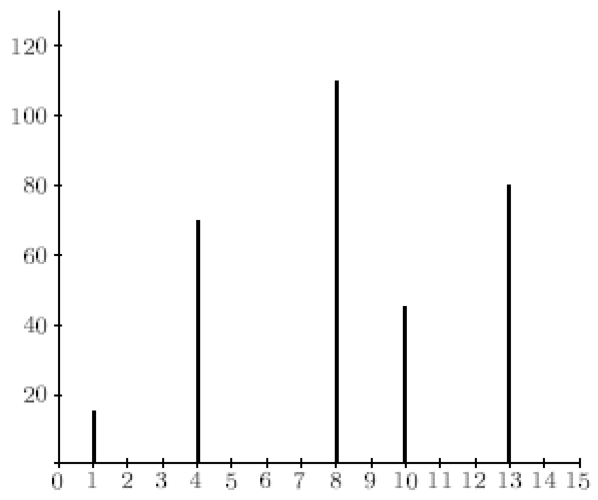
Original grey level i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Final grey level j	1	1	1	1	1	1	1	1	1	4	8	10	13	15	15	15

Histogram Equalization

Before Equalization



After Equalization



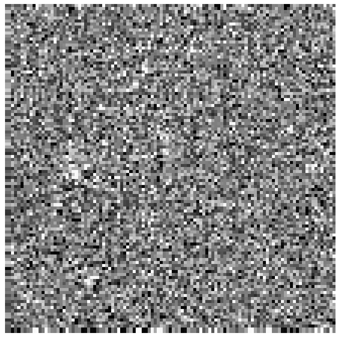
Bit Planes

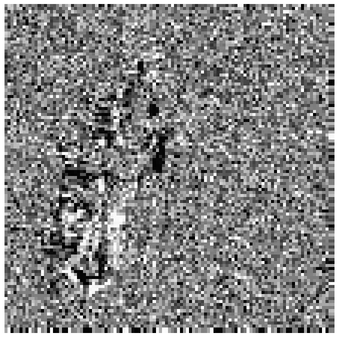
- Grey scale images can be transformed into a sequence of binary images by breaking them up into their bit-planes.
- If we consider the grey value of each pixel of an 8-bit image
- least significant bit plane
- Most significant bit plane

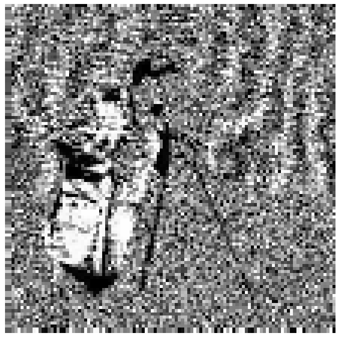
```
>> c=imread(‘image.jpg');
>> cd=double(c);

>> c0=mod(cd,2);
>> c1=mod(floor(cd/2),2);
>> c2=mod(floor(cd/4),2);
>> c3=mod(floor(cd/8),2);
>> c4=mod(floor(cd/16),2);
>> c5=mod(floor(cd/32),2);
>> c6=mod(floor(cd/64),2);
>> c7=mod(floor(cd/128),2);
```

Then try:

```
>> reconstructed_image=2*(2*(2*(2*(2*(2*(2*c7+c6)+c5)+c4)+c3)+c2)+c1)+c0;
>> imshow(uint8(reconstructed_image))
```


c0


c1


c2

c3

c4

c5

c6

c7

Spatial Resolution

- Spatial resolution is the density of pixels over the image:
- the greater the spatial resolution, the more pixels are used to display the image.
- Check imresize() function
  - imresize(x,1/2)
  - imresize(imresize(x,1/2),1/2)

Sampling & Interpolation

$$\begin{matrix} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & \cdots \\ x_{21} & \boxed{x_{22}} & x_{23} & \boxed{x_{24}} & x_{25} & \boxed{x_{26}} & \cdots \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & \cdots \\ x_{41} & \boxed{x_{42}} & x_{43} & \boxed{x_{44}} & x_{45} & \boxed{x_{46}} & \cdots \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & \cdots \\ x_{61} & \boxed{x_{62}} & x_{63} & \boxed{x_{64}} & x_{65} & \boxed{x_{66}} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{matrix} \longrightarrow \texttt{imresize(x,1/2)} \longrightarrow \begin{matrix} x_{22} & x_{24} & x_{26} & \cdots \\ x_{42} & x_{44} & x_{46} & \cdots \\ x_{62} & x_{64} & x_{66} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{matrix}$$

