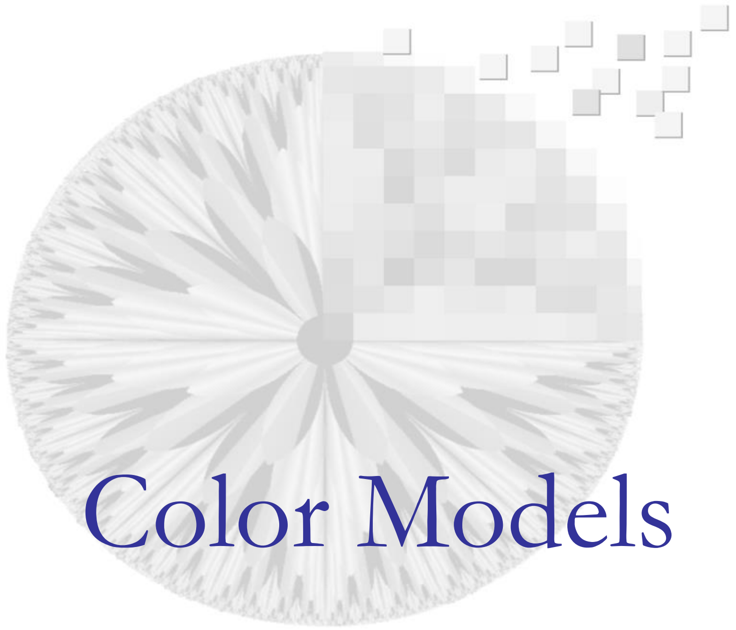
Low Level Image Processing

4097/5097

Deep Learning w CV Apps

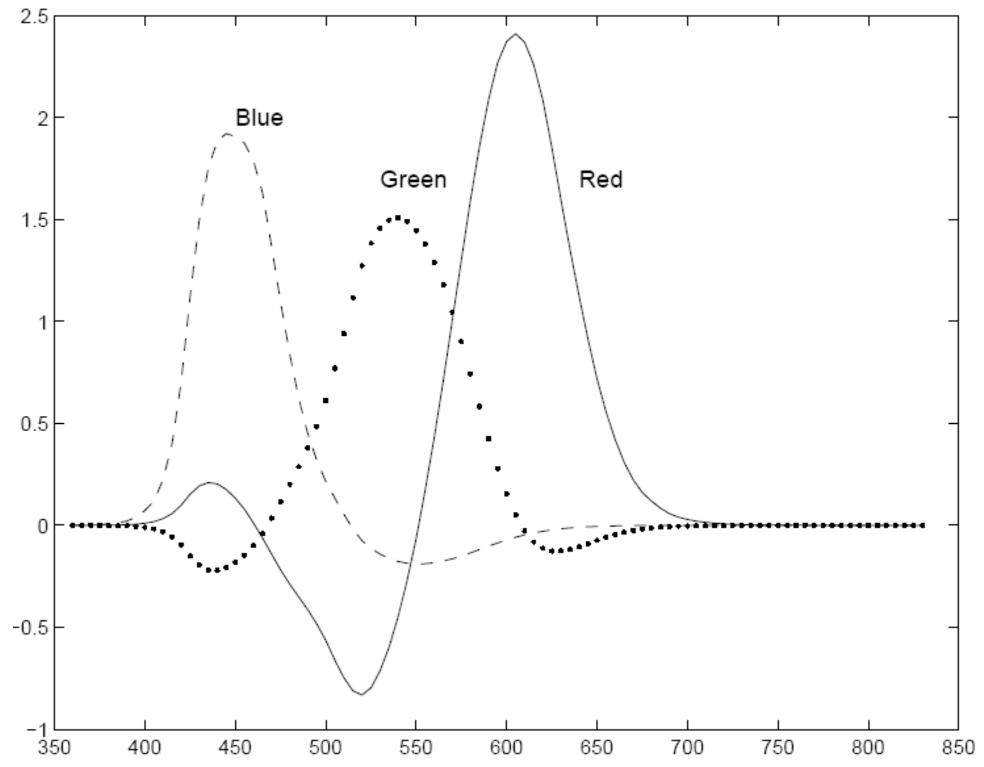
Slide Credits: Digital Image Processing, Gonzalez & Woods



Color Models

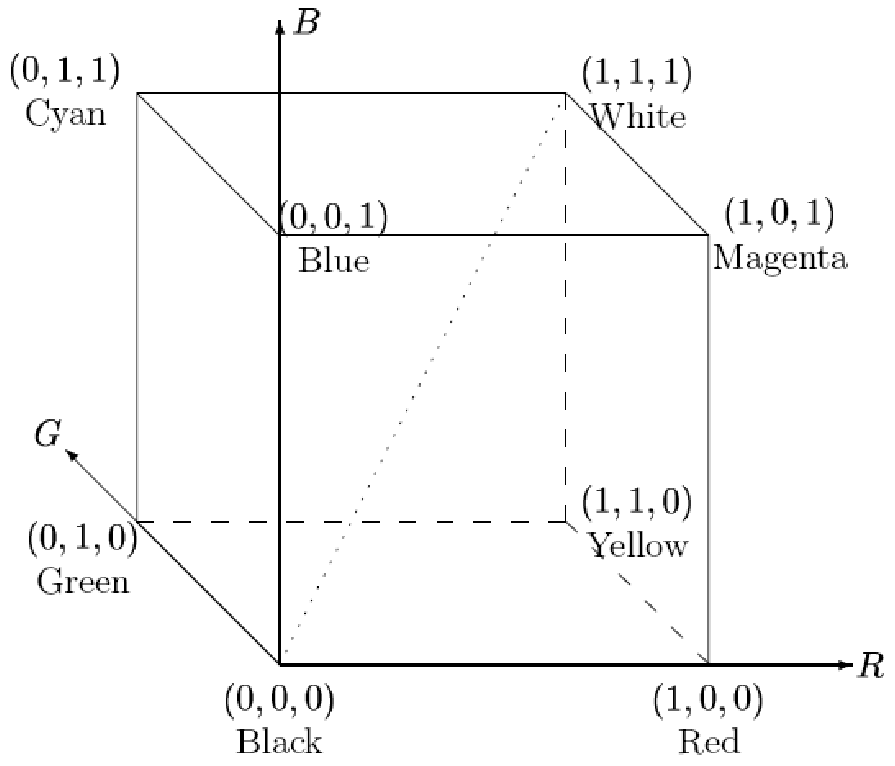
Color Perception

- Human perceive color as being made up of varying amounts of red, green and blue (aka primary colors)
- Experiment: people asked to match a given color to different amounts of the additive primaries red, green and blue.



RGB Images

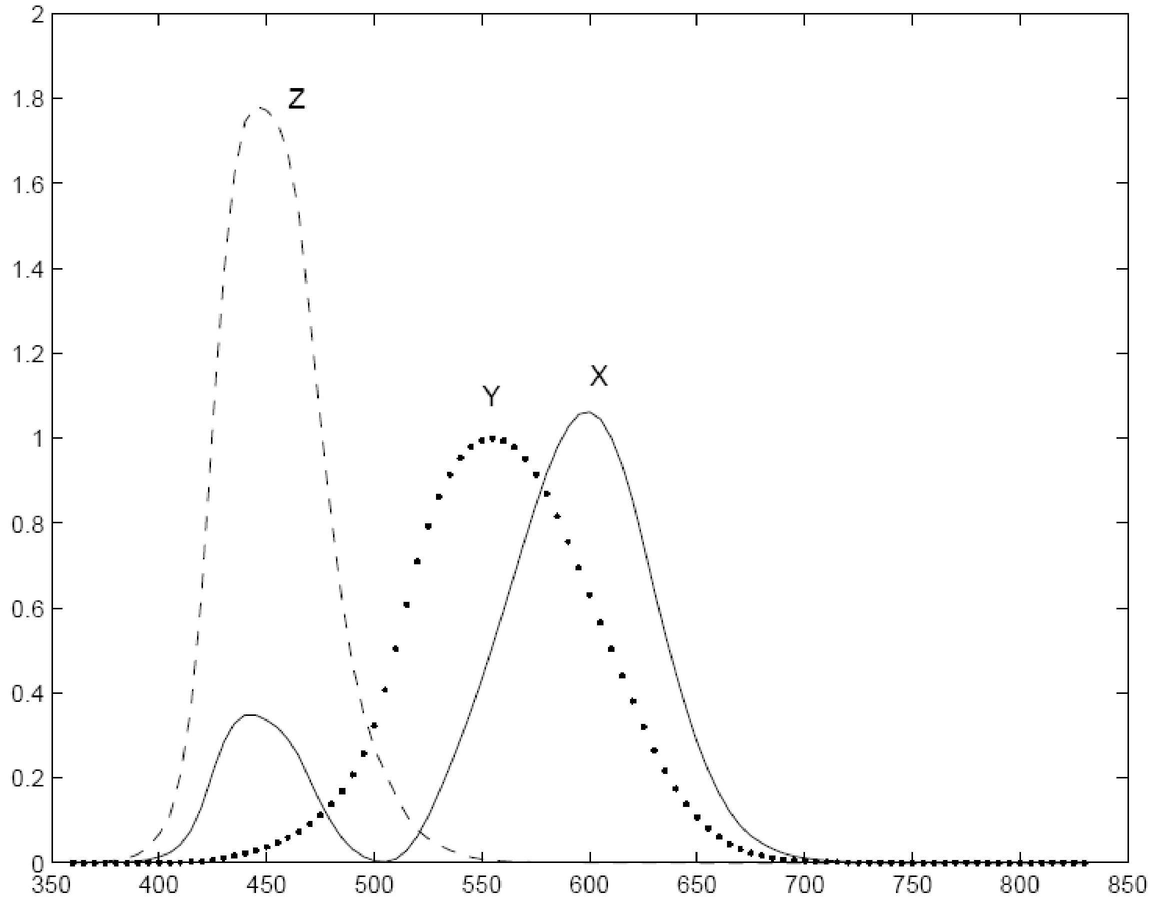
- Three matrices make up the image
- Each matrix represents: R, G, B
- RGB is the standard for the display of colors: on computer monitors; on TV sets.



- For some wavelengths negative values are need.
  - Physically impossible
- CIE (Commission Internationale d'Eclairage) responsible for color standards devised XYZ color model based on the experiment.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.431 & 0.342 & 0.178 \\ 0.222 & 0.707 & 0.071 \\ 0.020 & 0.130 & 0.939 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

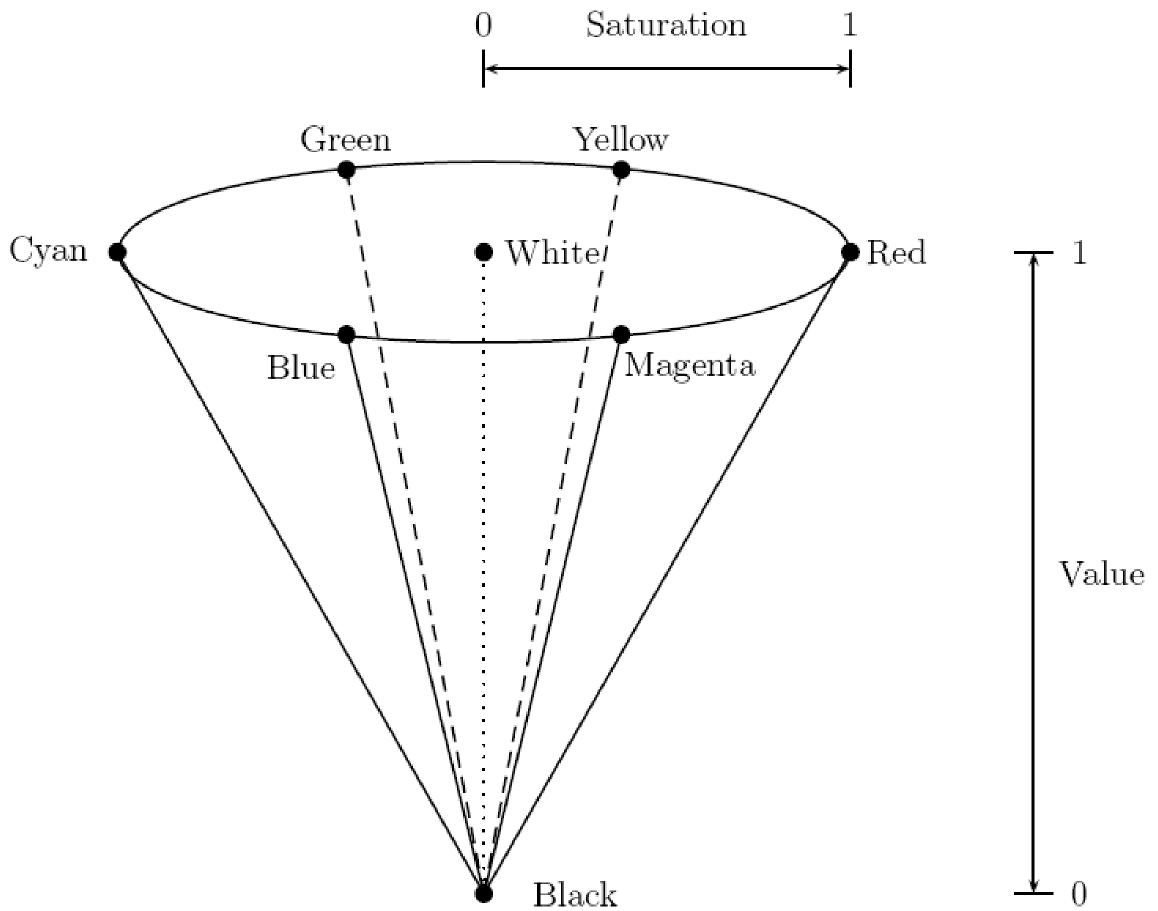
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.063 & -1.393 & -0.476 \\ -0.969 & 1.876 & 0.042 \\ 0.068 & -0.229 & 1.069 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$


Z
Y
X
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
350
400
450
500
550
600
650
700
750
800
850

HSV

- Hue: The true color
- Saturation: amount by which the color as been diluted with white.
- Value: intensity

Colour	Hue
Red	0
Yellow	0.1667
Green	0.3333
Cyan	0.5
Blue	0.6667
Magenta	0.8333



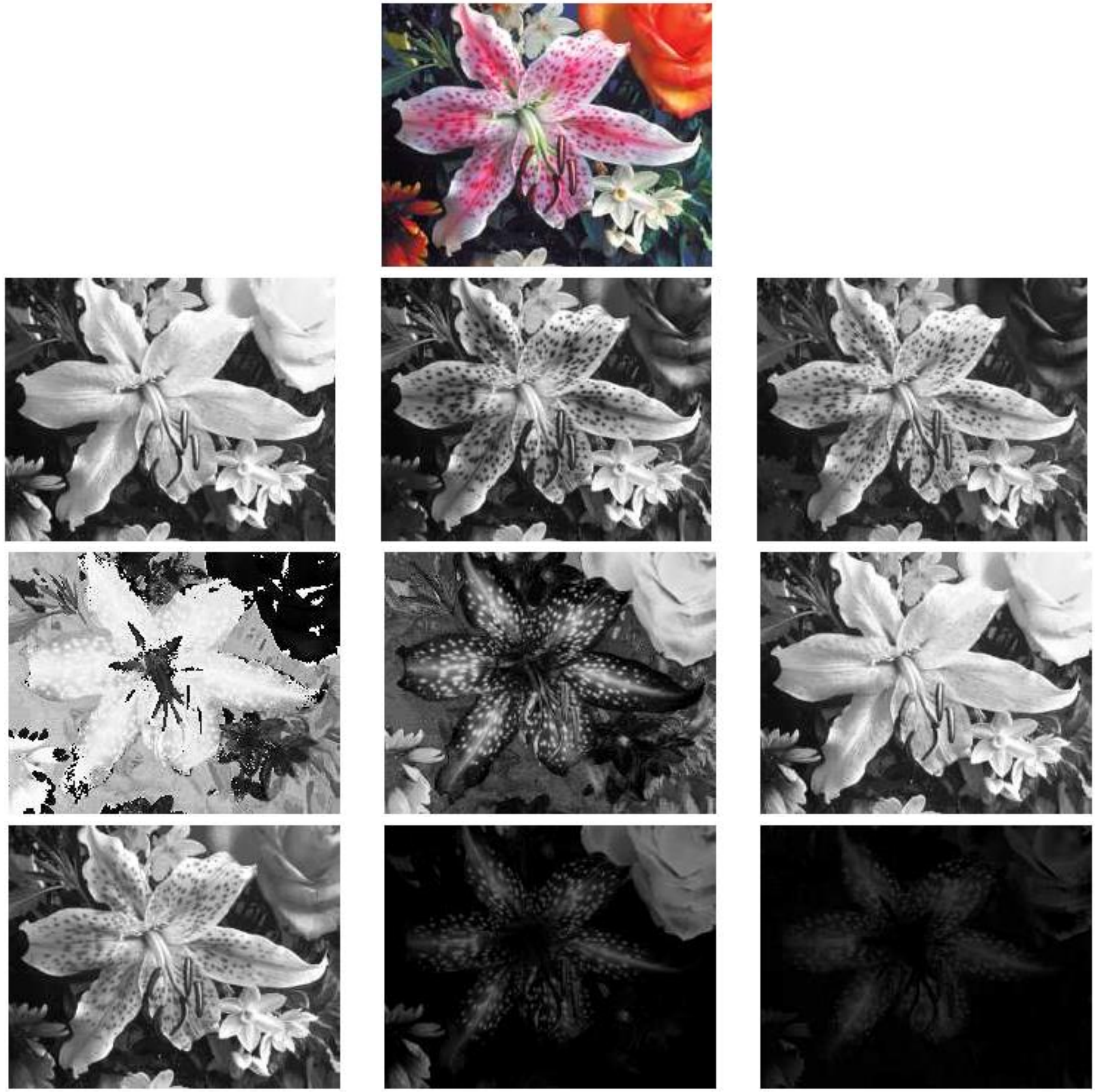
- Given rgb value
  - V = max(R,G,B)
  - d = max(R,G,B) – min(R,G,B)
  - S = d/ max(R,G,B)
  - If V=R => H = (G-B)/6d
  - If V=G => H = (B-R)/6d + 1/3
  - If V=B => H = (R-G)/6d + 2/3
  - If H < 0 add 1
  - RGB = (0,0,0) => HSV = (0,0,0)

YIQ

- Color model used for TV/video in US (NTSC)
- Y = luminance (intensity), I and Q carry the color information.

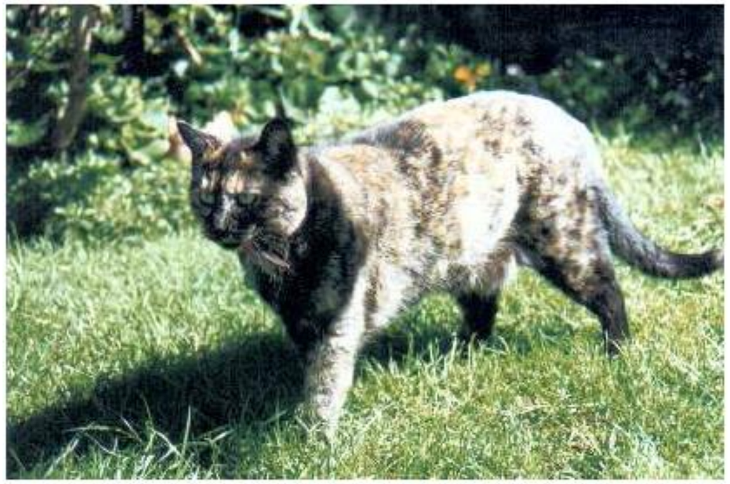
$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}$$

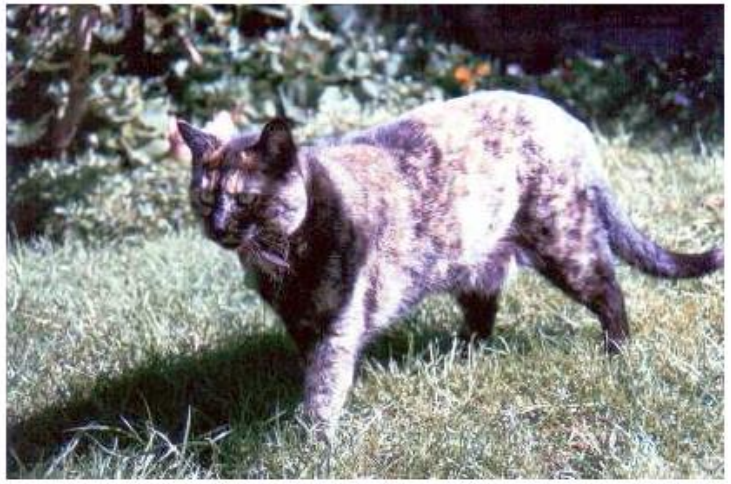


Applications

- Contrast enhancement: Histogram equilization of color.
  - Best: apply to Y of YIQ. Instead of applying to R, G, B.

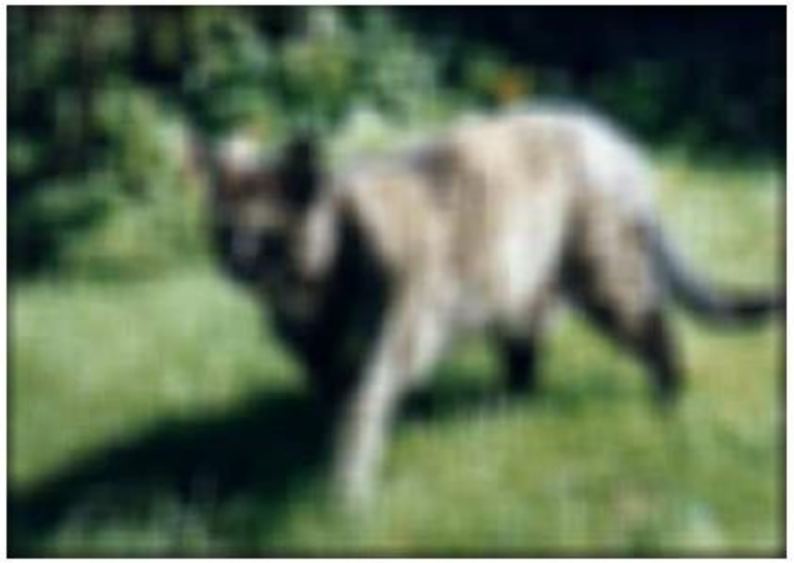


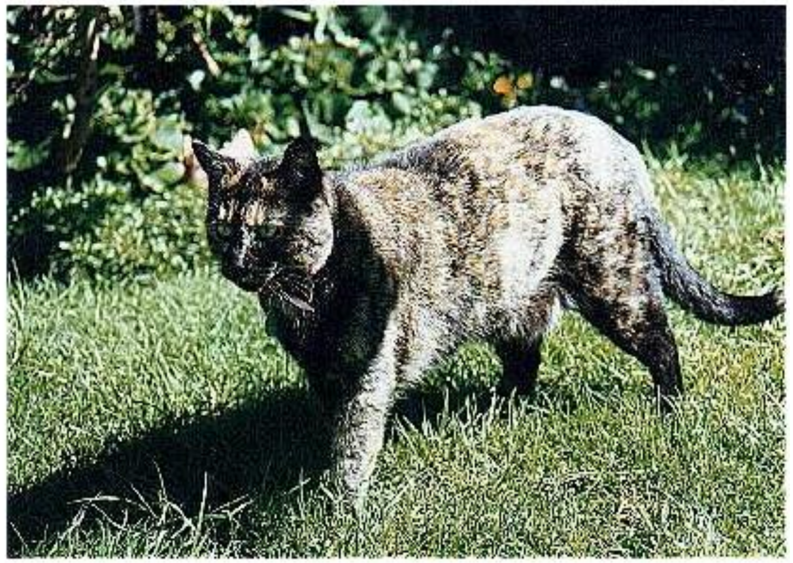
Intensity processing



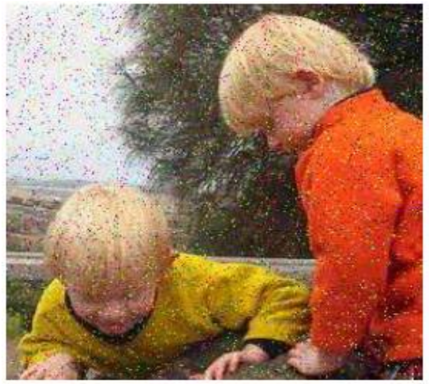
Using each RGB component

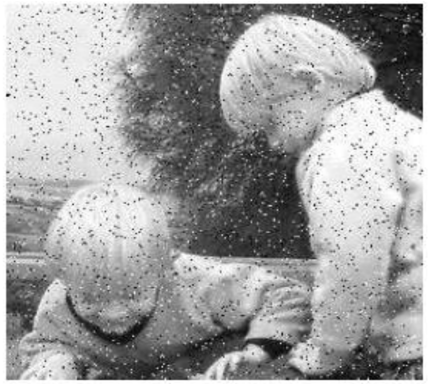
- Spatial filtering: Applying LPF, HPF to color
  - Can be applied to R, G, B seperately.
  - Best: apply to Y of YIQ.


Low pass filtering


High pass filtering

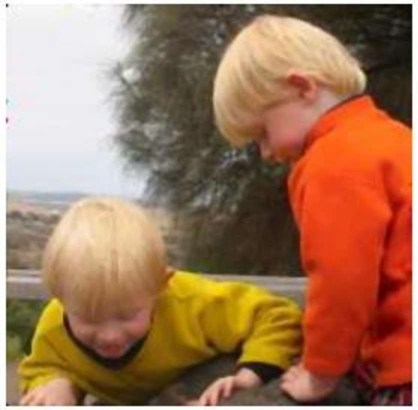
- Noise reduction
  - Application dependent
  - Salt n Pepper noise: best if applied to R, G, B

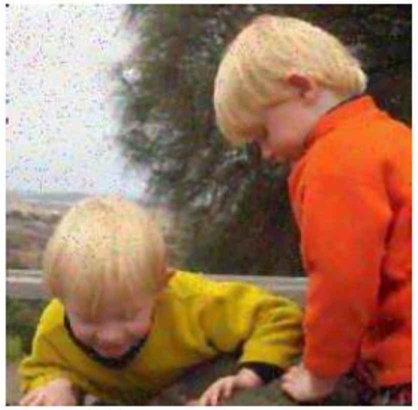

Salt & pepper noise


The red component

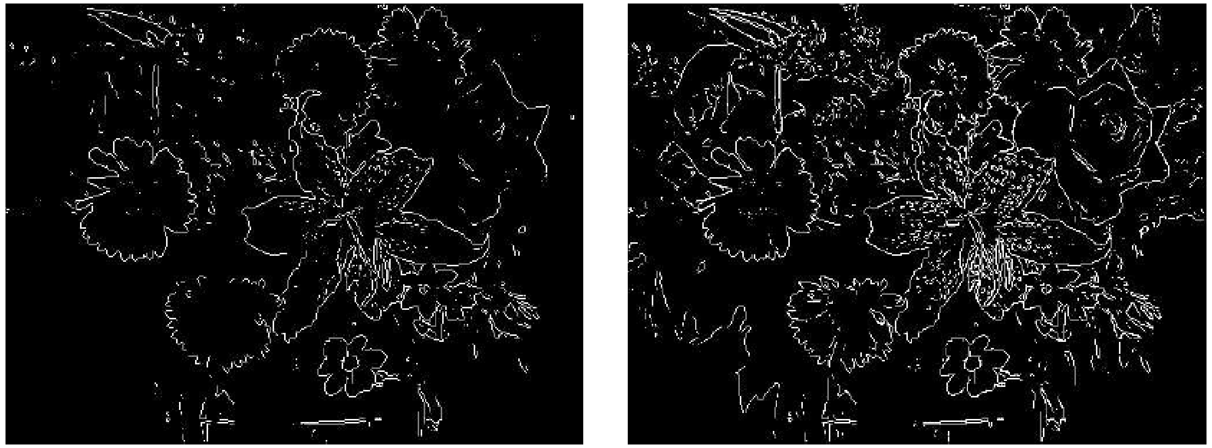
The green component

The blue component


Denoising each RGB component

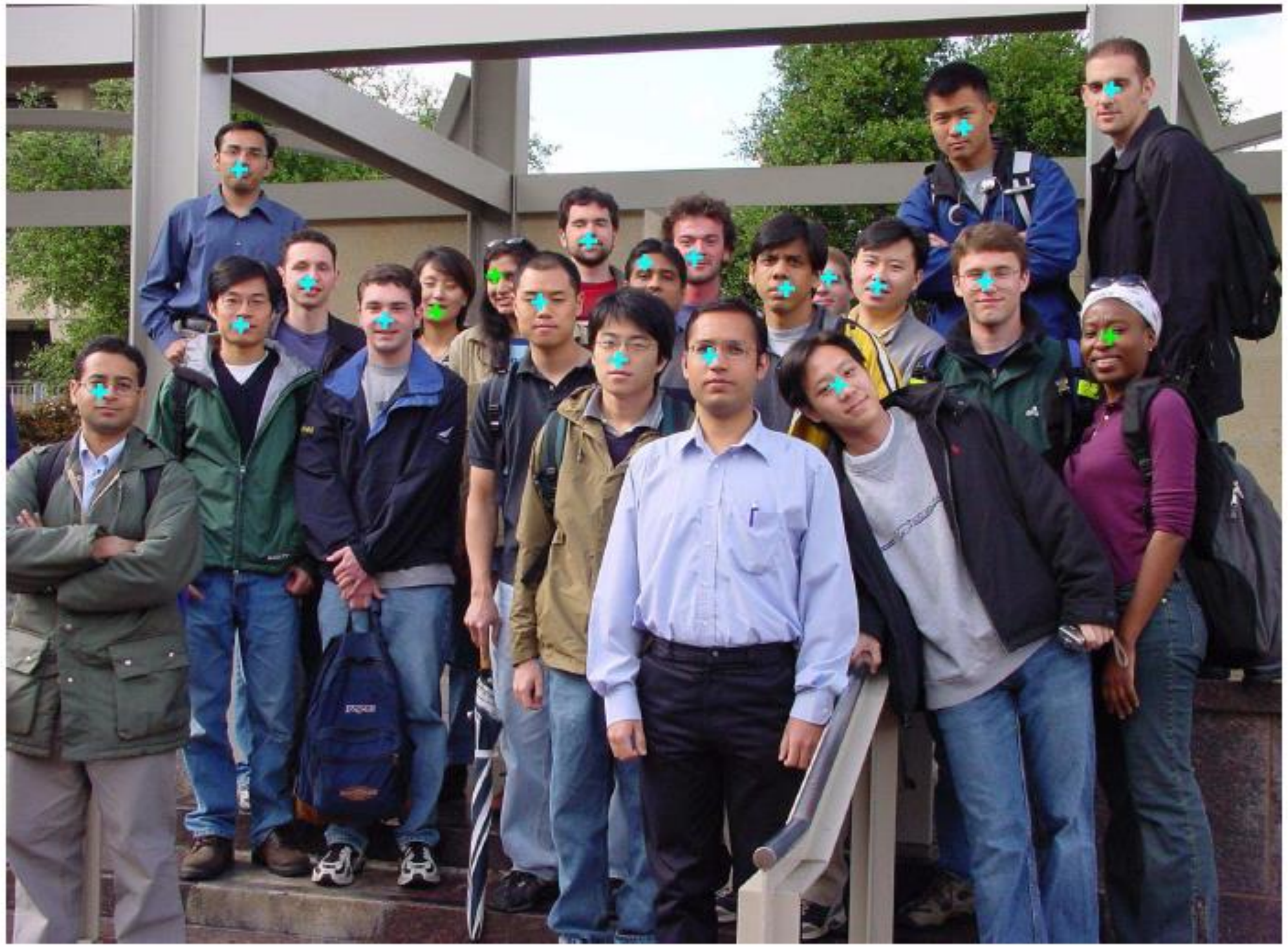

Denoising Y only

- Edge Detection: apply to intensity or RGB.

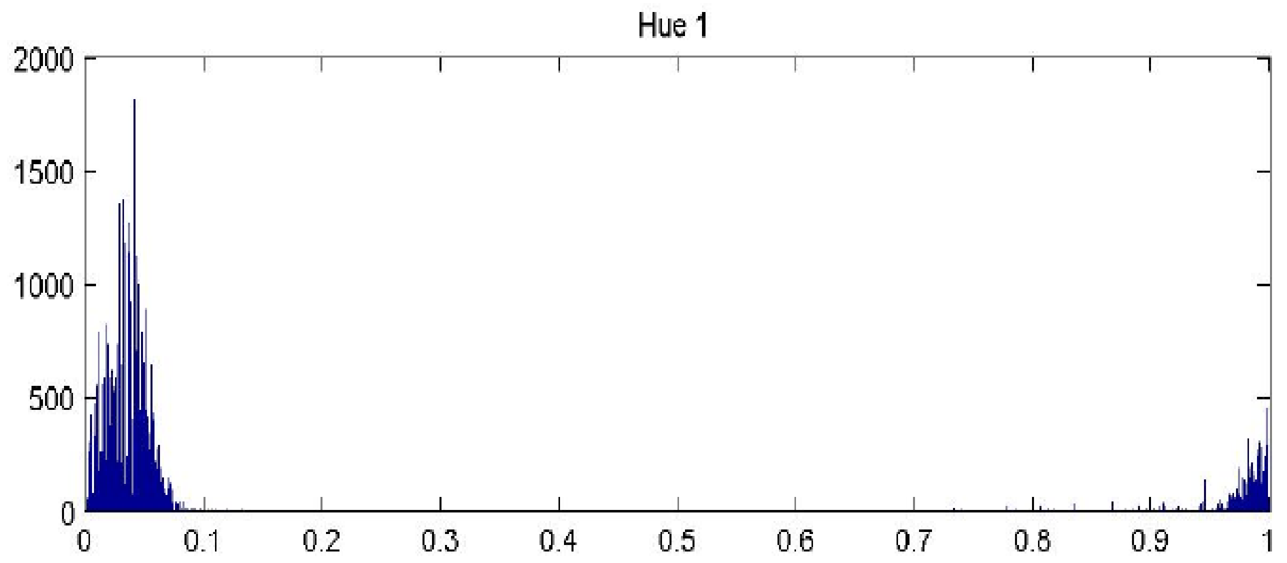


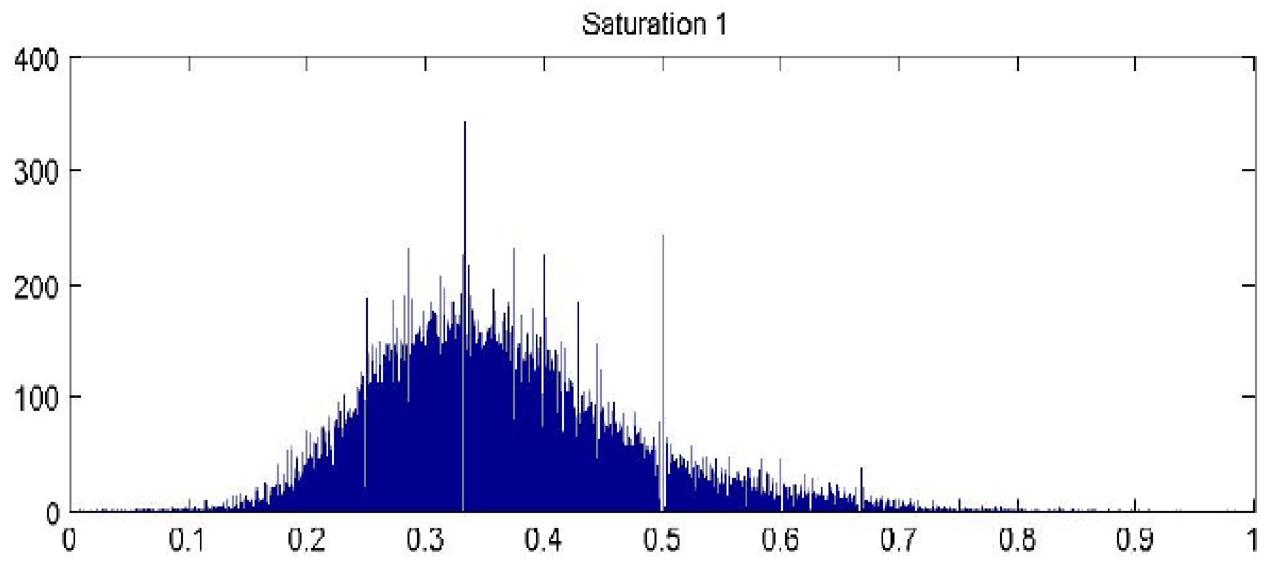
Edge detection applied to grayscale (left) and R,G,B (right)

Detecting Skin

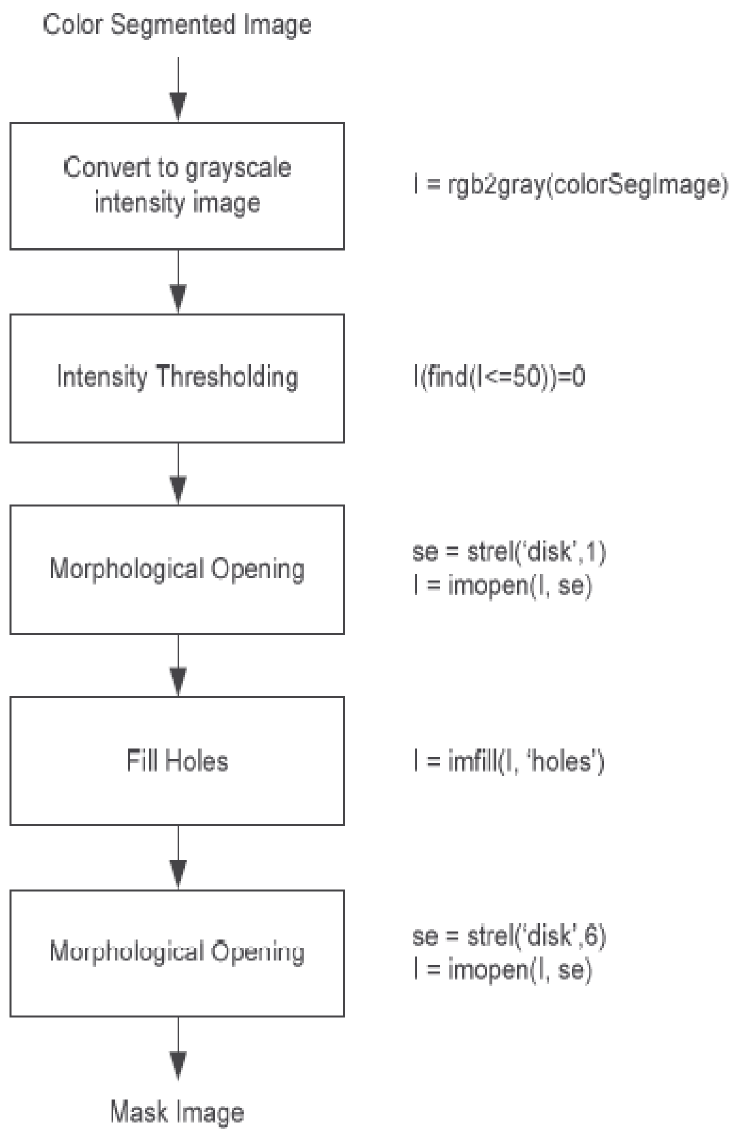


HSV

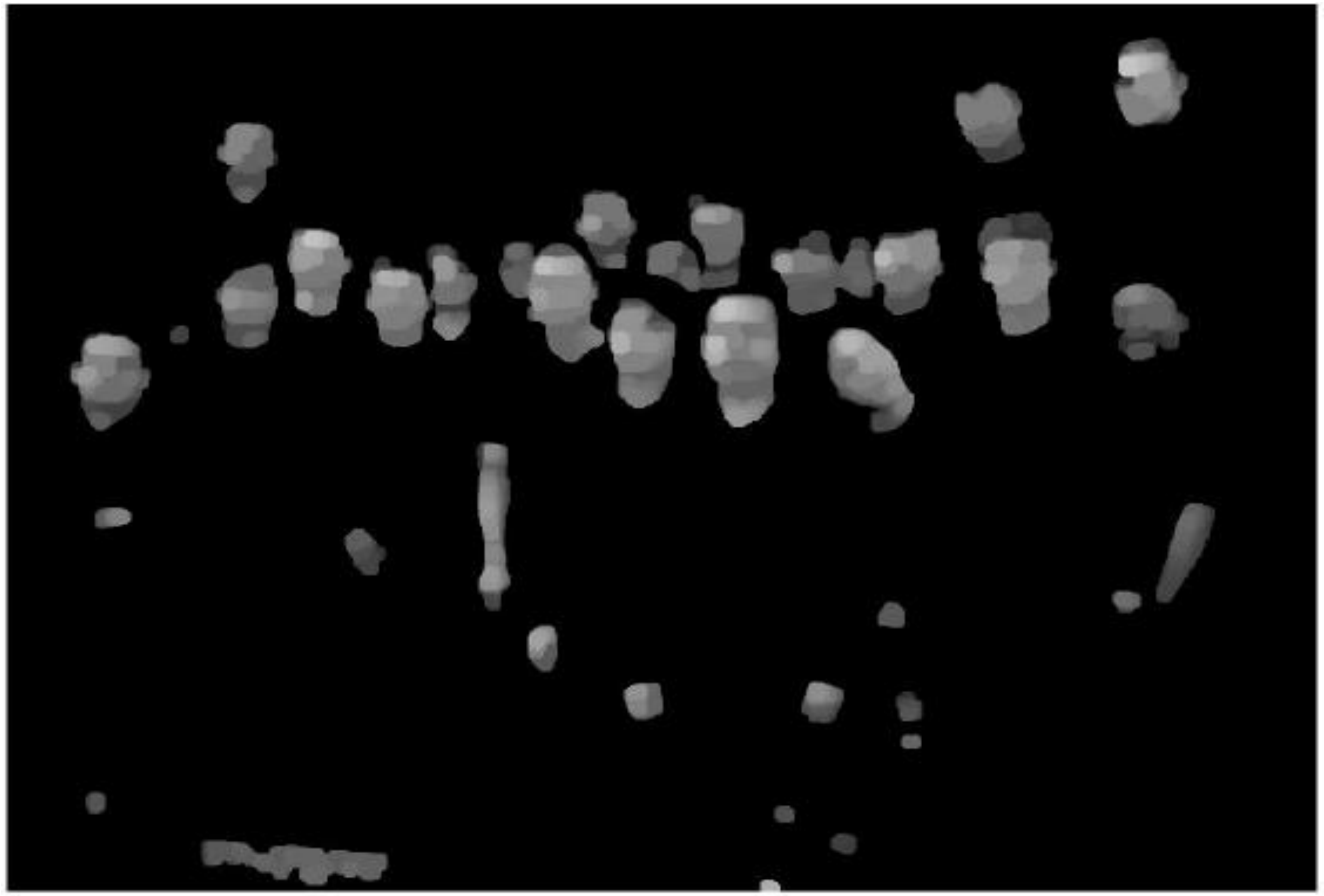

Hue 1
2000
1500
1000
500
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Saturation 1
400
300
200
100
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Morphological Processing