$$\begin{matrix} \boxed{\begin{matrix} x_{22} & x_{22} \\ x_{22} & x_{22} \end{matrix}} & \boxed{\begin{matrix} x_{24} & x_{24} \\ x_{24} & x_{24} \end{matrix}} & \boxed{\begin{matrix} x_{26} & x_{26} \\ x_{26} & x_{26} \end{matrix}} & \cdots \\ \boxed{\begin{matrix} x_{42} & x_{42} \\ x_{42} & x_{42} \end{matrix}} & \boxed{\begin{matrix} x_{44} & x_{44} \\ x_{44} & x_{44} \end{matrix}} & \boxed{\begin{matrix} x_{46} & x_{46} \\ x_{46} & x_{46} \end{matrix}} & \cdots \\ \boxed{\begin{matrix} x_{62} & x_{62} \\ x_{62} & x_{62} \end{matrix}} & \boxed{\begin{matrix} x_{64} & x_{64} \\ x_{64} & x_{64} \end{matrix}} & \boxed{\begin{matrix} x_{66} & x_{66} \\ x_{66} & x_{66} \end{matrix}} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{matrix}$$

Effects of reducing spatial resolution

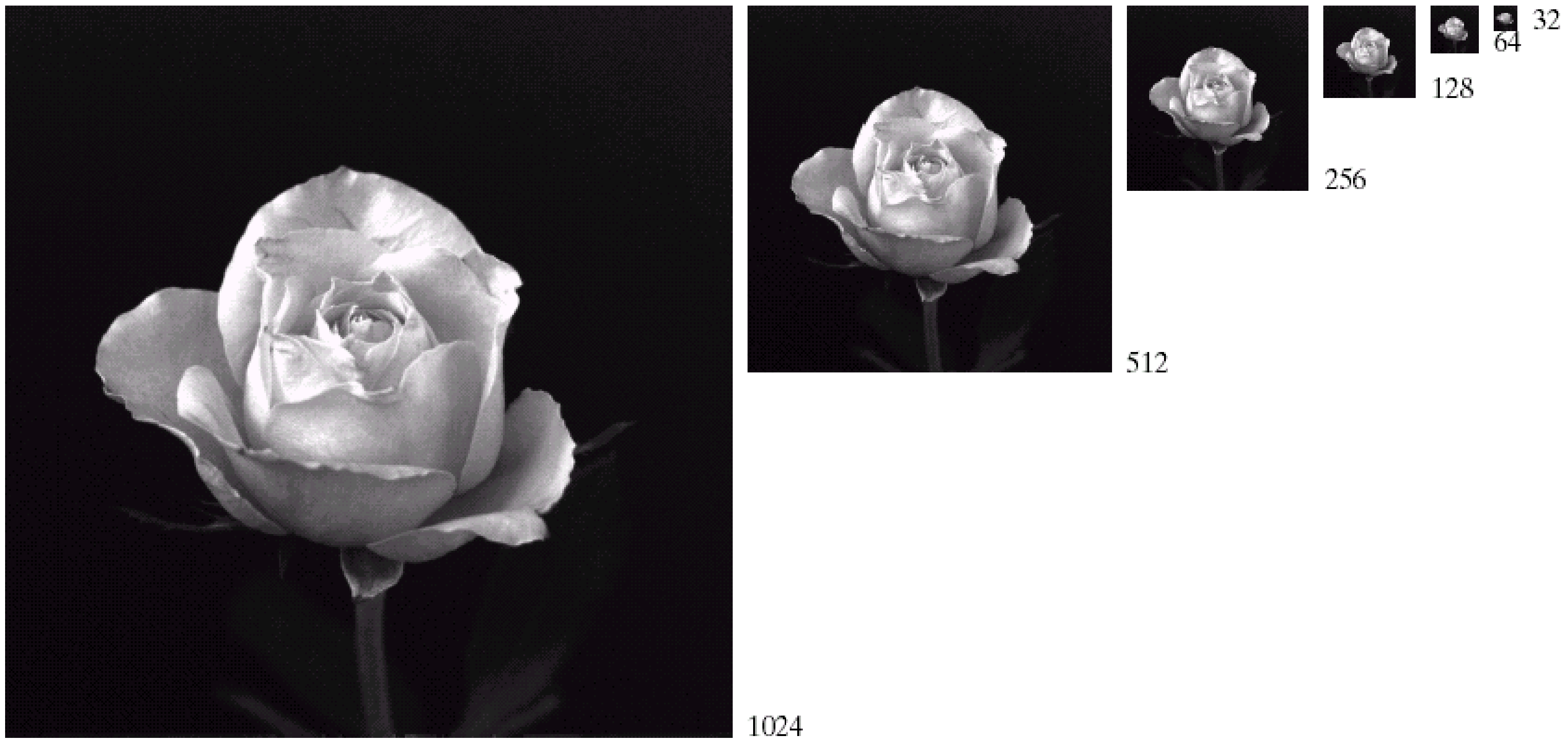


FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

Effects of reducing spatial resolution

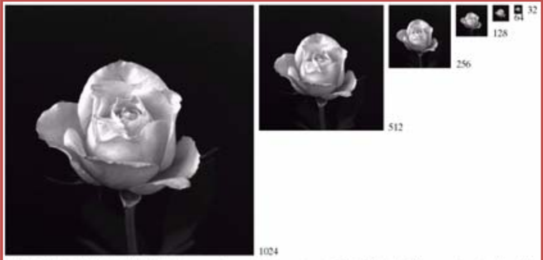


FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

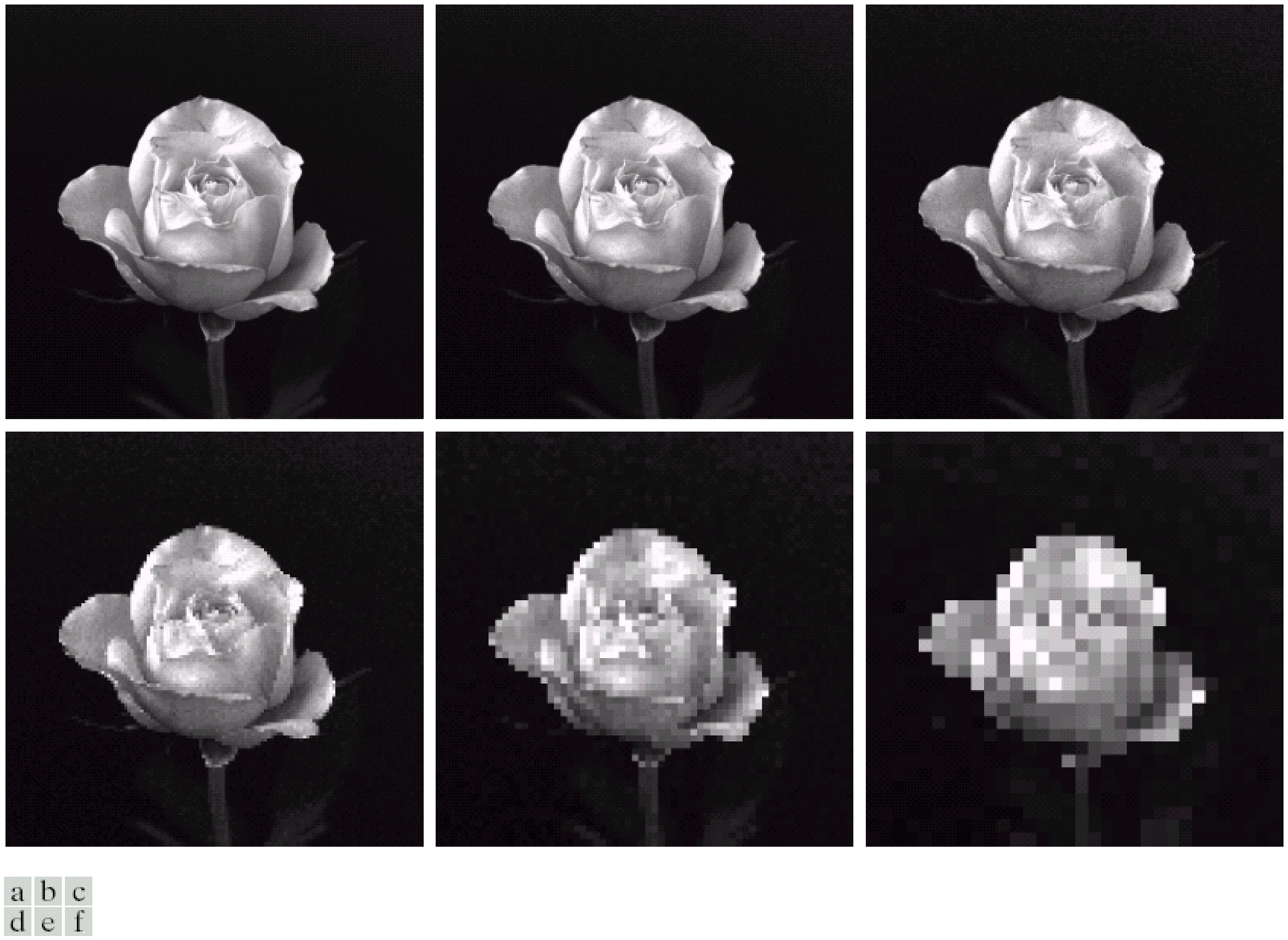


FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

Zooming by Interpolation

- Pixel replication (zero-order interpolation):
  Interpolated pixel in the enlarged image is set to the gray level of its nearest in the original image.

- Bilinear Interpolation
  - Interpolated pixel in the enlarged image is set to the weighted sum of its four nearest pixels in the original image.