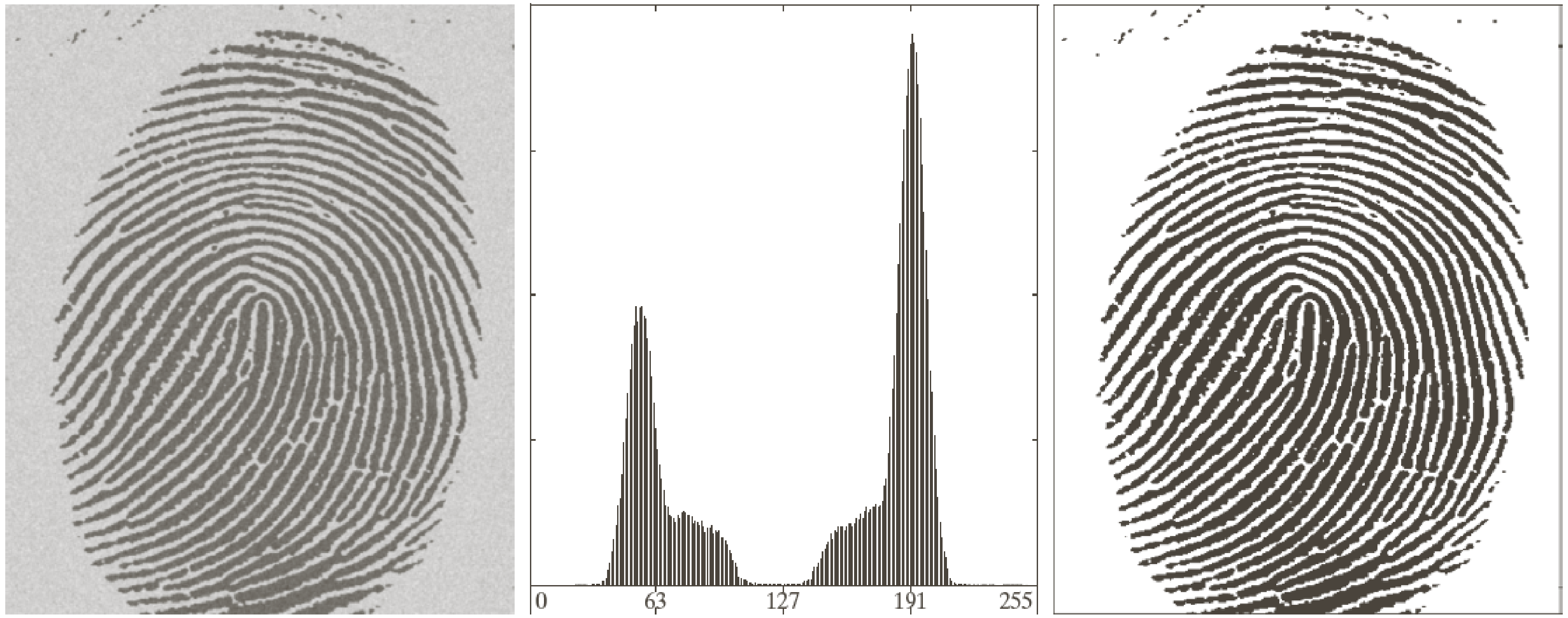
```
I = rgb2gray(colorSegImage)

I(find(I<=50))=0

se = strel('disk',1)
I = imopen(I, se)

I = imfill(I, 'holes')

se = strel('disk',6)
I = imopen(I, se)
```



Thresholding

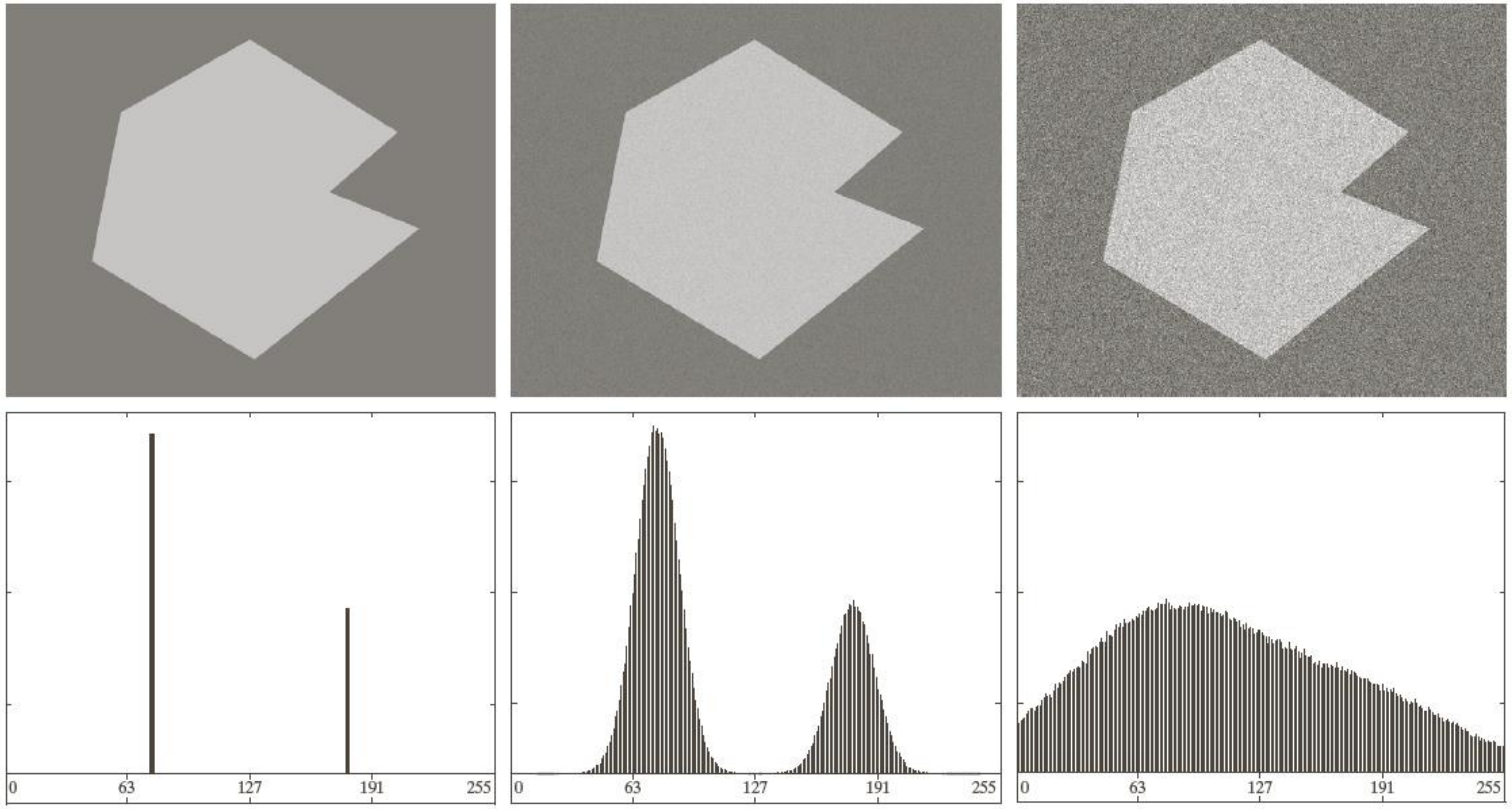
Thresholding

- Regions of an image can be separated by thresholding
- The peaks in the histogram must be well separated
  - Low noise, large signal-to-noise ratio
  - Can be improved by smoothing
- The illumination over the image should be close to uniform
- The reflectance properties of the objects and the background should be close to uniform



a b c

FIGURE 10.38 (a) Noisy fingerprint. (b) Histogram. (c) Segmented result using a global threshold (the border was added for clarity). (Original courtesy of the National Institute of Standards and Technology.)



a b c
d e f

FIGURE 10.36 (a) Noiseless 8-bit image. (b) Image with additive Gaussian noise of mean 0 and standard deviation of 10 intensity levels. (c) Image with additive Gaussian noise of mean 0 and standard deviation of 50 intensity levels. (d)–(f) Corresponding histograms.

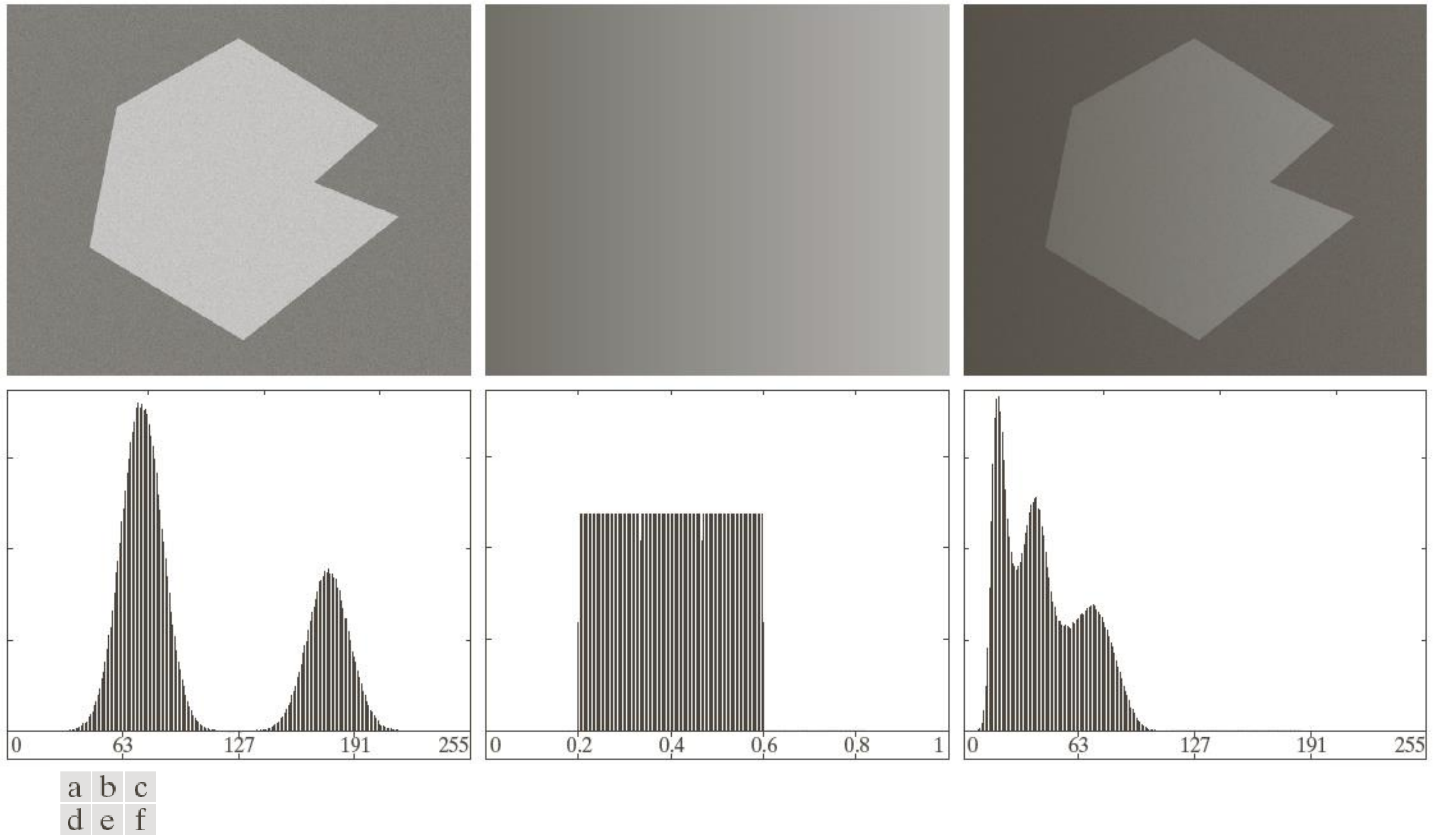


FIGURE 10.37 (a) Noisy image. (b) Intensity ramp in the range [0.2, 0.6]. (c) Product of (a) and (b). (d)–(f) Corresponding histograms.

Optimum Global Thresh

- Aka Otsu’s method.
- See graythresh in Matlab
- Histogram is a probability distrib: $p_i = n_i/MN$
- Global Average: m

$$m = \sum_{i=0}^{L-1} i p_i$$

- Thresholding at level k:

$$\omega(k) = \sum_{i=0}^{k} p_i, \quad \mu(k) = \sum_{i=k+1}^{L-1} p_i$$

- We would like to maximize the inter-class variance, i.e. the difference between ω(k) and μ(k)
- k is set to the value that maximize

$$\frac{\left(m\omega(k) - \mu(k)\right)^2}{\omega(k)\mu(k)}$$

Global Thresholding Challenges

- Noise
  - Noise causes a spread in the histogram distribution
  - Solution: apply localized smoothing
- Object is much smaller in size than the background
  - The peak corresponding to the object might be hidden in a global histogram
  - Solution1: use edge detection to limit the histogram
  - Solution2: use localized histograms
- Non-uniform intensity of object/background
  - Non uniform intensity causes loss of global contrast
  - Solution: Sub-divide image and apply variable localized thresholding to various sections of the image.

Noise Solution: Smoothing

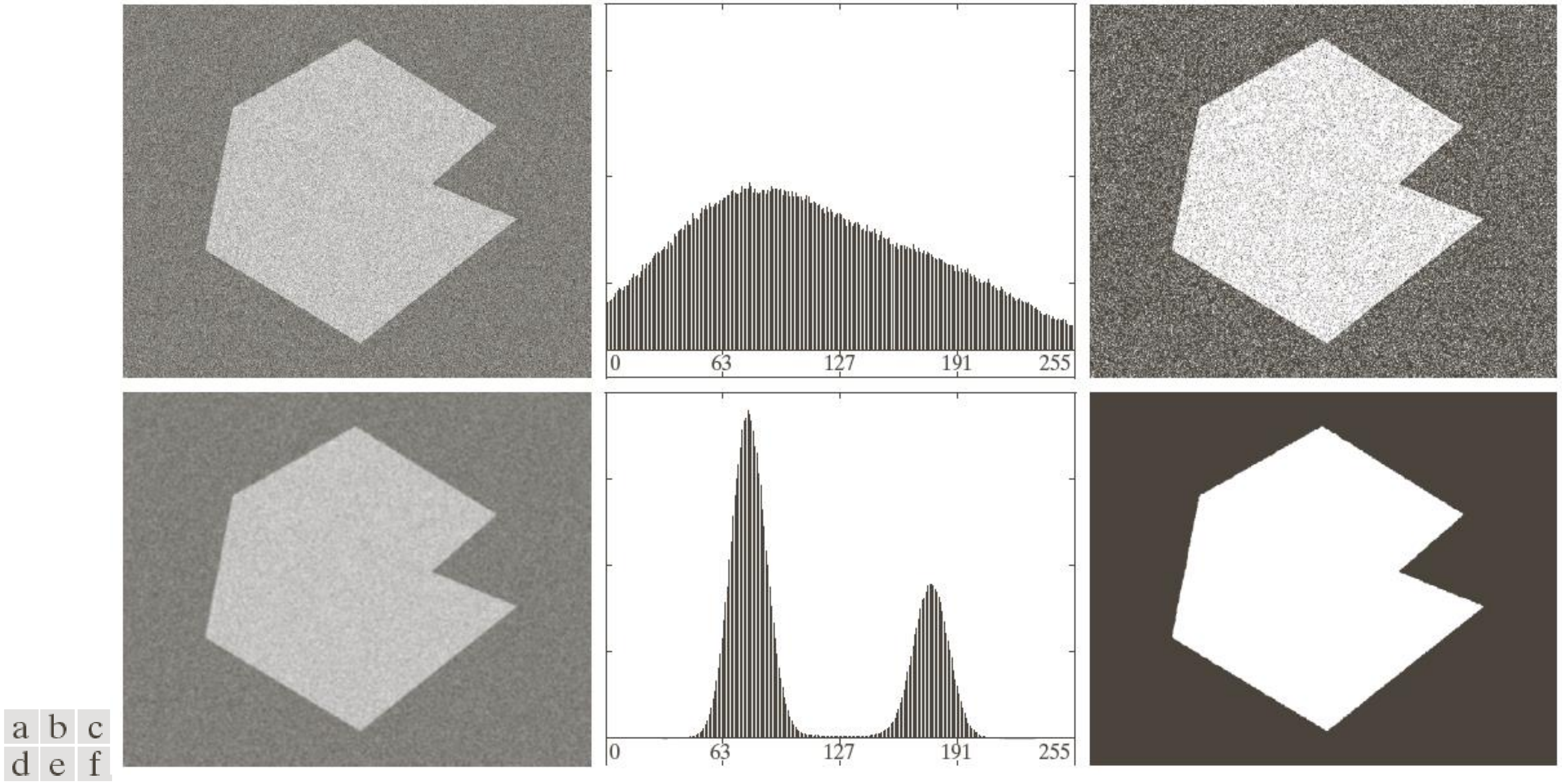


FIGURE 10.40 (a) Noisy image from Fig. 10.36 and (b) its histogram. (c) Result obtained using Otsu's method. (d) Noisy image smoothed using a 5 × 5 averaging mask and (e) its histogram. (f) Result of thresholding using Otsu's method.

Small Object Problem

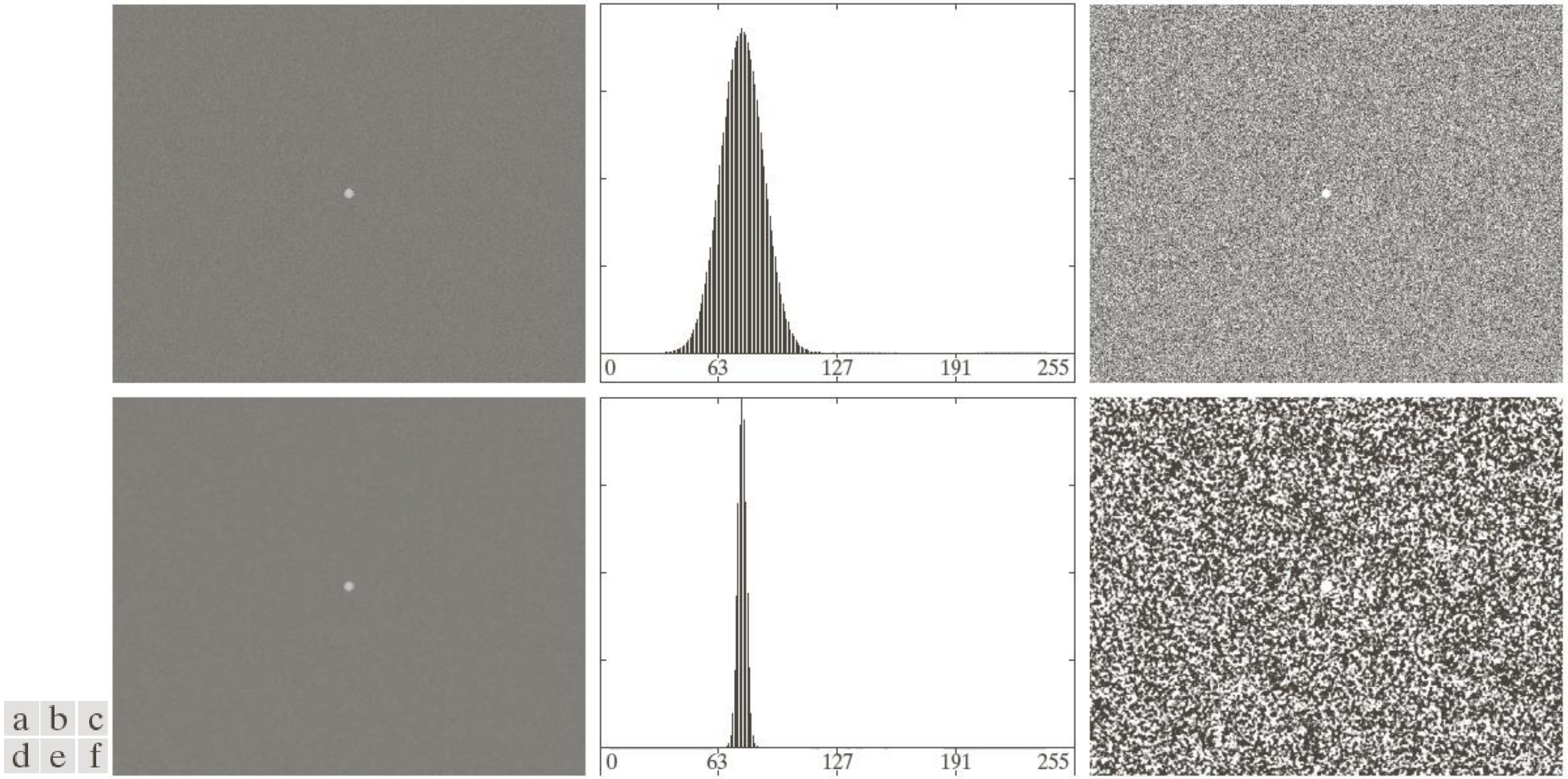


FIGURE 10.41 (a) Noisy image and (b) its histogram. (c) Result obtained using Otsu's method. (d) Noisy image smoothed using a 5×5 averaging mask and (e) its histogram. (f) Result of thresholding using Otsu's method. Thresholding failed in both cases.

Small Object Solution: Edge Detection

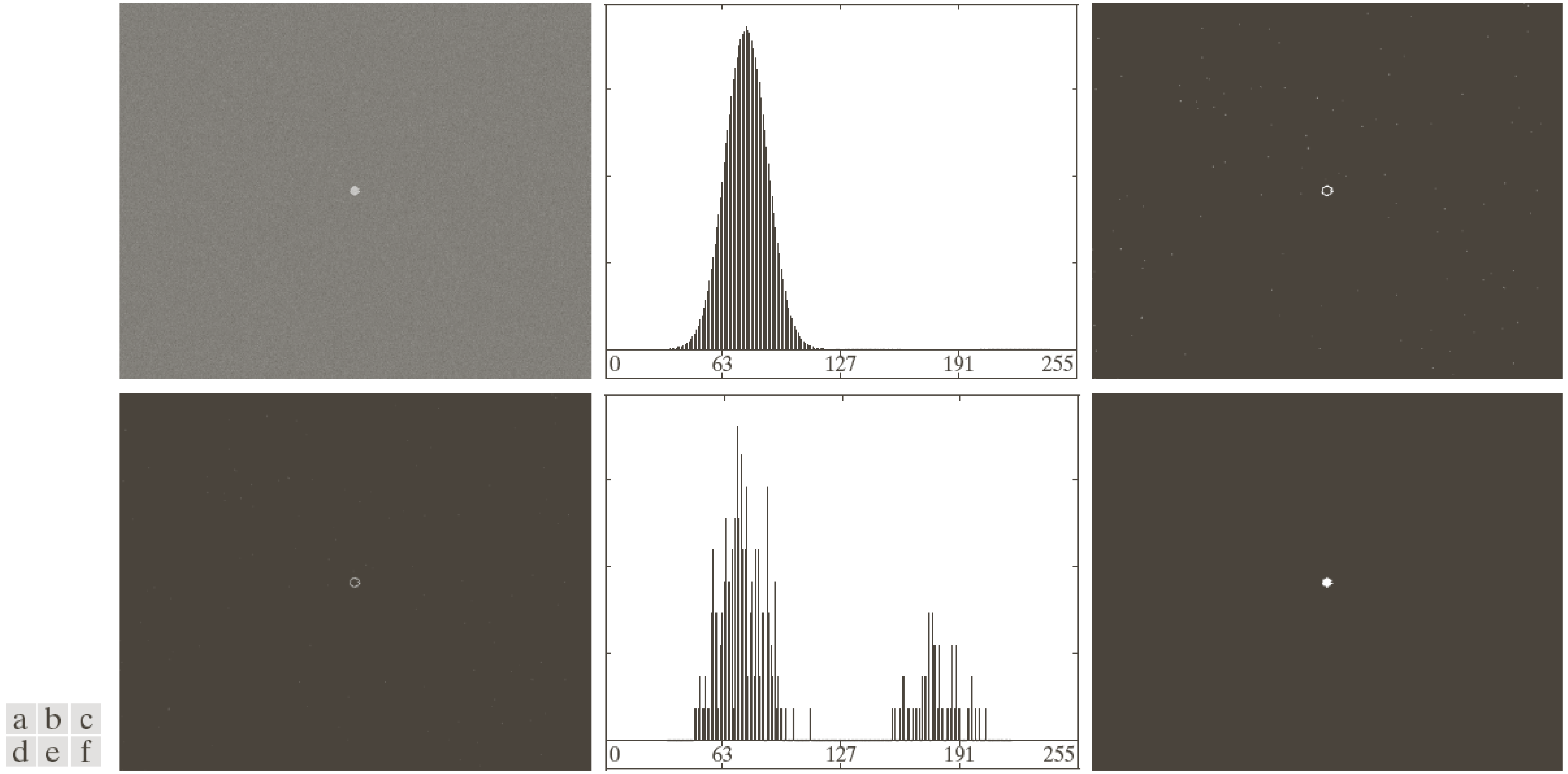


FIGURE 10.42 (a) Noisy image from Fig. 10.41(a) and (b) its histogram. (c) Gradient magnitude image thresholded at the 99.7 percentile. (d) Image formed as the product of (a) and (c). (e) Histogram of the nonzero pixels in the image in (d). (f) Result of segmenting image (a) with the Otsu threshold based on the histogram in (e). The threshold was 134, which is approximately midway between the peaks in this histogram.

Small Object Solution: Localized Thresh

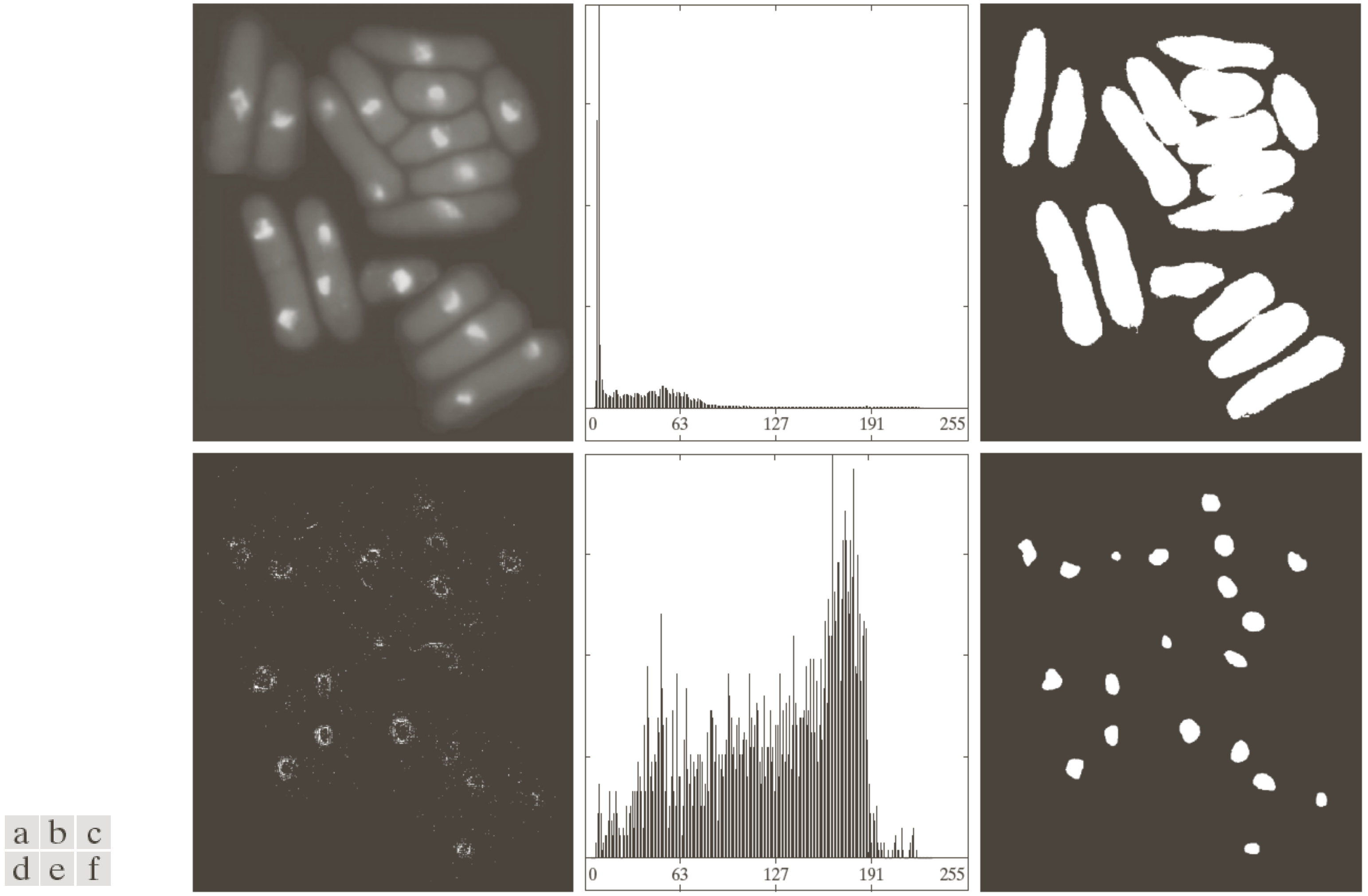
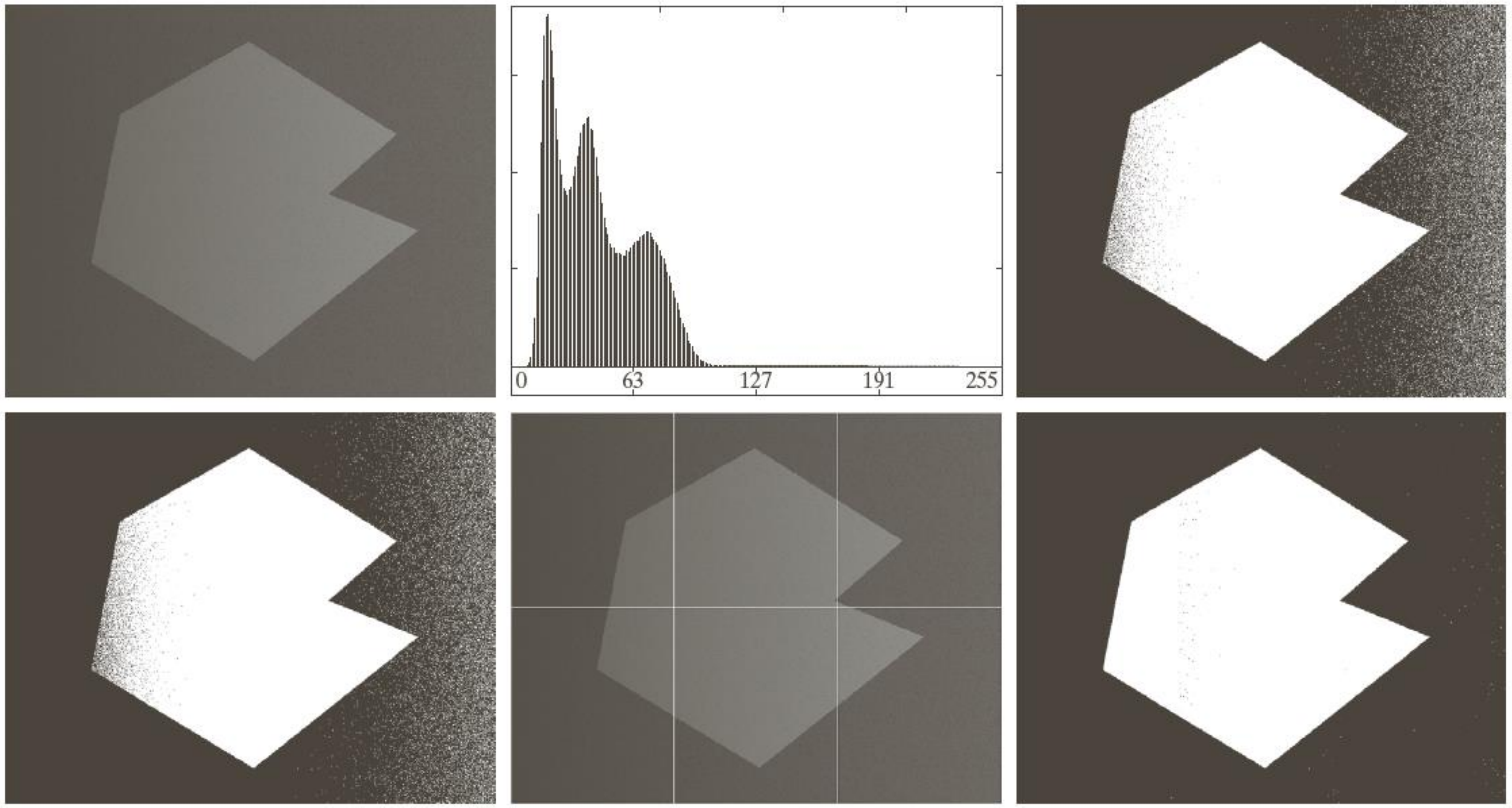


FIGURE 10.43 (a) Image of yeast cells. (b) Histogram of (a). (c) Segmentation of (a) with Otsu's method using the histogram in (b). (d) Thresholded absolute Laplacian. (e) Histogram of the nonzero pixels in the product of (a) and (d). (f) Original image thresholded using Otsu's method based on the histogram in (e). (Original image courtesy of Professor Susan L. Forsburg, University of Southern California.)

Nonuniform Intensities Solution: Partitioning

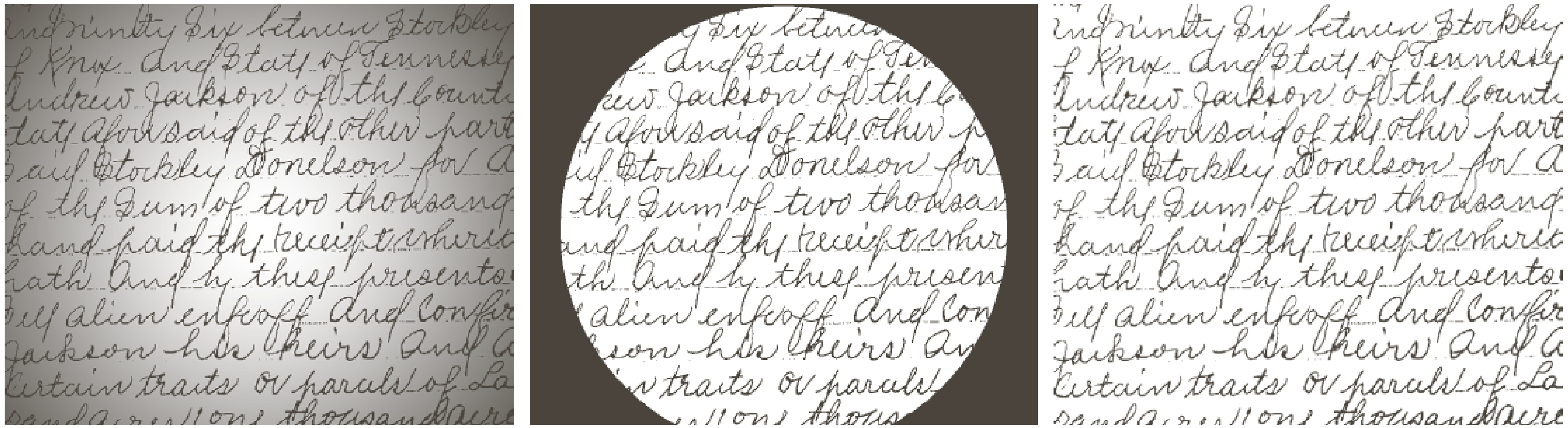


a b c
d e f

FIGURE 10.46 (a) Noisy, shaded image and (b) its histogram. (c) Segmentation of (a) using the iterative global algorithm from Section 10.3.2. (d) Result obtained using Otsu's method. (e) Image subdivided into six subimages. (f) Result of applying Otsu's method to each subimage individually.

Moving Averages

- Computing averages along scanned zig-zag lines
  - Useful for scanned documents
- Mean at point k is: $m(k+1) = 1/n \sum z_i = m(k) + 1/n(z_{k+1} - z_{k-n})$
  - z_k is the intensity at k
  - n is average size
- Segmentation threshold: $T_{xy} = bm(k+1)$
  - b = constant. m(k+1) = moving average



a b c

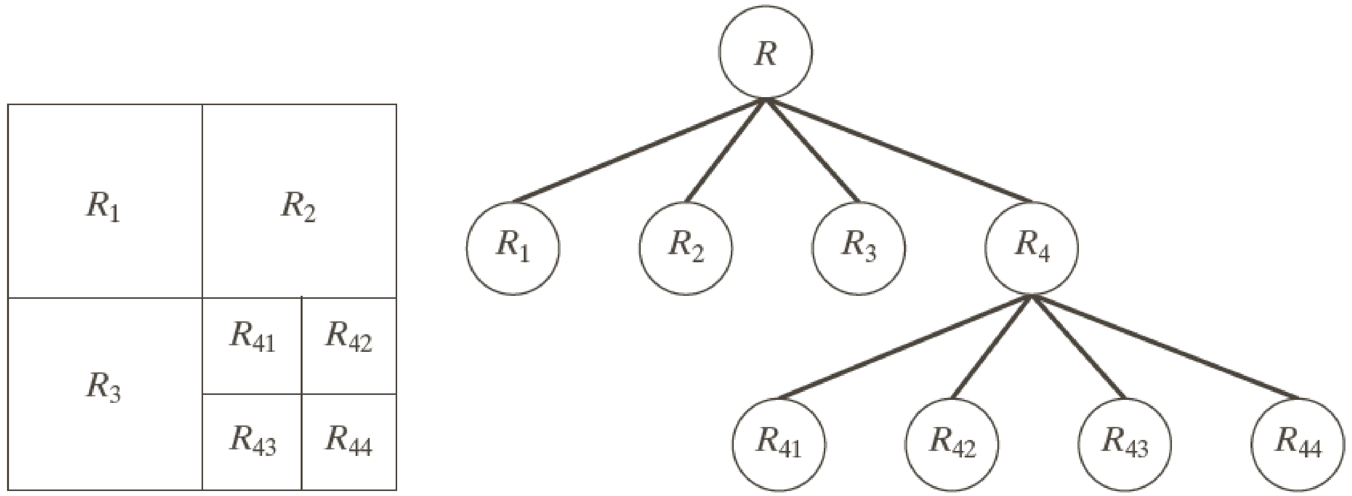
FIGURE 10.49 (a) Text image corrupted by spot shading. (b) Result of global thresholding using Otsu's method. (c) Result of local thresholding using moving averages.

Region Based Segmentation

- Objective: partition an image into regions
- R : entire image
- Segmentation is a process that partitions R into n subregions, R_1, R_2, ..., R_n, such that
  - union of is $R_i = R$.
  - R_i is connected region
  - intersection of R_i and R_j is null
    - P(R_i and R_j) = FALSE
  - $P(R_i)$ = TRUE : logical predicate

Region Splitting/Merging

- Region growing is a bottom-up approach, region splitting and merging is a top-down procedure.
- It starts with the whole image and subdivides it recursively based on a given predicate Q.
- Quad tree: popular region split/merge
  - Split into Ri any region for which Q(Ri) = false
  - When no further splitting is possible, merge adjacent R_j and R_k for which Q$(R_j \cup R_k) =$ *TRUE*
  - Stop if no more merging is possible
- Example: NASA x-ray supernova image
  - Q=true if $(\sigma > a)$ & $(0<m<b)$
  - a,b are constants