Bilinear interpolation

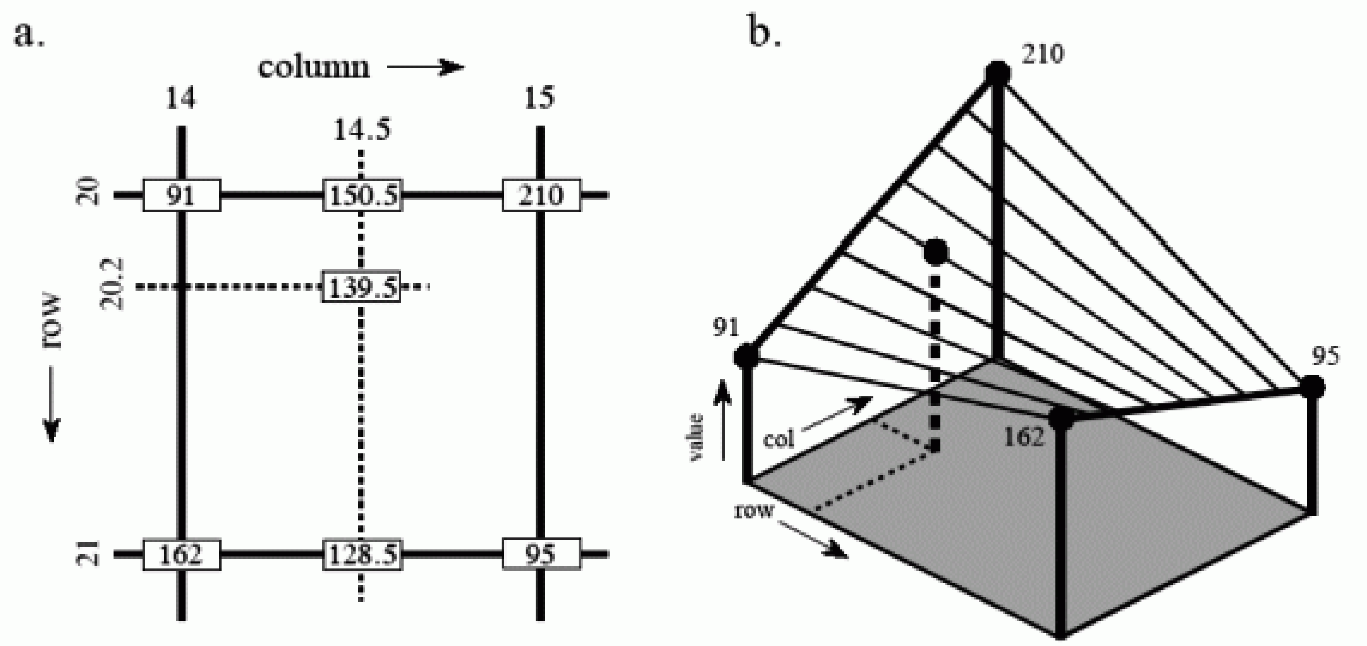
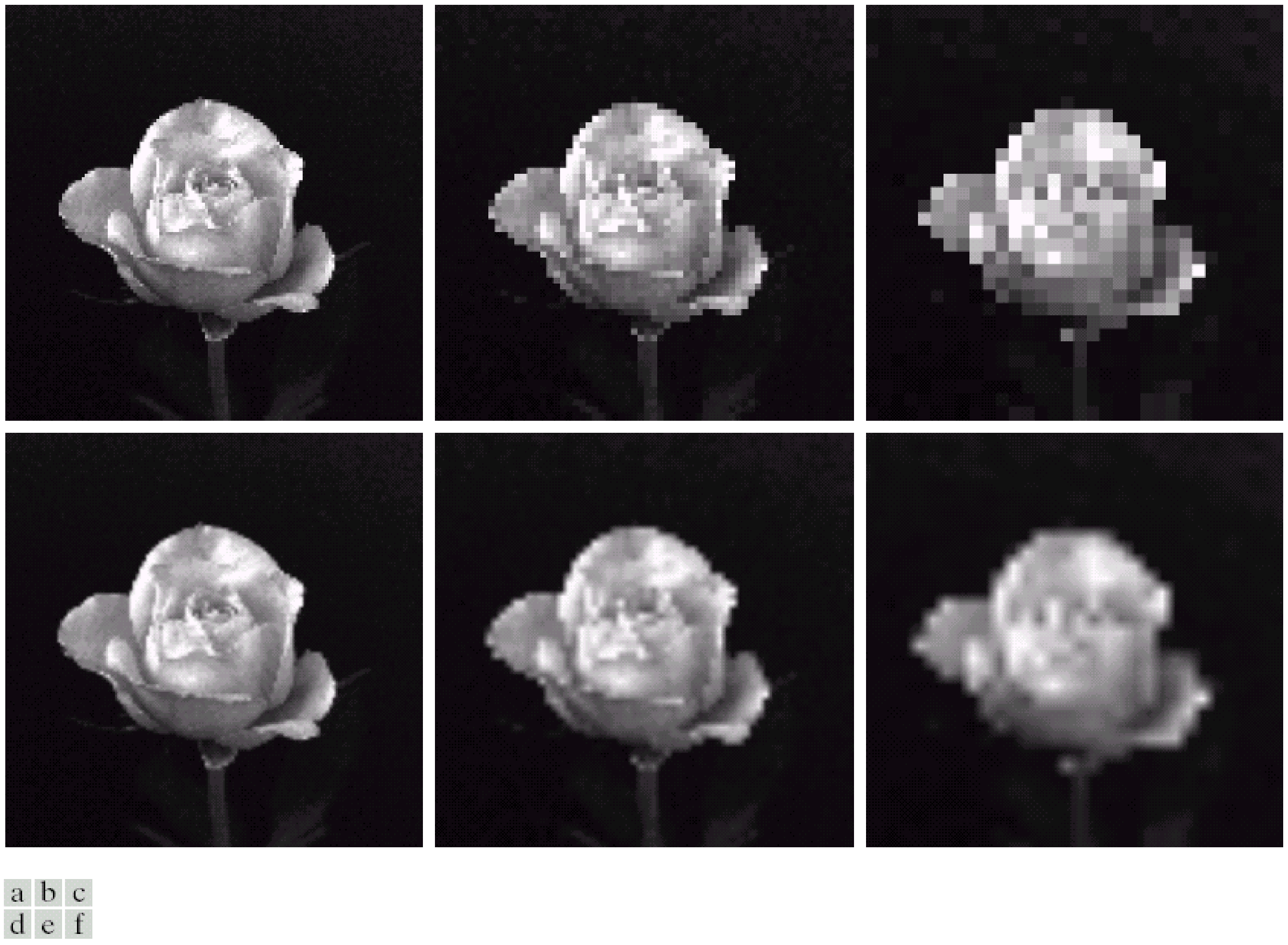