R_1
R_2
R_3
R_{41}
R_{42}
R_{43}
R_{44}
R
R_1
R_2
R_3
R_4
R_{41}
R_{42}
R_{43}
R_{44}

a	b
c	d

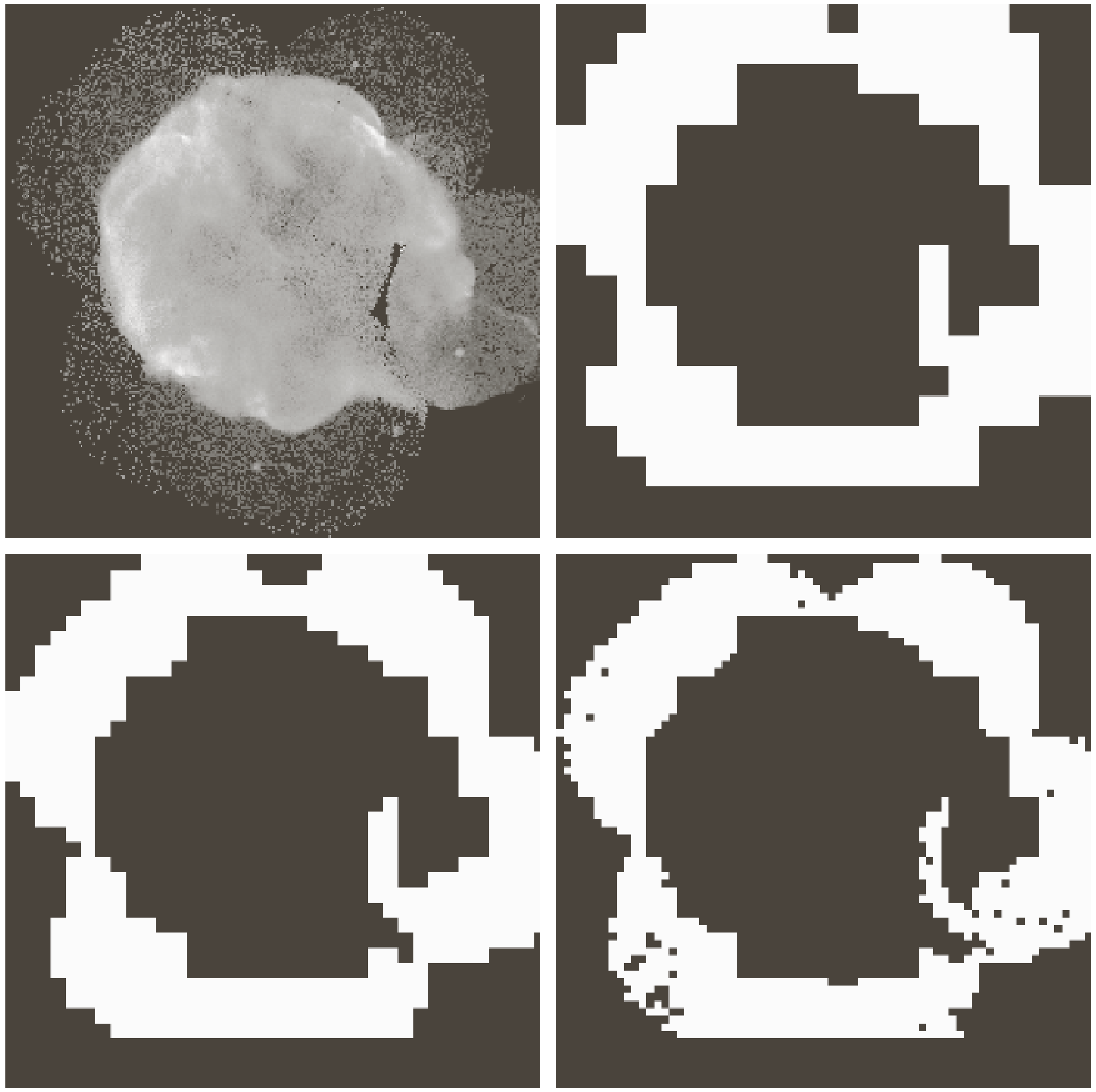
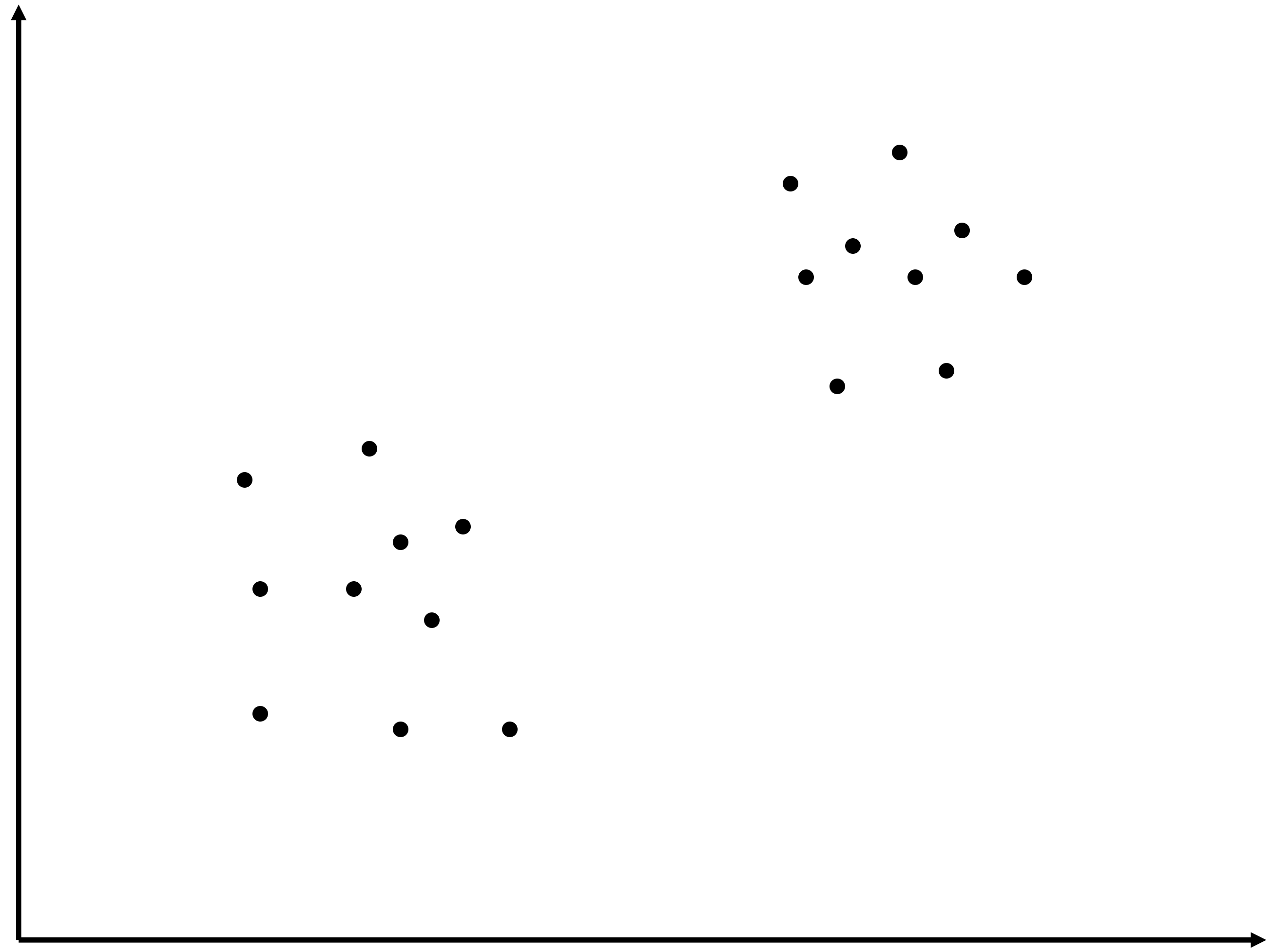


FIGURE 10.53 (a) Image of the Cygnus Loop supernova, taken in the X-ray band by NASA's Hubble Telescope. (b)–(d) Results of limiting the smallest allowed quadregion to sizes of 32×32, 16×16, and 8×8 pixels, respectively. (Original image courtesy of NASA.)

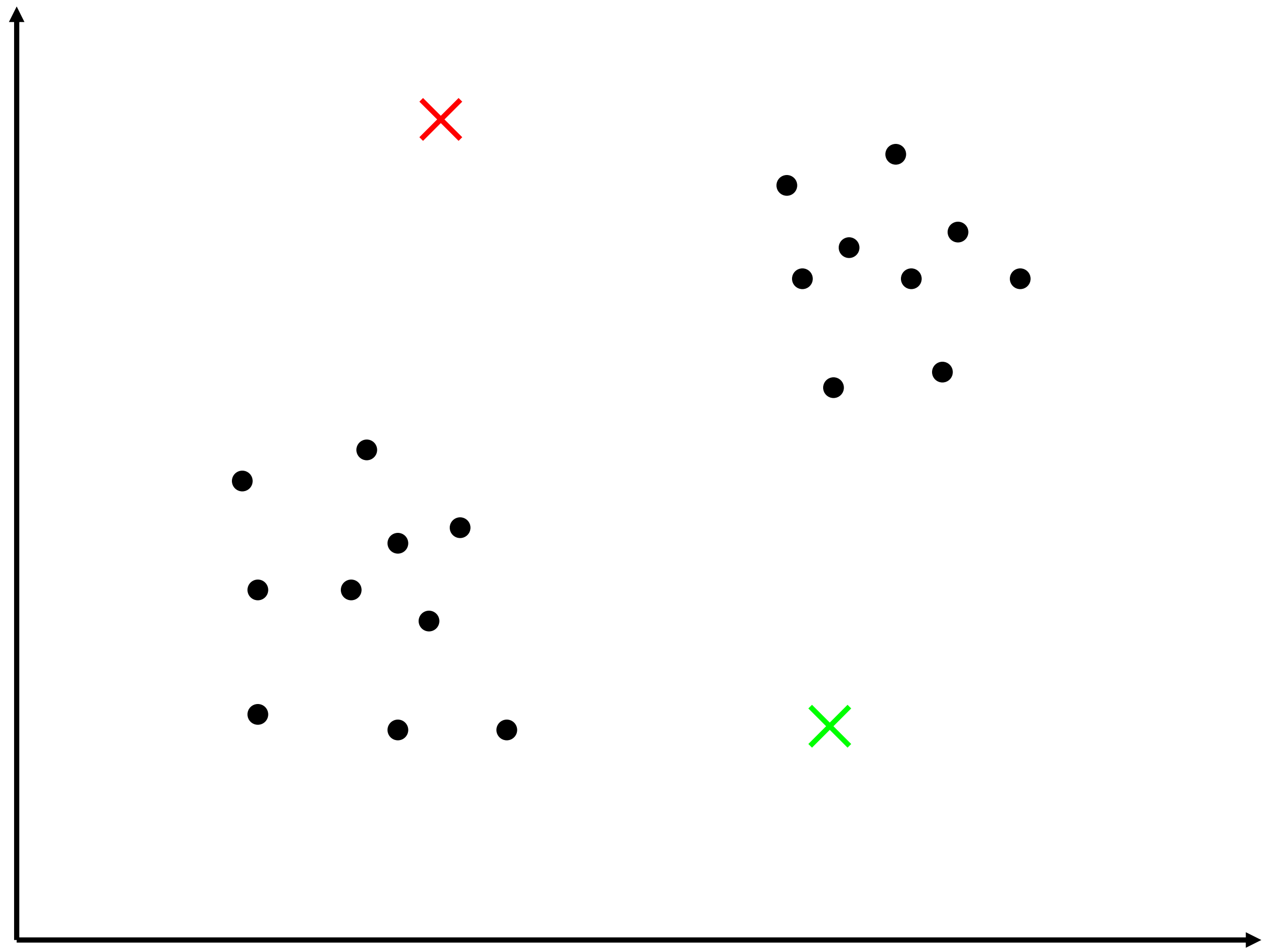
K means Clustering

1. Partition the data points into K clusters randomly. Find the centroids of each cluster.
2. For each data point:
   - Calculate the distance from the data point to each cluster.
   - Assign the data point to the closest cluster.
3. Recompute the centroid of each cluster.
4. Repeat steps 2 and 3 until there is no further change in the assignment of data points (or in the centroids).

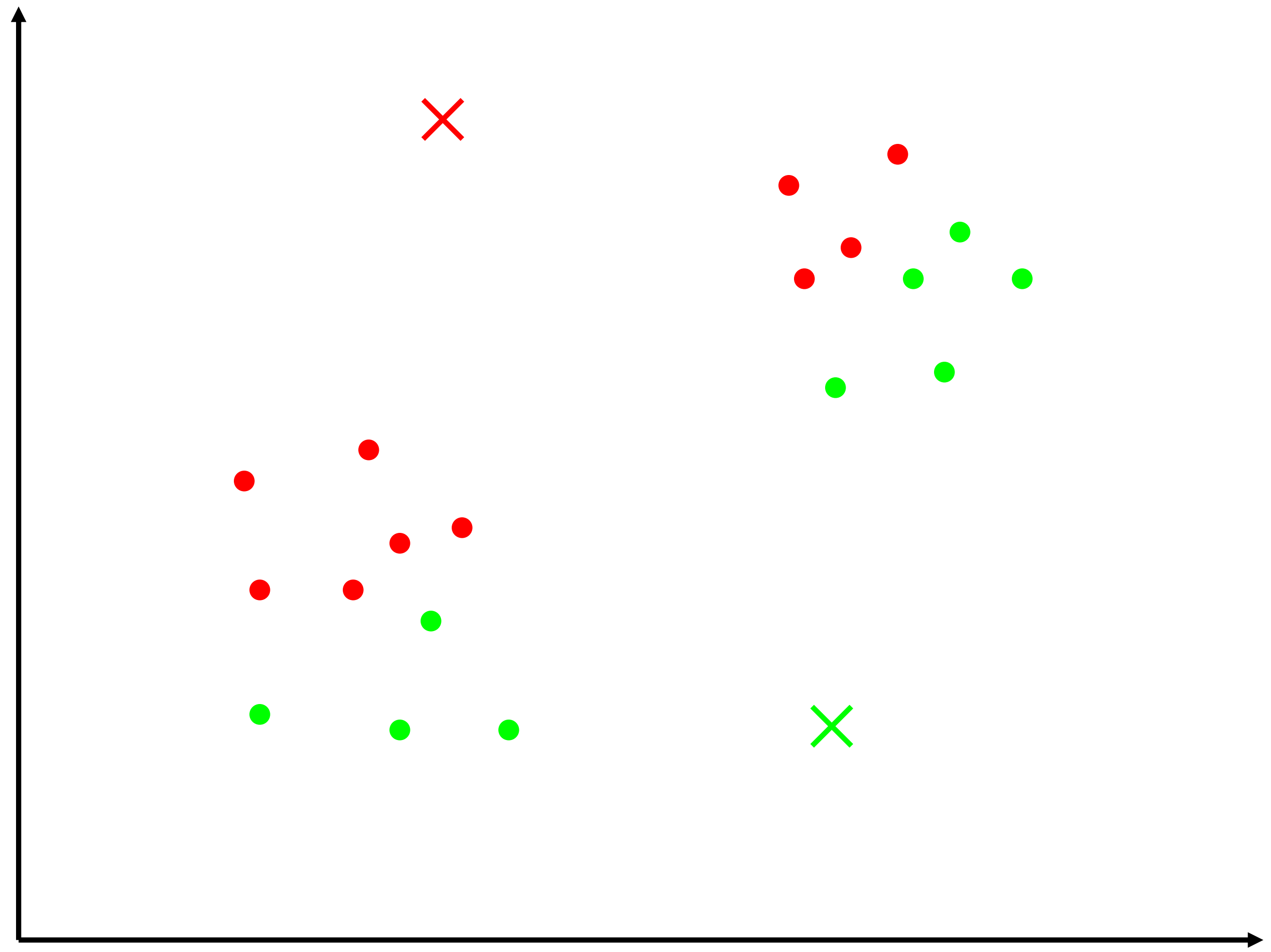
K-Means Clustering



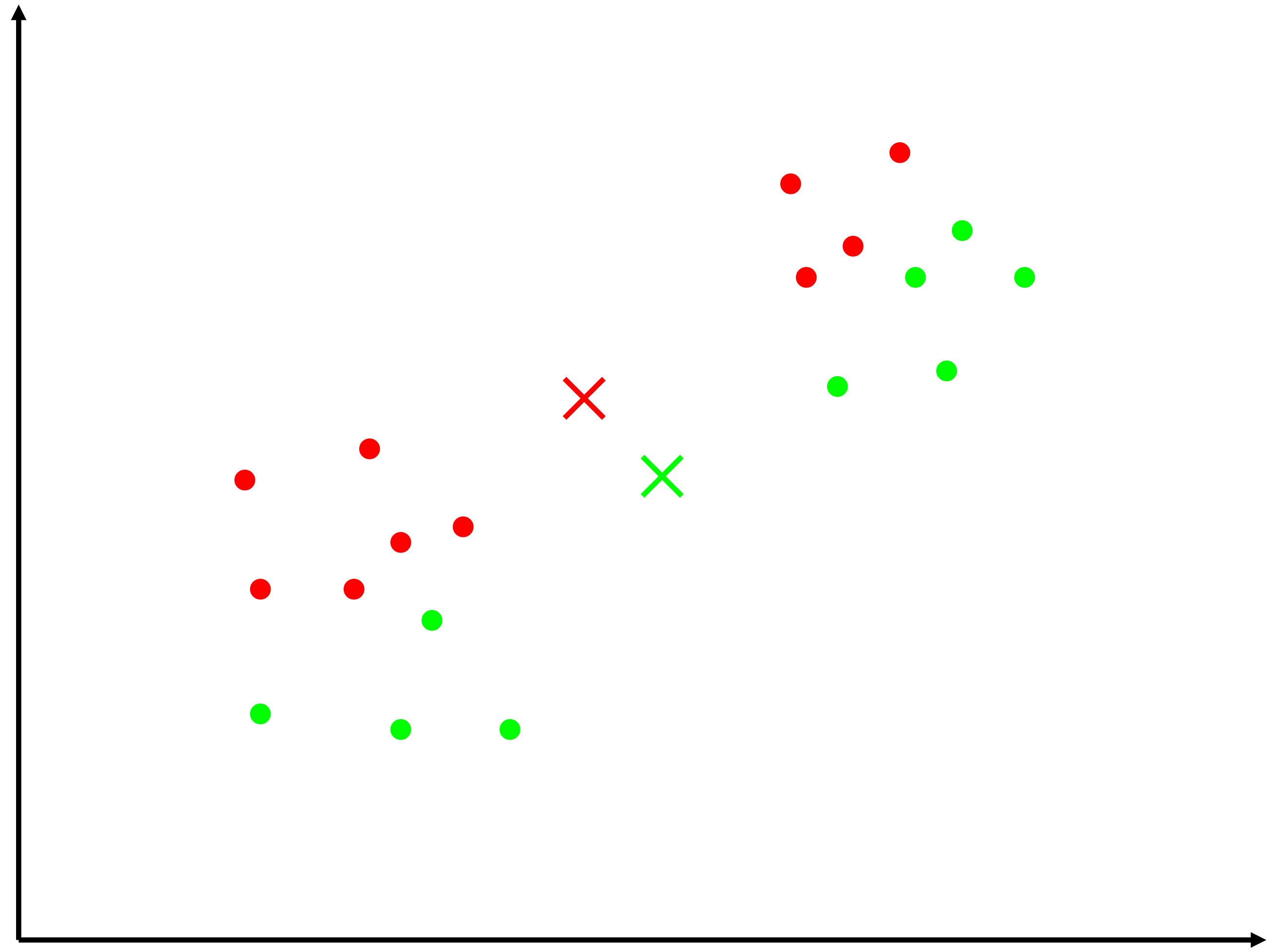
K-Means Clustering



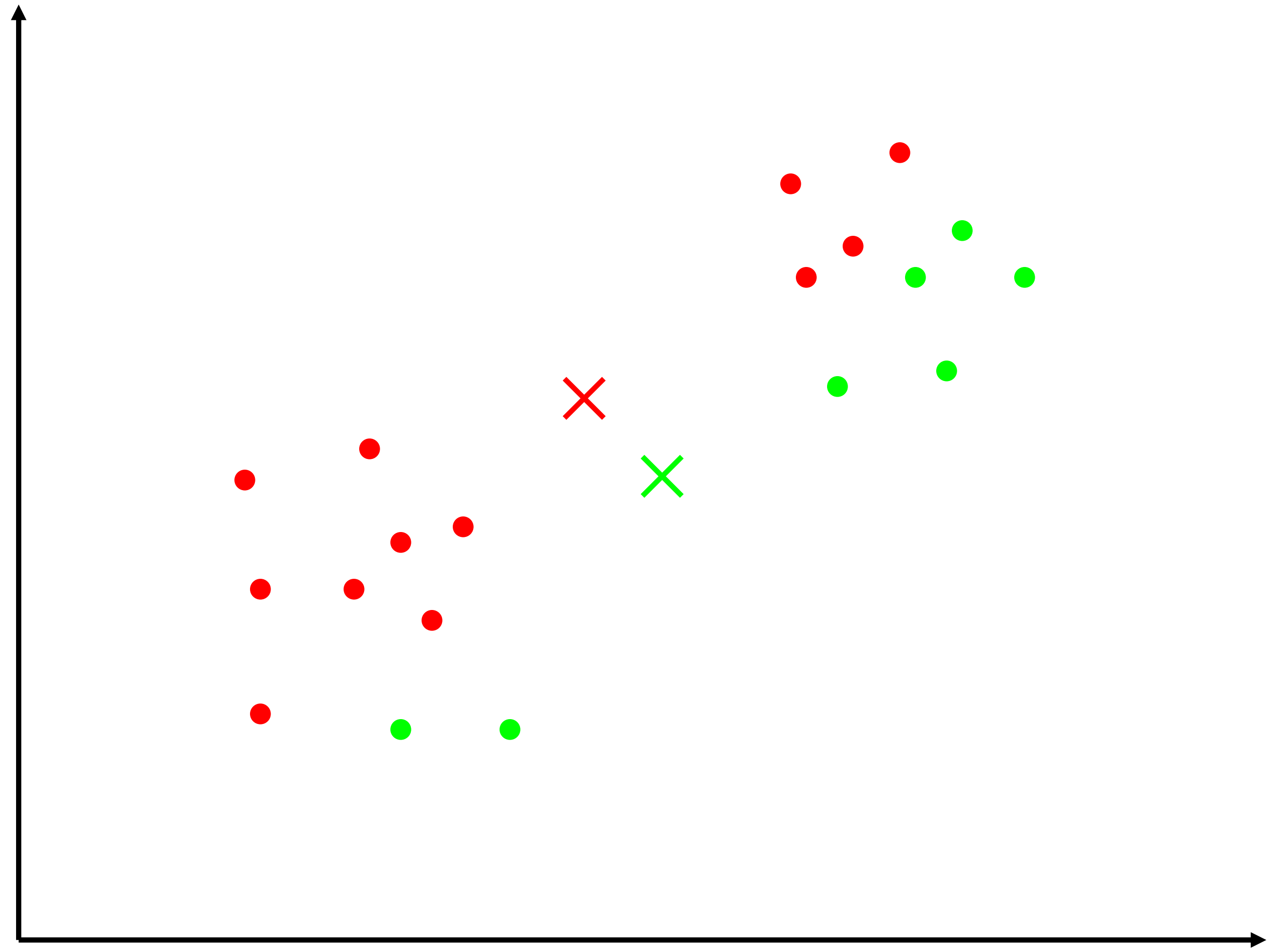
K-Means Clustering



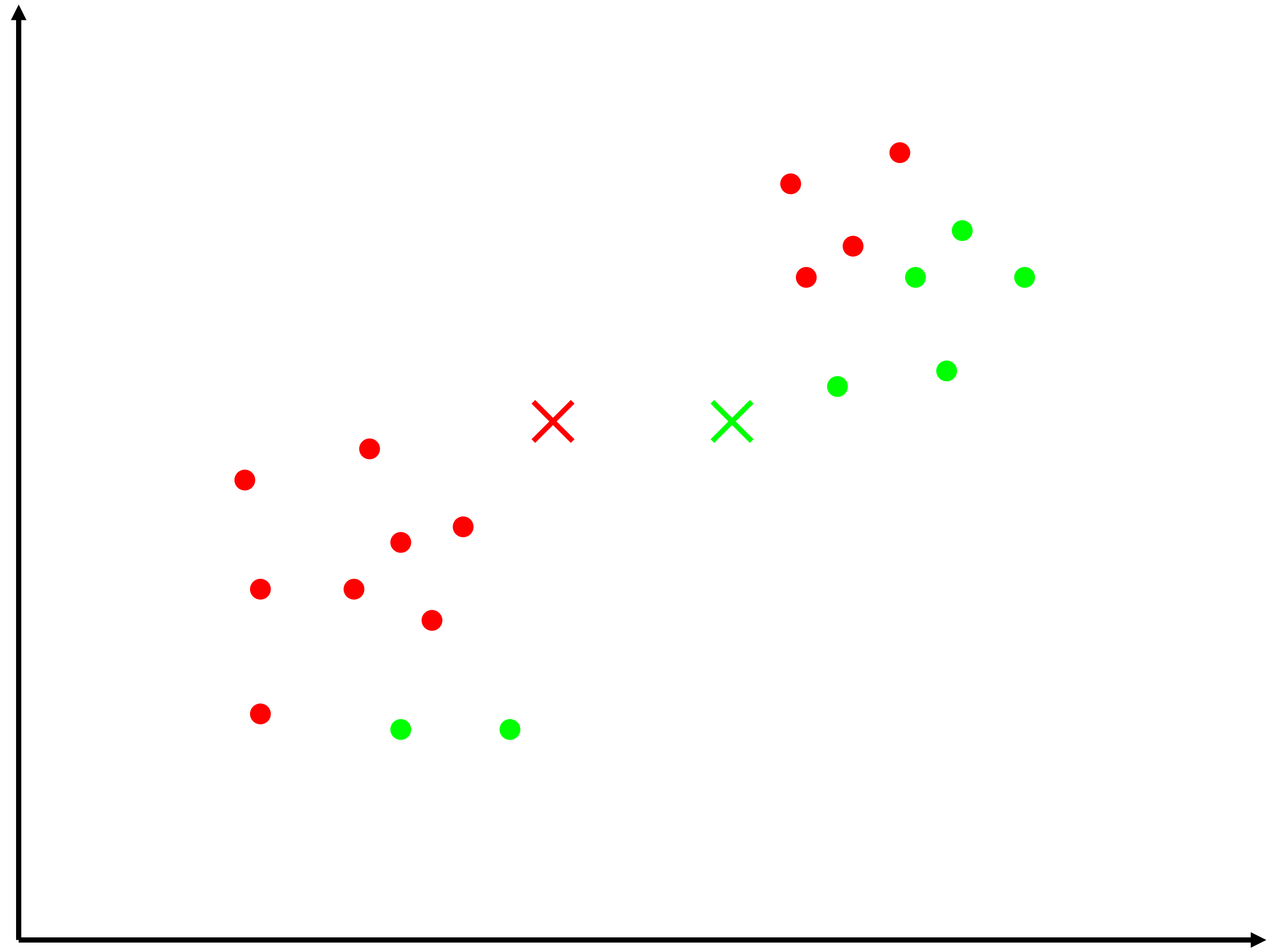
K-Means Clustering



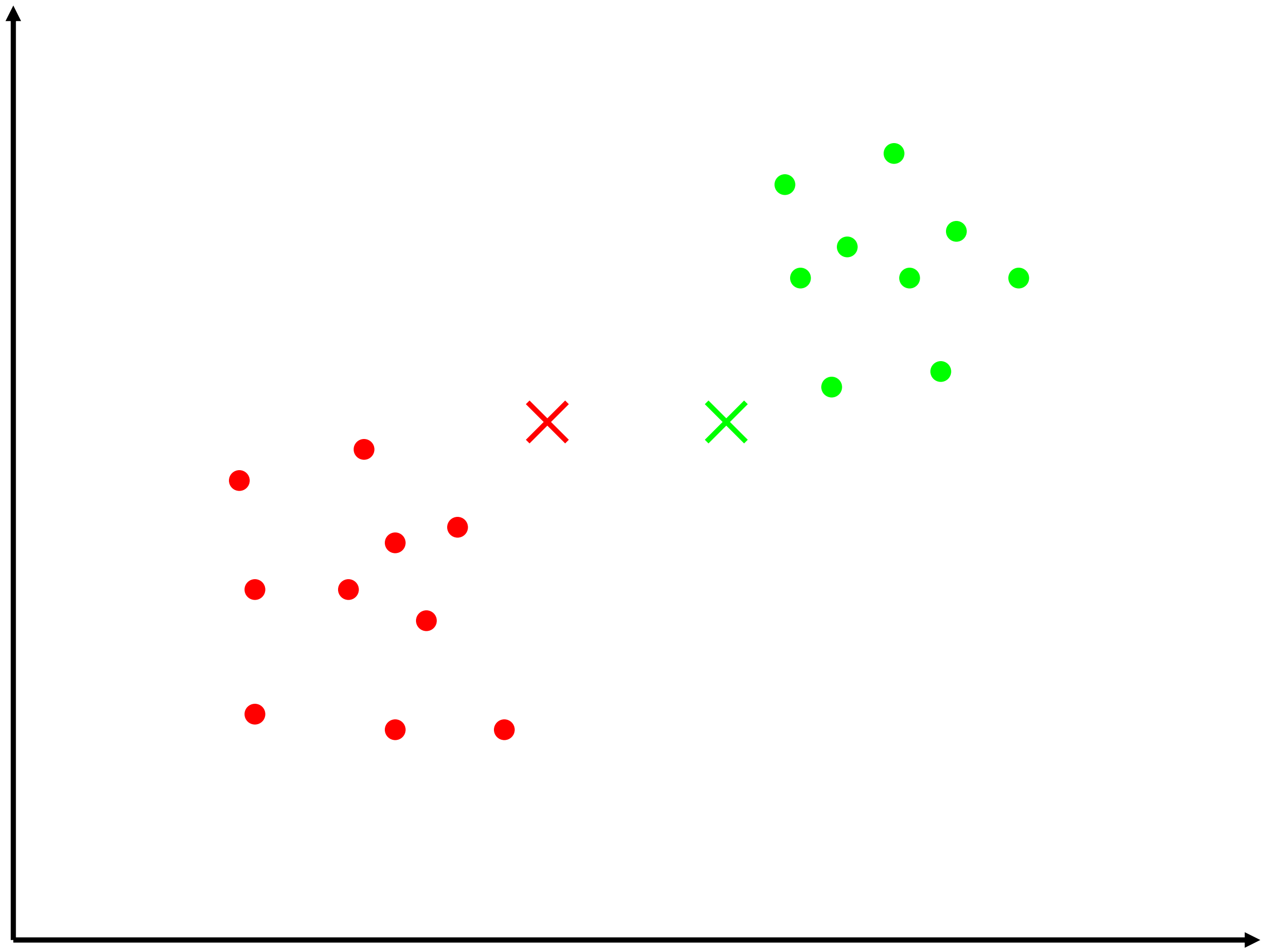
K-Means Clustering



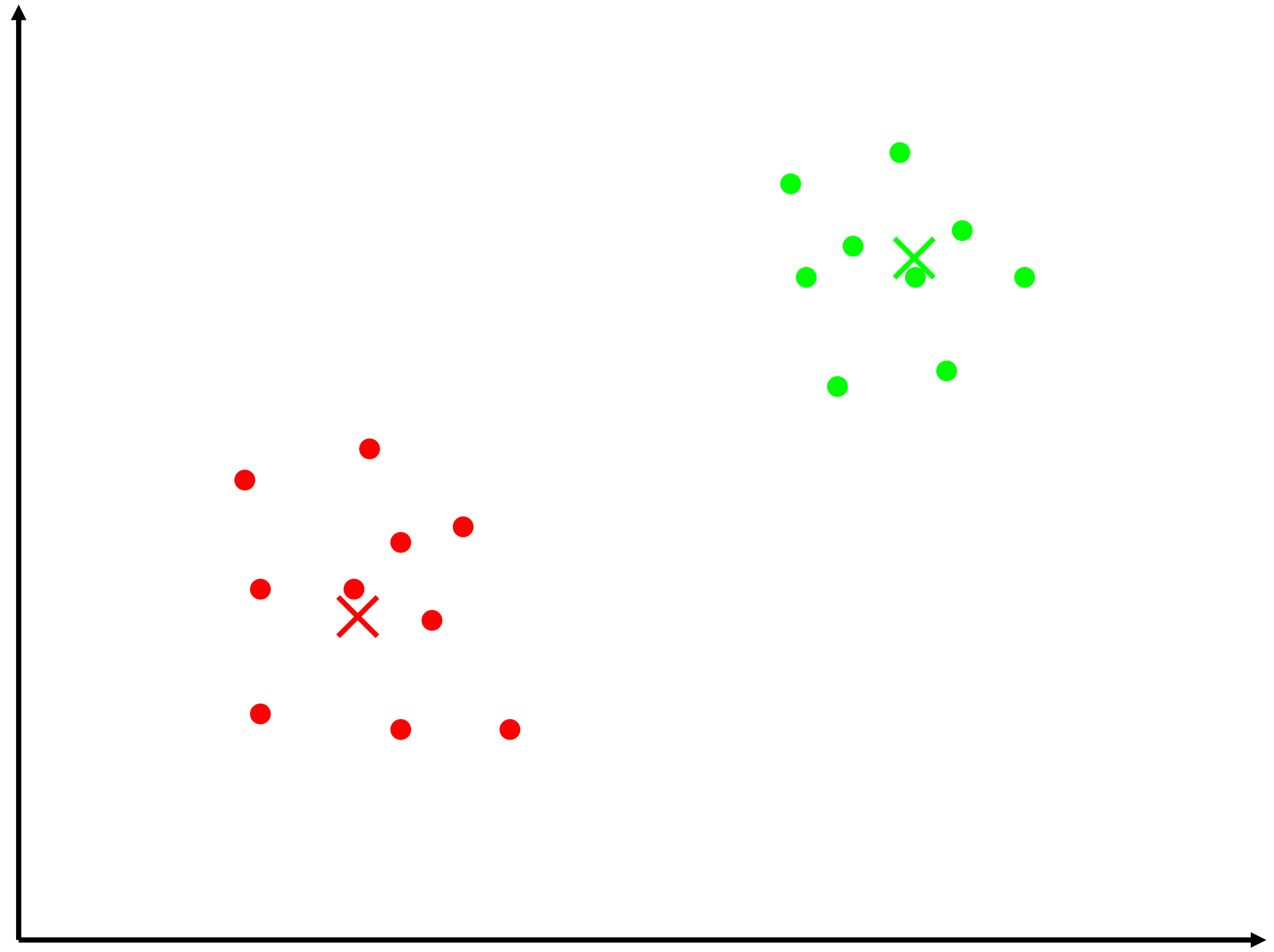
K-Means Clustering



K-Means Clustering



K-Means Clustering



K-means Algorithm

- An automated clustering technique
  - The number of clusters K must be supplied
  - Works by determining means of data clusters (centroids)
- Steps
  1. Set k
  2. Initialize centroids μ_1, .. μ_k, randomly
  3. For each data point, determine the centroid closest to it:

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

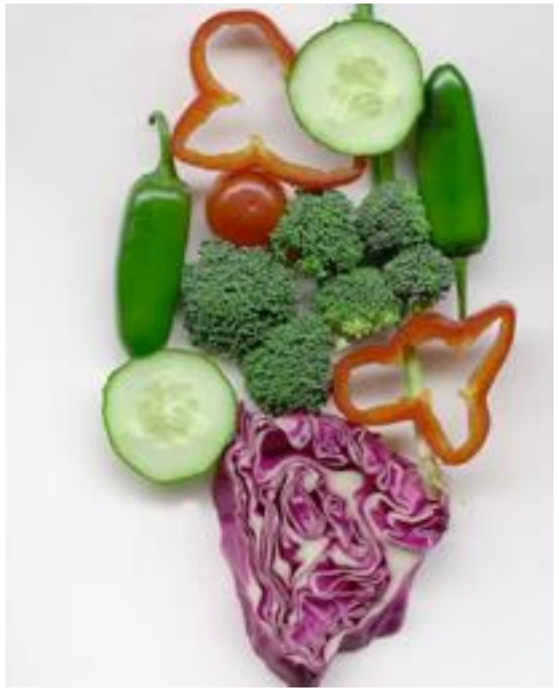
  4. Compute new centroids:

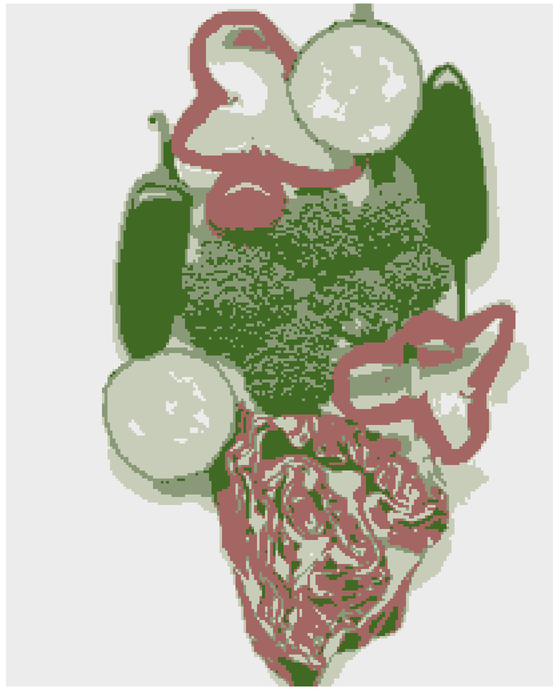
$$\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$

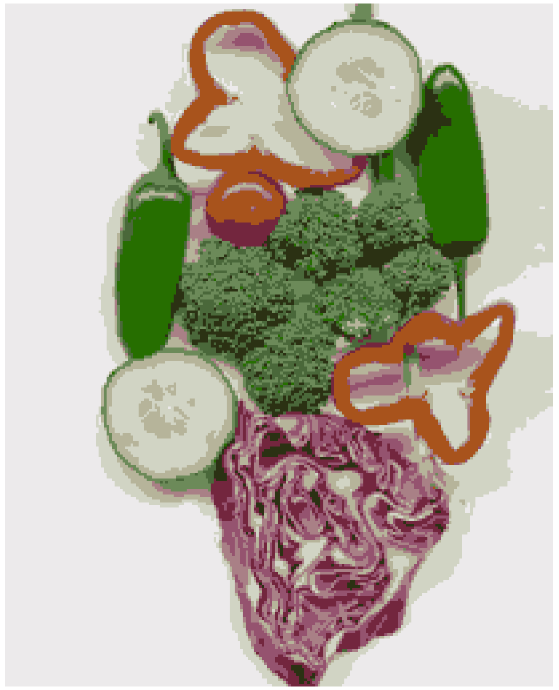
  5. Repeat 3-5 until convergence

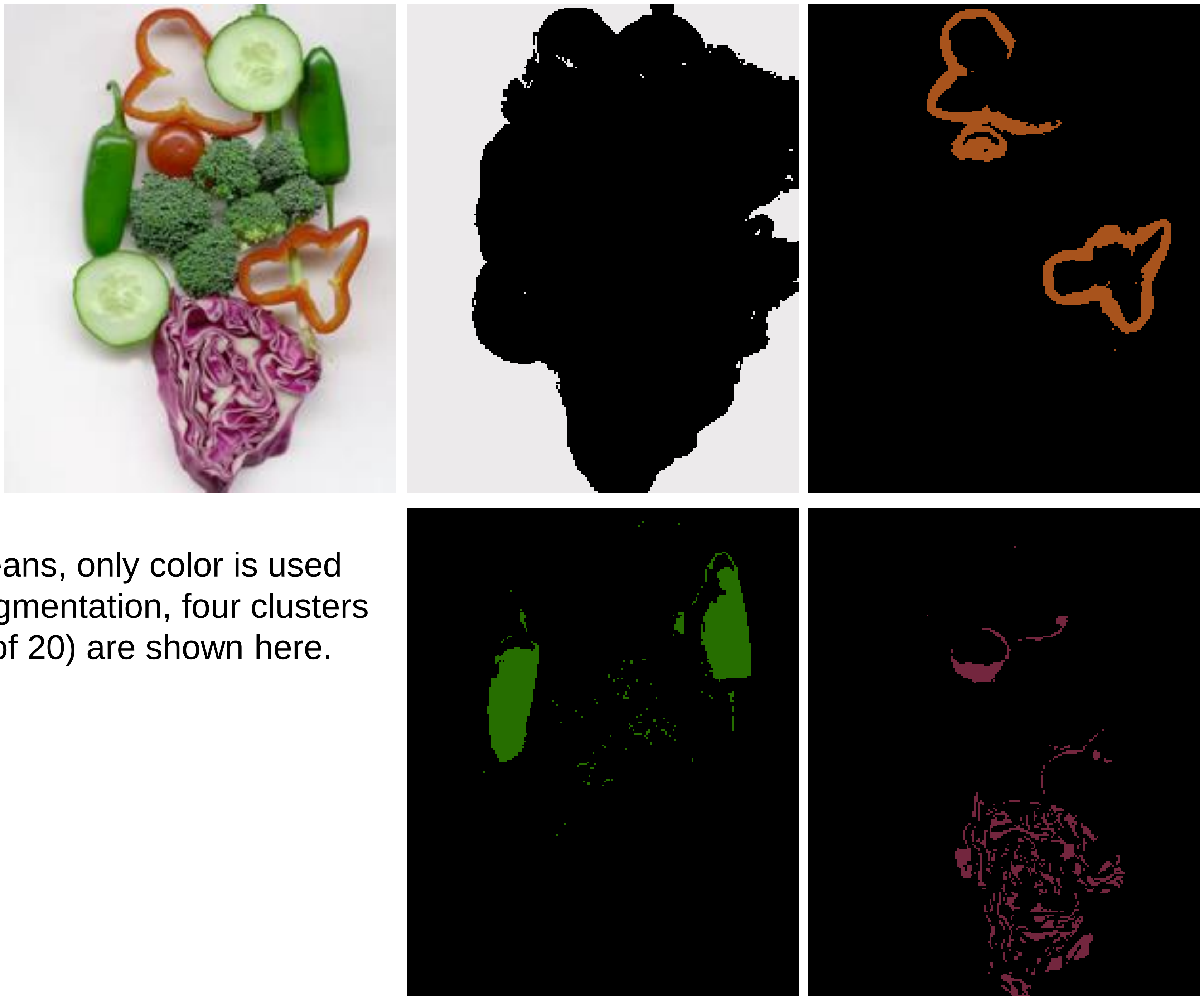
K-Means Clustering

- Example

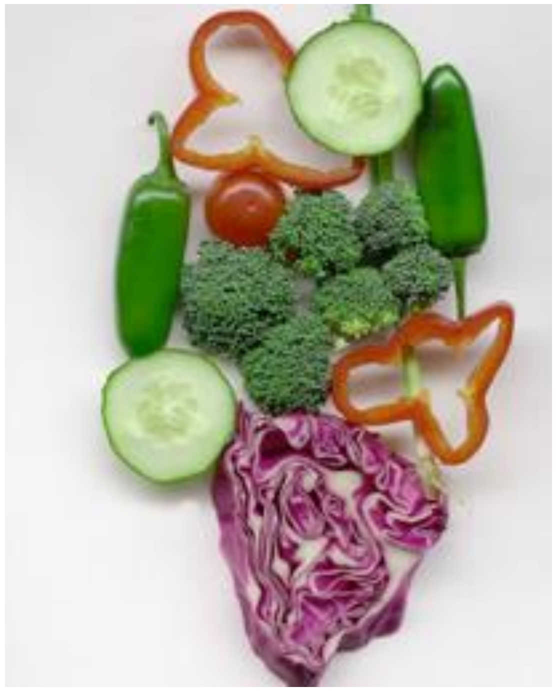

Original

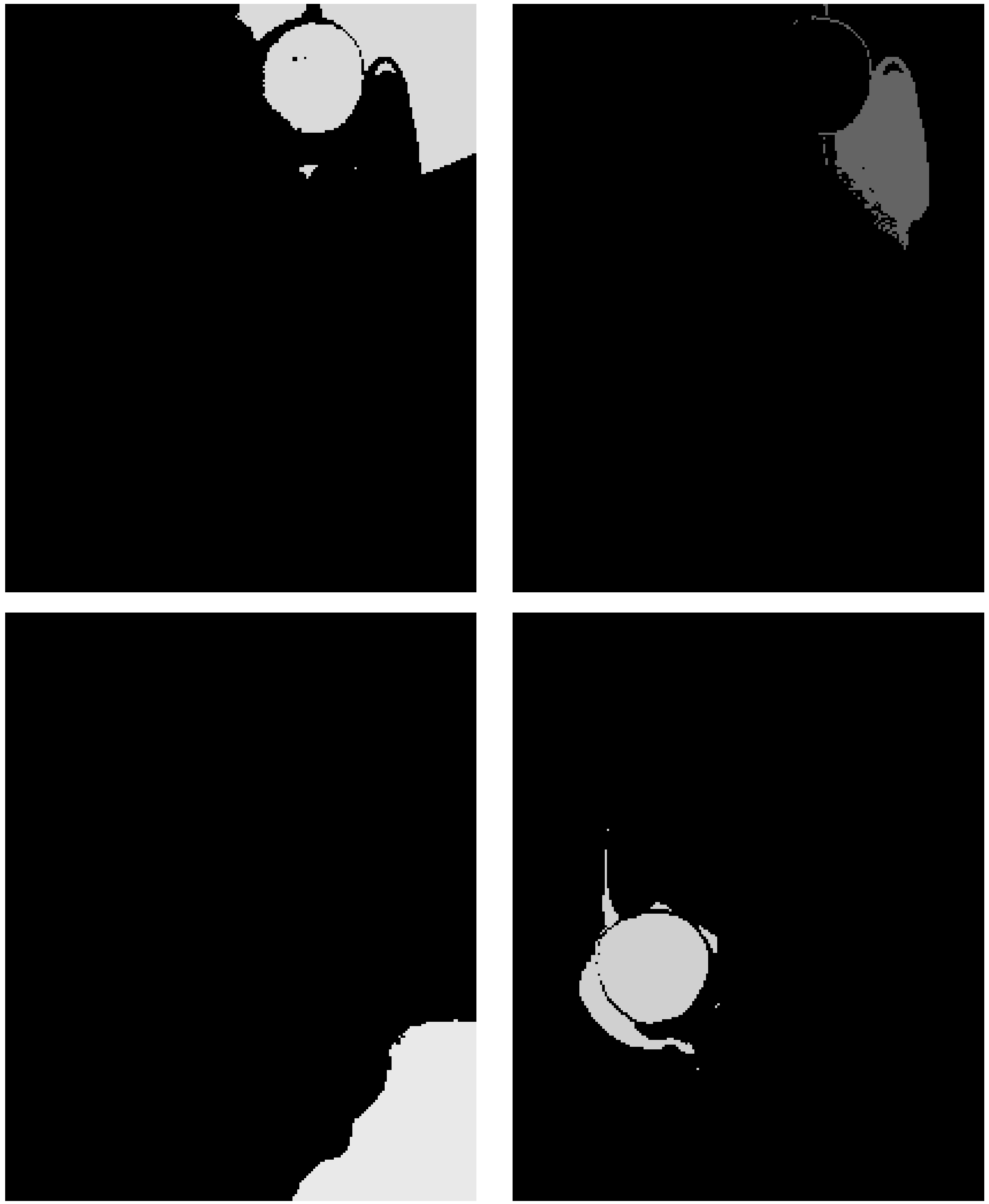

K=5


K=11



K-means, only color is used in segmentation, four clusters (out of 20) are shown here.