FIGURE 23-16
Subpixel interpolation. Subpixel interpolation for image warping is usually accomplished with bilinear interpolation. As shown in (a), two intermediate values are calculated by linear interpolation in the horizontal direction. The final value is then found by using linear interpolation in the vertical direction between the intermediate values. As shown by the three-dimensional illustration in (b), this procedure uniquely defines all values between the four known pixels at each of the corners.

Example of interpolation



Pixel repetition

Bilinear interpolation

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.

Image difference measure

- MSE
- PSNR
- SNR
- MAE

Image difference measure

MSE: Mean Squared Error for images *f* and *g*:

$$\mathrm{MSE} = \frac{1}{N_x N_y} \sum_{y=1}^{N_y} \sum_{x=1}^{N_x} \left(f(x,y) - g(x,y) \right)^2$$

Image difference measure

PSNR: Peak-Signal-Noise-Ratio

$$PSNR = 10 \cdot \log_{10} \frac{f_{peak}^2}{MSE}$$

f_{peak} is the maximum possible pixel value of the image (f_{peak}=255).

and

$$\mathrm{MSE} = \frac{1}{N_x N_y} \sum_{y=1}^{N_y} \sum_{x=1}^{N_x} \left(f(x,y) - g(x,y) \right)^2$$

Image difference measure

SNR: Signal-Noise-Ratio

$$SNR = 10\log_{10}\frac{\sigma_f^2}{MSE}$$

Variance of *f*

$$\sigma_f^2 = \frac{1}{N_x N_y}\sum_{y=1}^{N_y}\sum_{x=1}^{N_x}\left(f(x,y) - a_f\right)^2$$

Mean value of *f*

$$a_f = \frac{1}{N_x N_y}\sum_{y=1}^{N_y}\sum_{x=1}^{N_x} f(x,y)$$

MSE

$$\mathrm{MSE} = \frac{1}{N_x N_y}\sum_{y=1}^{N_y}\sum_{x=1}^{N_x}\left(f(x,y) - g(x,y)\right)^2$$

Image difference measure

MAE: Mean Absolute Error for images *f* and *g*:

$$\mathrm{MAE} = \frac{1}{N_x N_y} \sum_{y=1}^{N_y} \sum_{x=1}^{N_x} \left| f(x,y) - g(x,y) \right|$$

Relationship between pixels

- Neighborhood
- Adjacency
- Connectivity
- Region & Boundary
- Distance

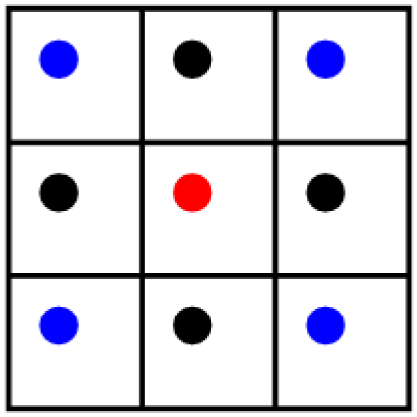
Neighborhood

A pixel P at coordinates (x,y) has eight neighbors:

$N_4(P) = \{(x+1,y), (x-1,y),(x,y+1),(x,y-1)\}$

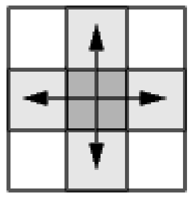
$N_D(P) = \{(x+1,y+1), (x-1,y-1),(x-1,y+1),(x+1,y-1)\}$

$N_8(P) = N_4(P) \cup N_D(P)$

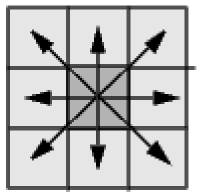


Connectivity

4-connectivity: Pixels are connected if their edges touch. This means that a pair of adjoining pixels are part of the same object only if they are both on and are connected along the horizontal or vertical direction.



8-connectivity: Pixels are connected if their edges or corners touch.
This means that if two adjoining pixels are on, they are part of the same object, regardless of whether they are connected along the horizontal, vertical, or diagonal direction.

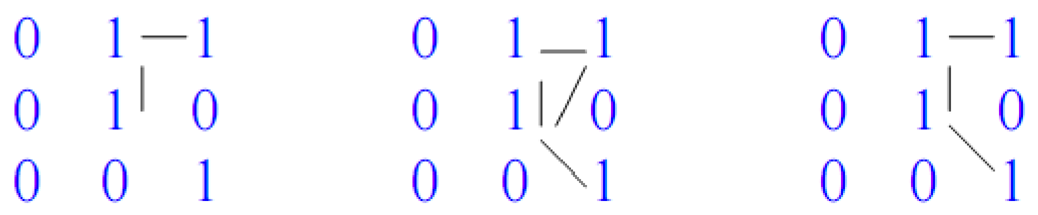


Adjacency

- Two pixels are adjacent if they are neighbors and their gray levels are similar
- V: set of gray levels
- Similar gray level means that the gray levels of both pixels belong to set V
- Exp:
  - Binary images: V={1}
  - Gray level image: V={32,33, …,63,64}

Adjacency

- 4-adjacency: Two pixels p and q with values from V are 4-adjacent if q is in $N_4(p)$
- 8-adjacency: Two pixels p and q with values from V are 8-adjacent if q is in $N_8(p)$
- 4-adjacency: broken paths
- 8-adjacency: multiple paths

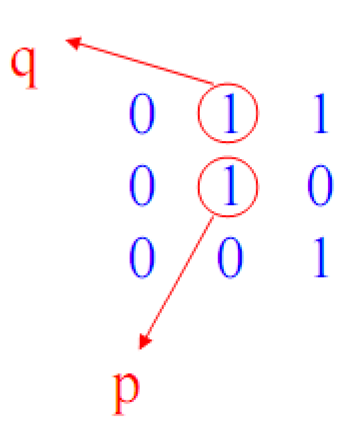


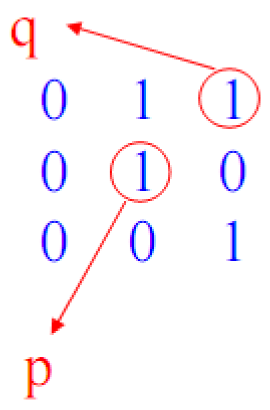
M-adjacency

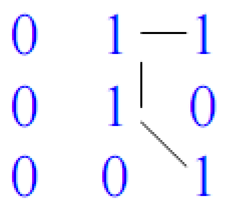
- m-adjacency: Two pixels p and q with values from V are m-adjacent if:

  q is in $N_4(p)$ or

  q is in $N_D(p)$ and the intersection of $N_4(p)$ and $N_4(q)$ has no pixels with values in V.

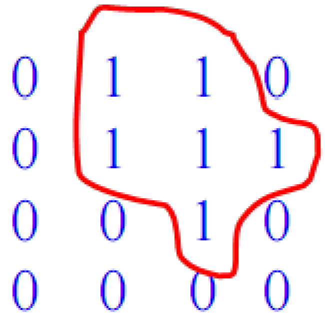






Region & Boundary

- R: a subset of pixels in an image.
- R is called a region if every pixel in R is connected to any other pixel in R. *(4- or 8-connected?)*
- Boundary (border or contour) of a region: set of pixels in the region that have one or more neighbors that are not in R.



$$\begin{array}{cccc} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{cccc} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{array}$$

Distance measure

- For pixels p,q, and z with coordinates (x,y), (s,t) and (v,w), respectively, D is a distance functions if:

$$D(p,q) \geq 0$$

$$D(p,q) = D(q,p)$$

$$D(p,z) \leq D(p,q) + D(q,z)$$

$$D_e = [(x-s)^2 + (y-t)^2]^{1/2}$$

Distance measure

- L_1, D_4 , City-block distance:

  $D(p1,p2)=|x1 - x2| + |y1 - y2|$

- L_∞, D_8, Chessboard, Chebyshev distance

  $D(p1,p2)=\max\{|x1 - x2|, |y1 - y2|\}$

- L_2, D_e, Euclidean distance:

  $D(p1,p2)=((x1 - x2)^2 + (y1 - y2)^2)^{1/2}$

Distance measure

$$\boxed{\begin{array}{ccc} 2 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 2 \end{array}}$$

- L_1, City-block, or D_4 distance

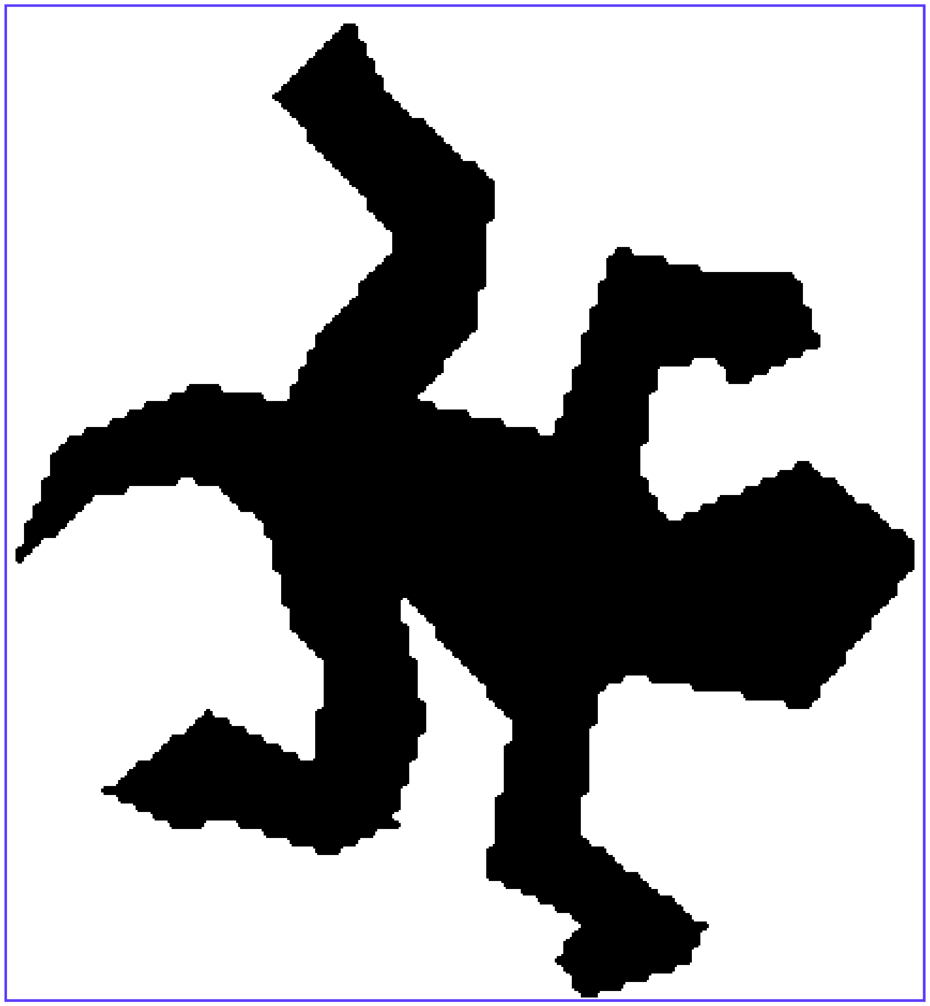
$$\boxed{\begin{array}{ccc} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{array}}$$

- L_∞, Chessboard, Chebyshev, or D_8 distance

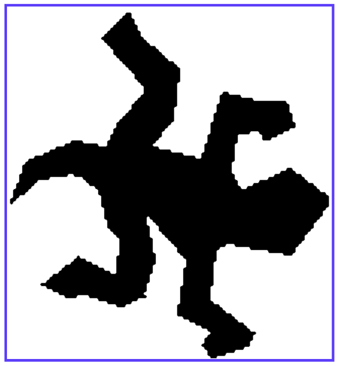
$$\boxed{\begin{array}{ccc} \sqrt{2} & 1 & \sqrt{2} \\ 1 & 0 & 1 \\ \sqrt{2} & 1 & \sqrt{2} \end{array}}$$