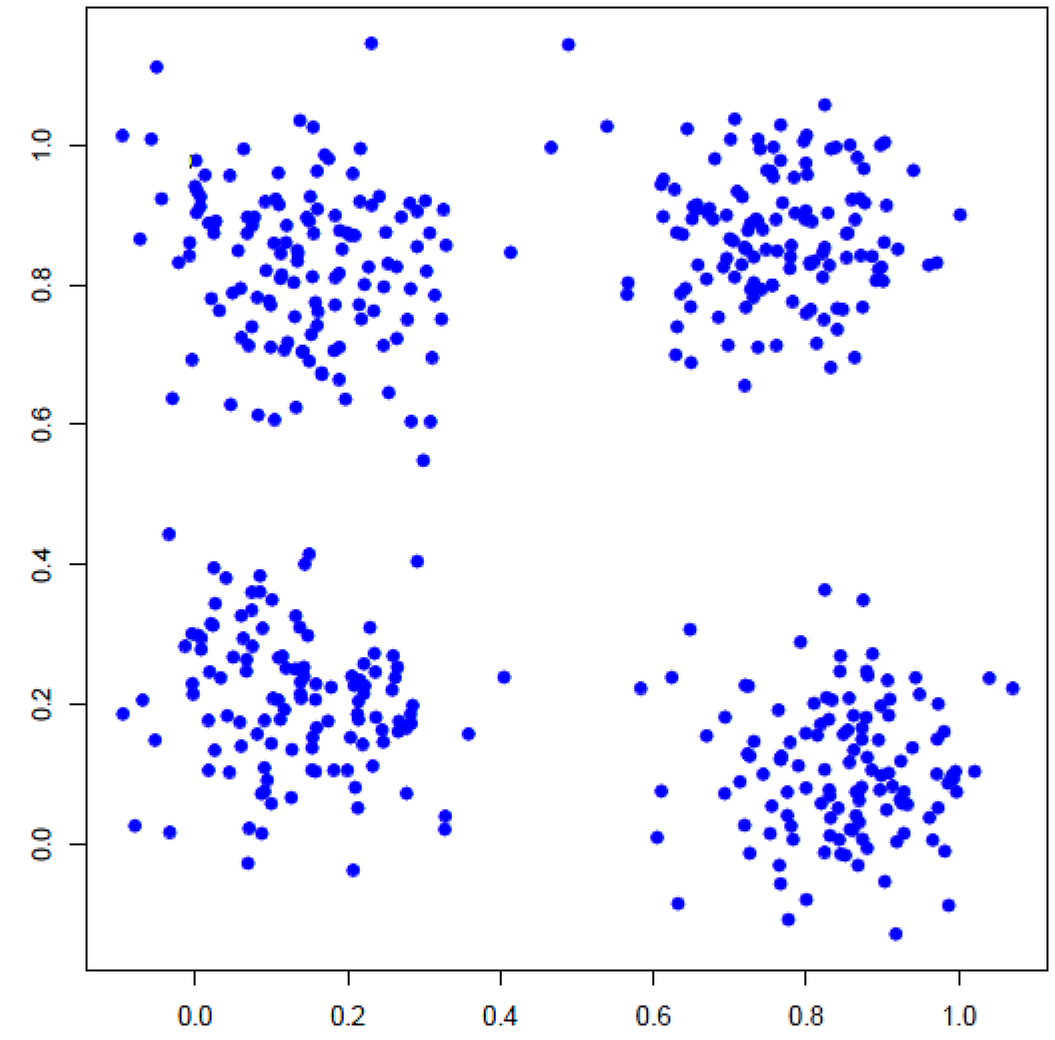


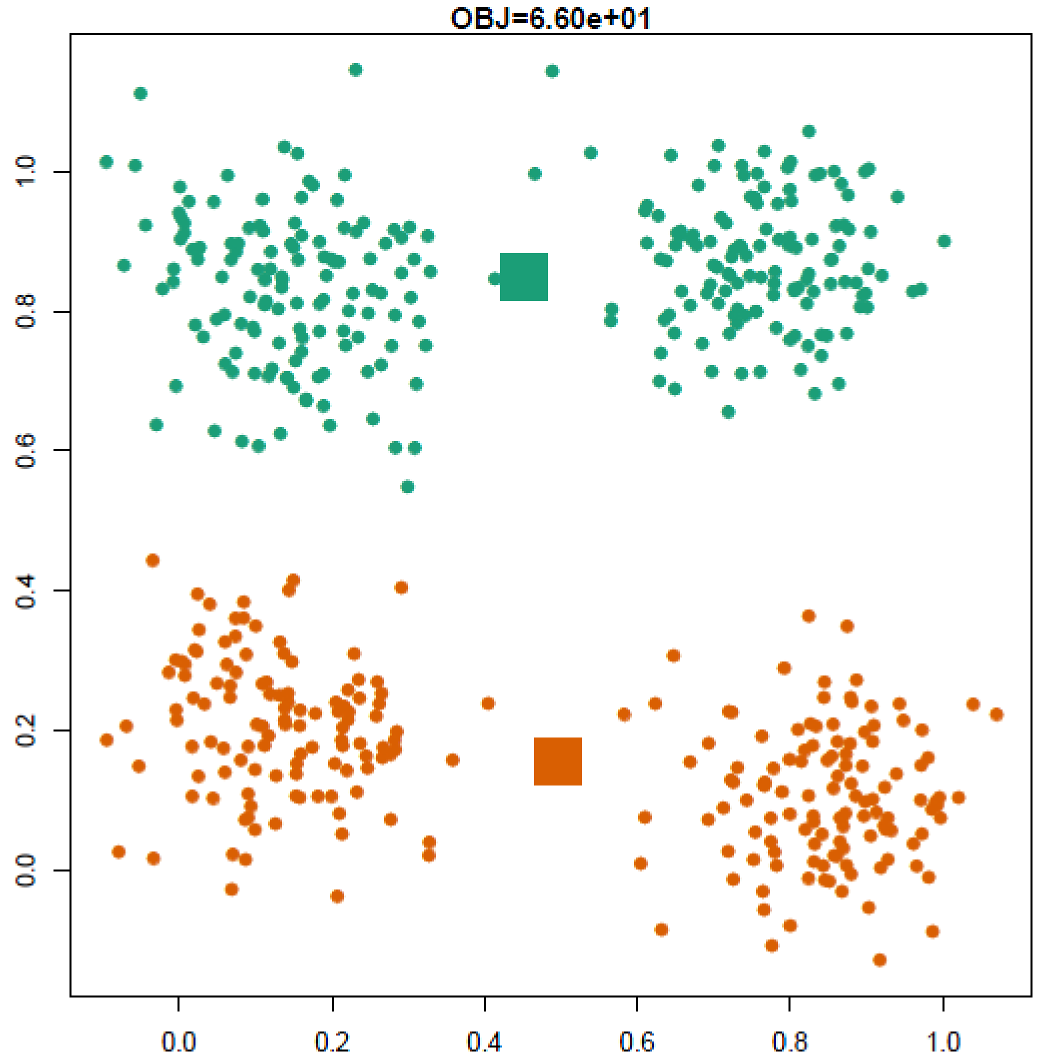


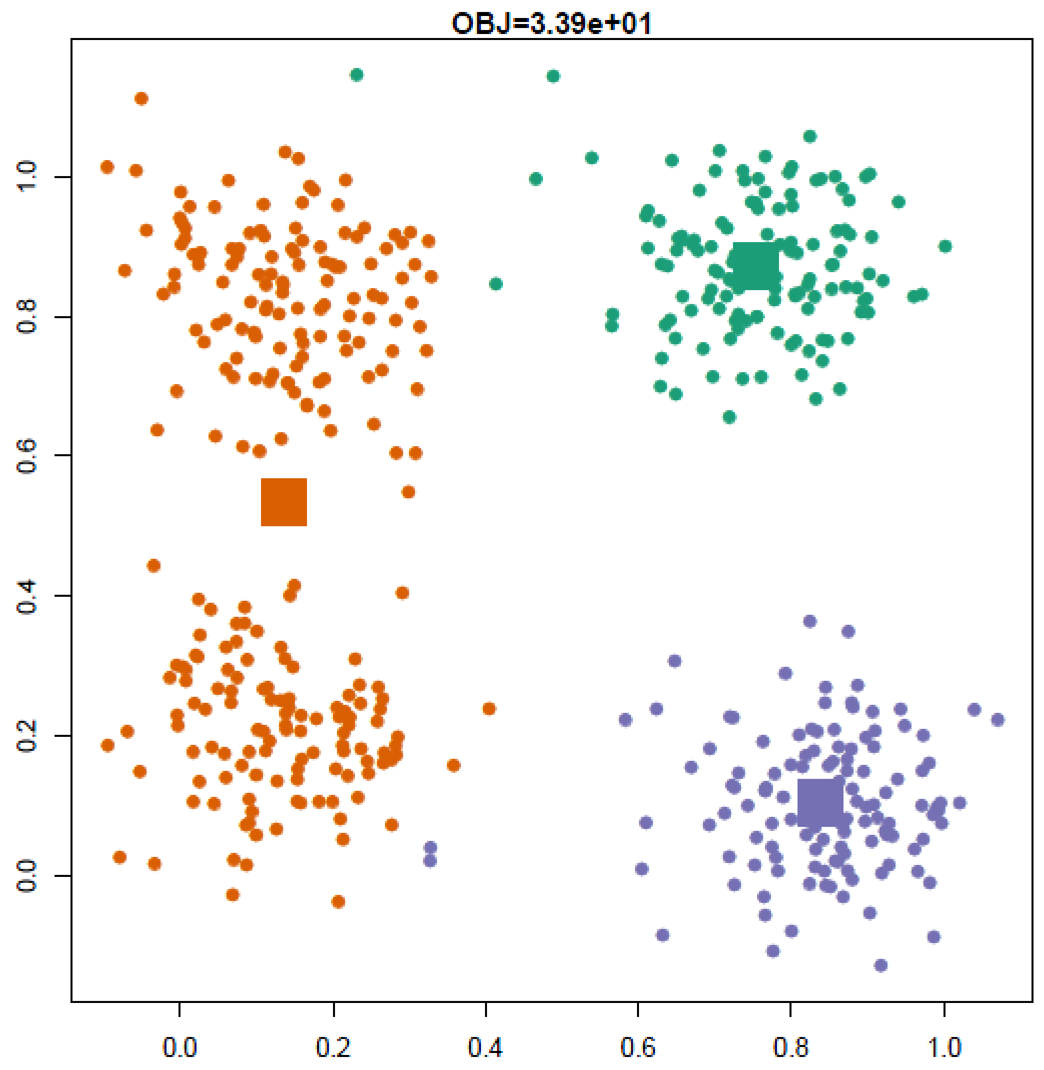
K-means, color and position is used in segmentation, four clusters (out of 20) are shown here.

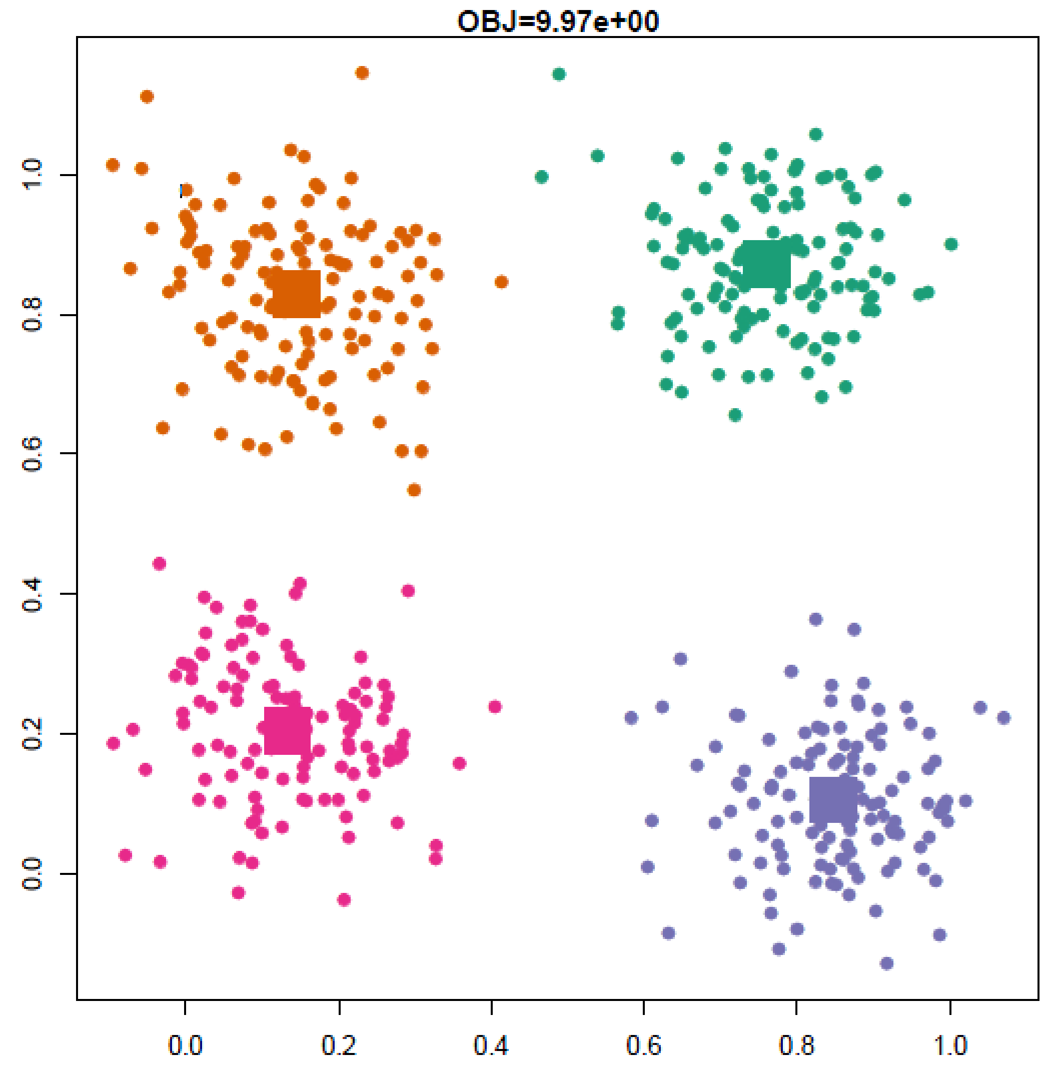
Each vector is (R,G,B,x,y).