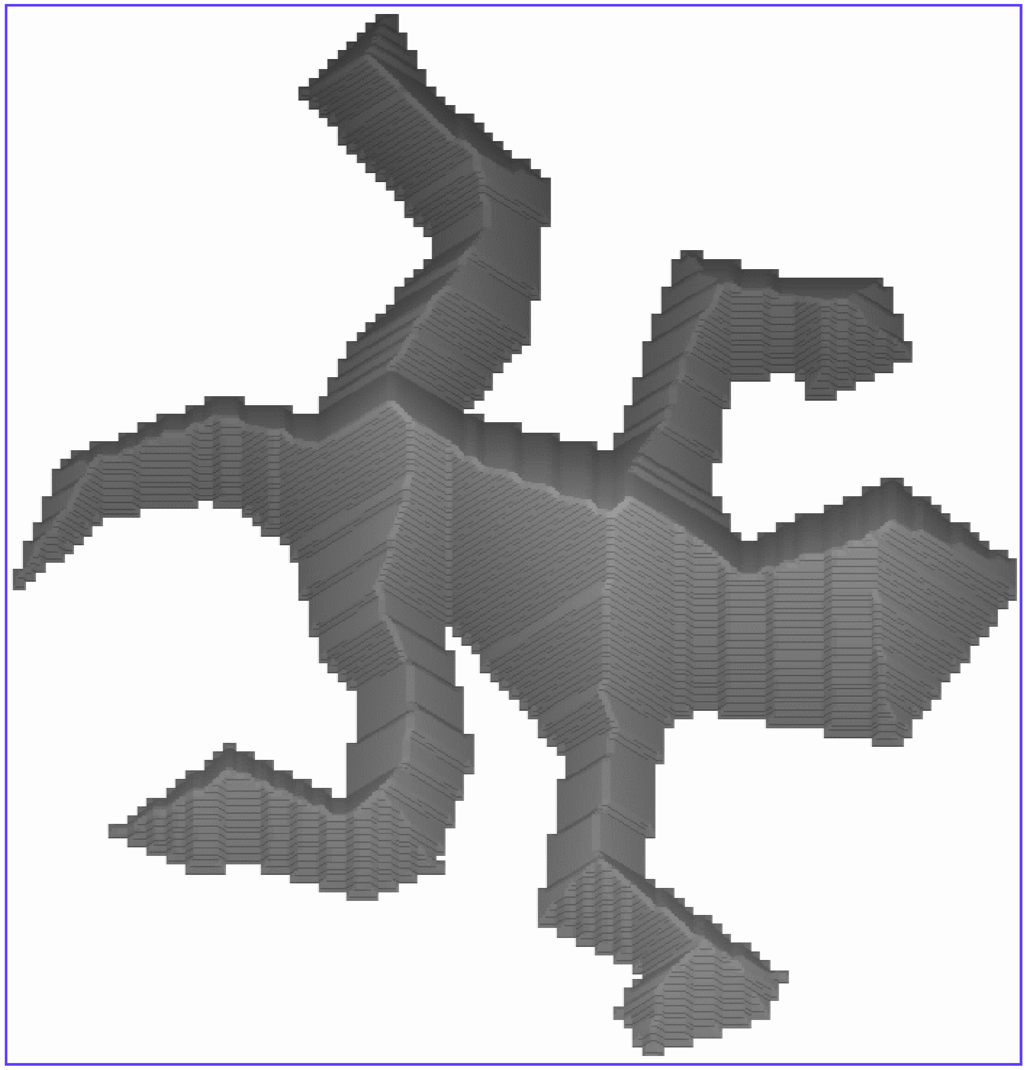
- L_2, or Euclidean distance

Distance Transform



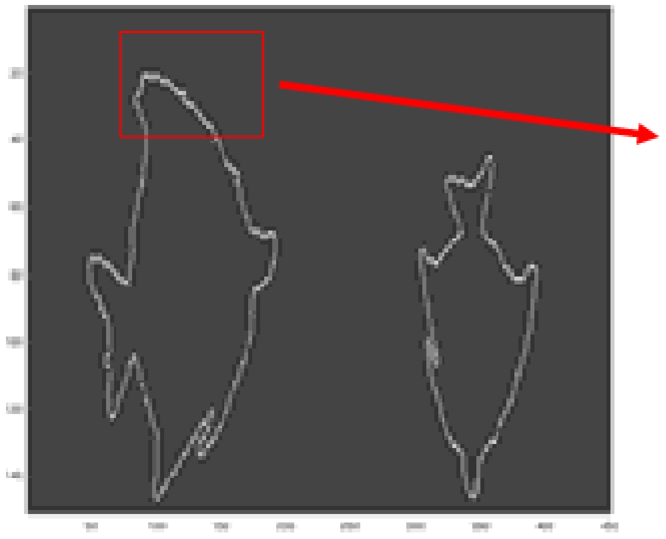
Distance Transform





(more later...)

Relationship between pixels



$$\begin{matrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0
\end{matrix}$$