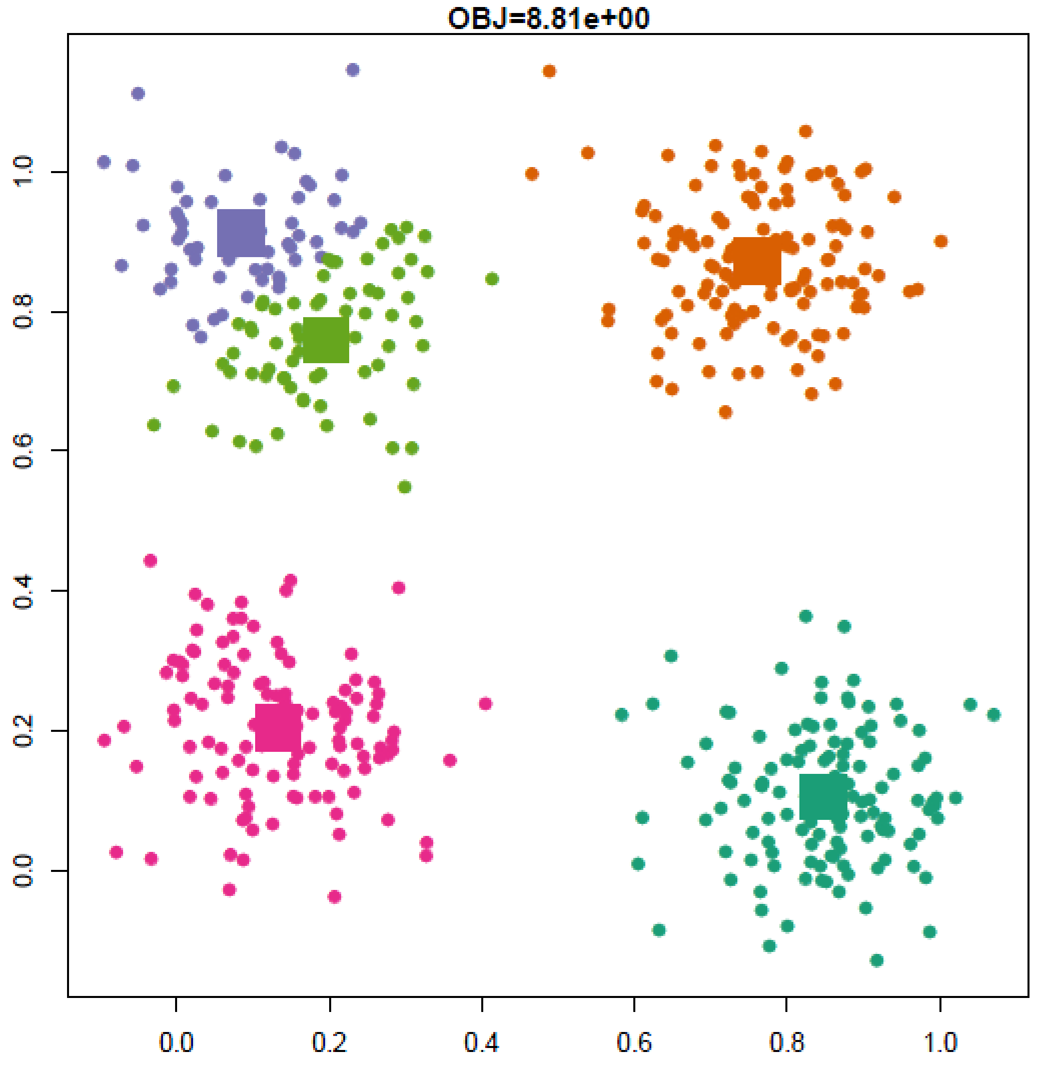
Determining K

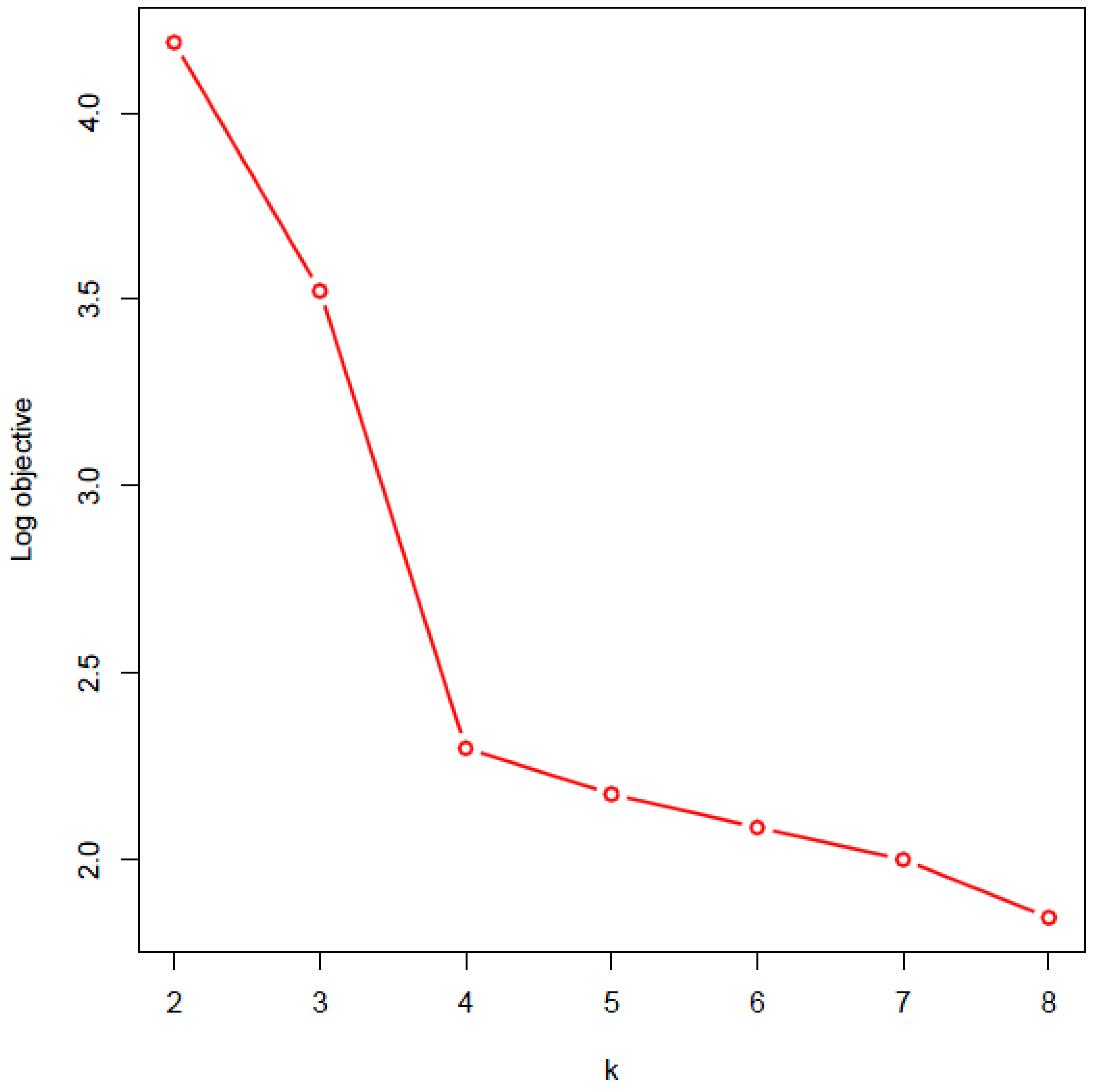



OBJ=6.60e+01
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0


OBJ=3.39e+01
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0


OBJ=9.97e+00
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

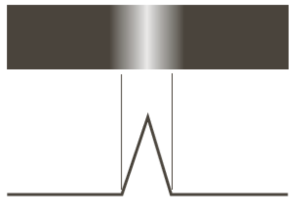

OBJ=8.81e+00
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0


Log objective
4.0
3.5
3.0
2.5
2.0
2
3
4
5
6
7
8
k

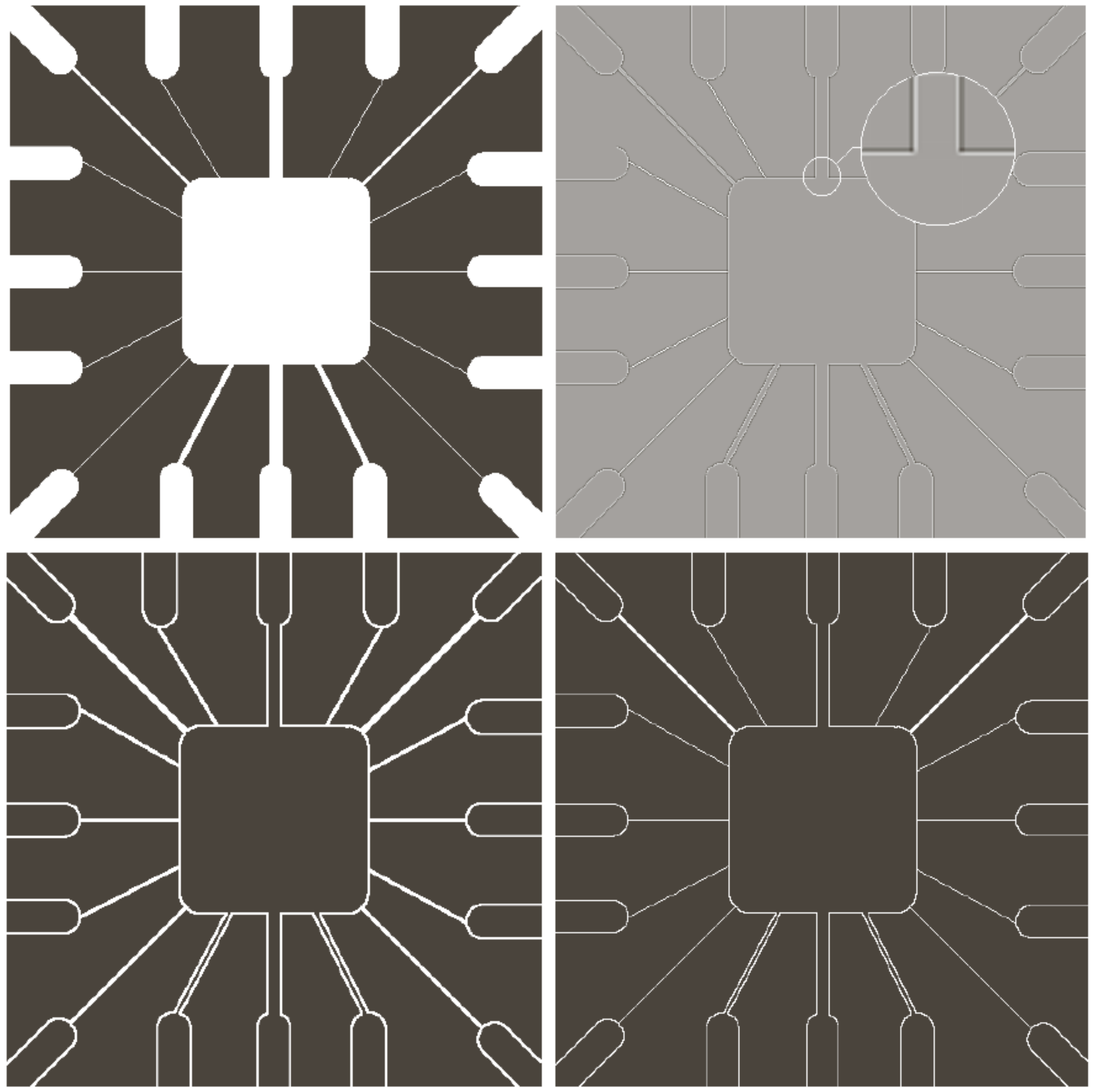
Filtering, Line Detection

Line Detection

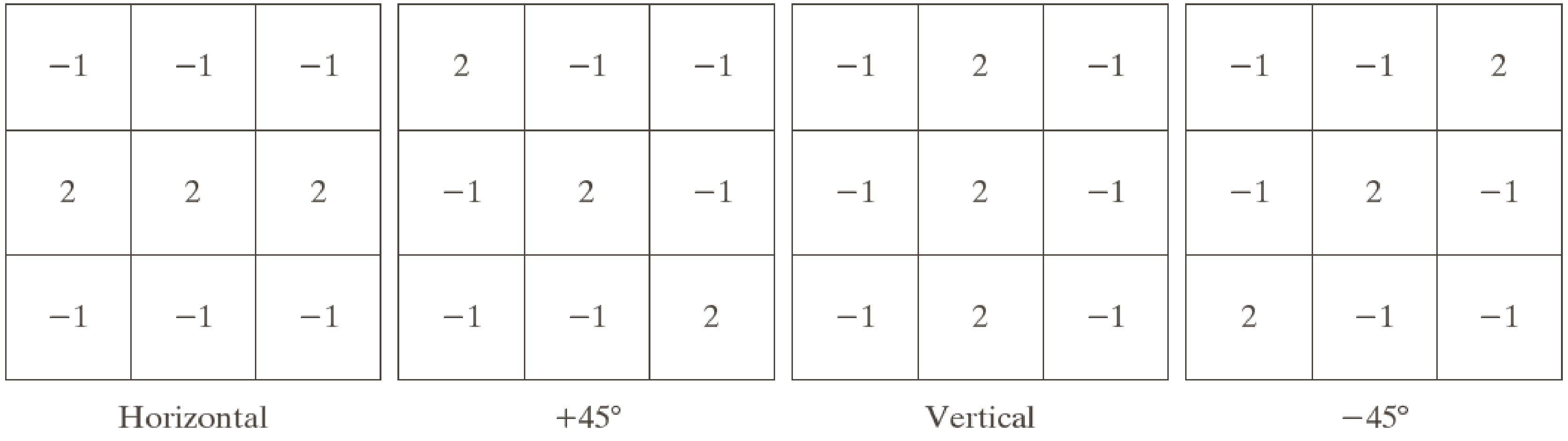
- Lines have a “roof-edge” intensity variation



- Laplacian can be used
- To highlight edges:
  - Threshold or
  - take absolute or positive-only values



- Laplacian works well for fine lines but
  - Issue 1: Gives positive response for non-straight edges, points
  - Issue 2: Gives double edges for thick lines
- To solve issue 1 and give a stronger response: Use multiple directional 2nd order filters



−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

2	−1	−1
−1	2	−1
−1	−1	2

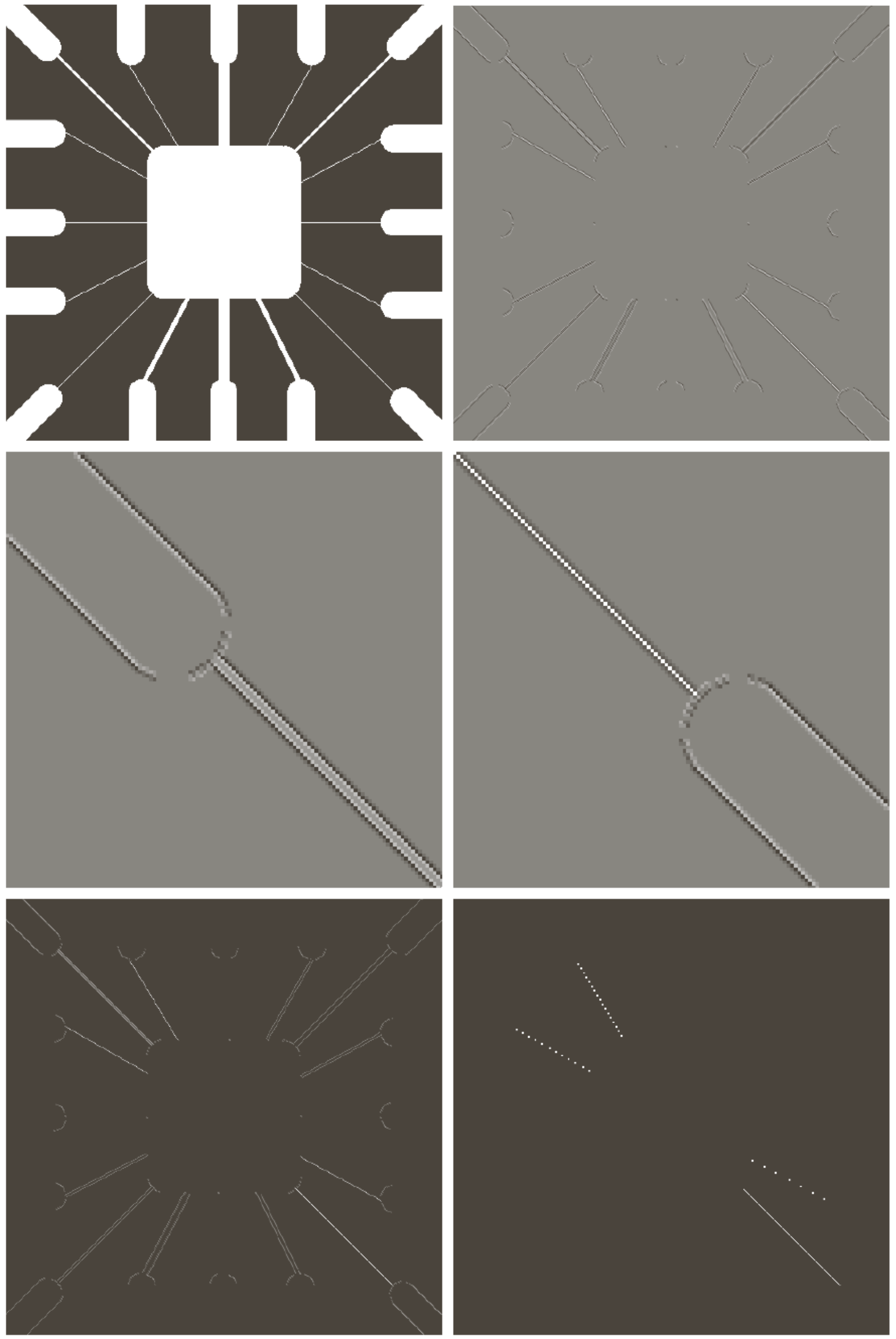
+45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

−1	−1	2
−1	2	−1
2	−1	−1

−45°

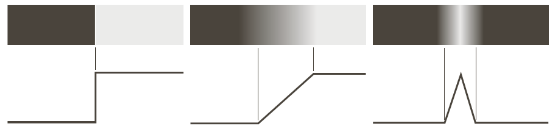


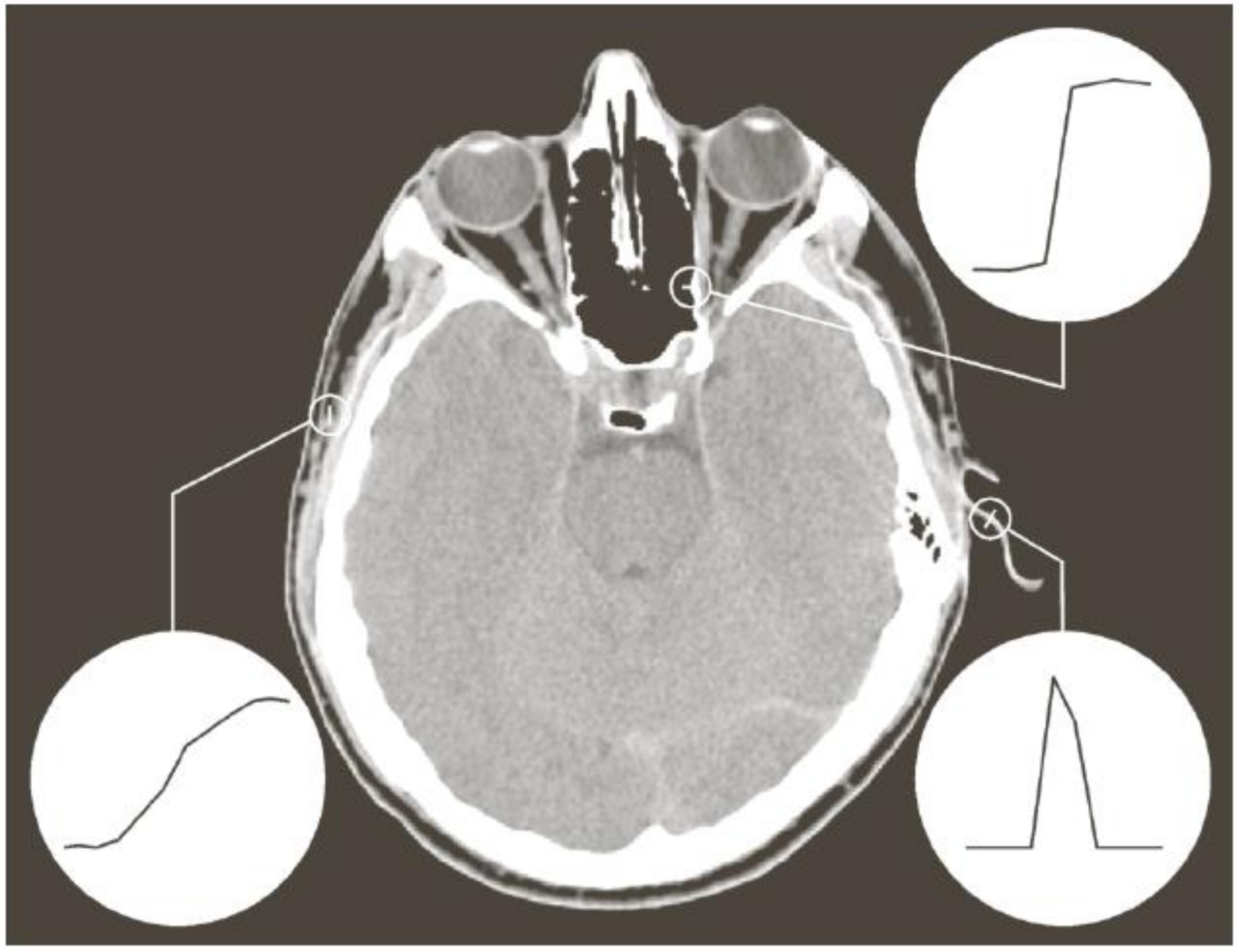
a	b
c	d
e	f

FIGURE 10.7 (a) Image of a wire-bond template. (b) Result of processing with the +45° line detector mask in Fig. 10.6. (c) Zoomed view of the top left region of (b). (d) Zoomed view of the bottom right region of (b). (e) The image in (b) with all negative values set to zero. (f) All points (in white) whose values satisfied the condition $g \geq T$, where g is the image in (e). (The points in (f) were enlarged to make them easier to see.)

Edge Detection

- 3 types of edge models: step, ramp, roof





- Edges are mostly ramps and sometimes step
- Effect of derivative filters
  - 1st order derivative can be used to locate edges
  - The negative response of the 2nd order derivative can be us e
    more precisely.
- 1st order derivative: gradient

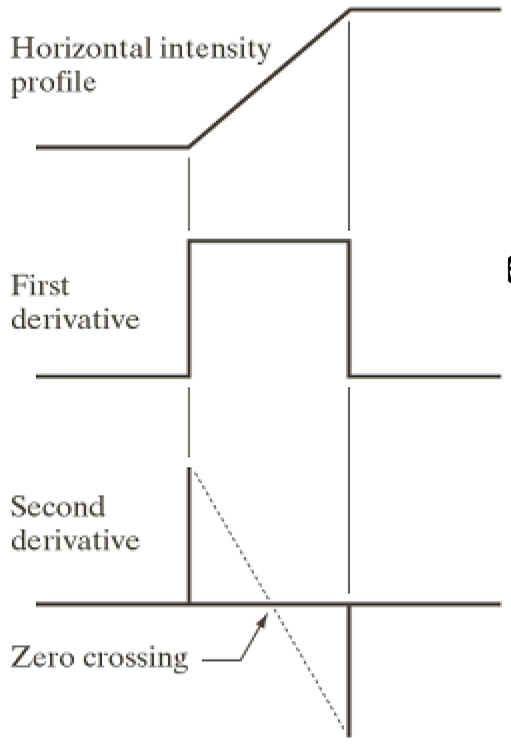


Image Gradient

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- Gradient:

$$\nabla f = [g_x ; g_y]$$

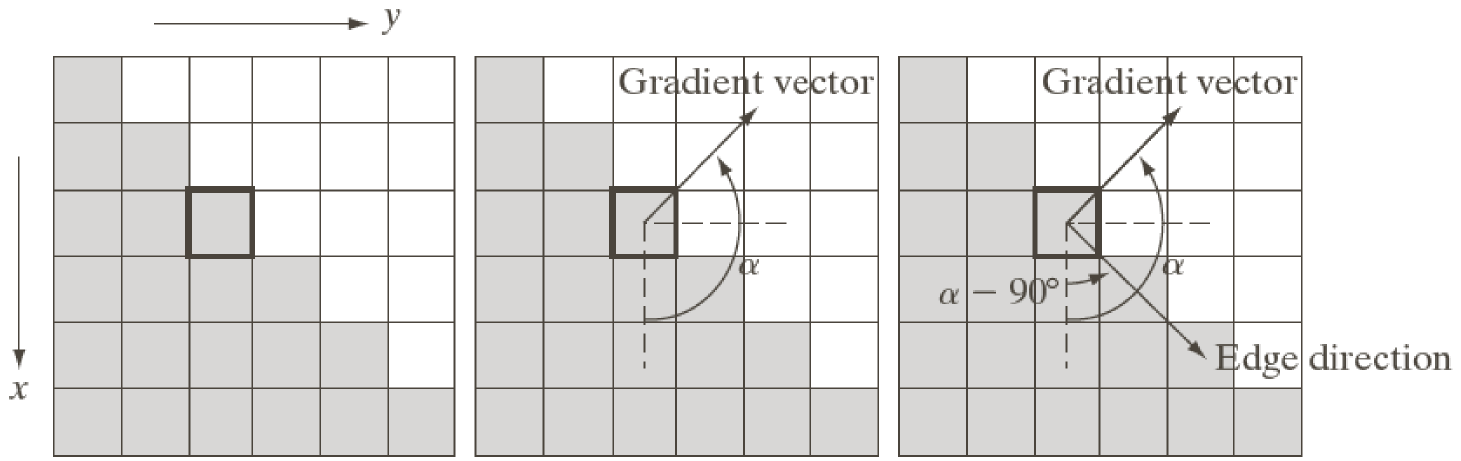
- Magnitude of ∇f:

$$M = \mathrm{sqrt}(g_x^2 ; g_y^2) \approx |g_x| + |g_y|$$

  - Simplified to reduce computational complexity
- Orientation of the ∇f:

$$\alpha = \tan\text{-}1(g_y/g_x)$$

  - The direction of the edge is perpendicular to α.



Gradient Operators

−1
1

−1	1

−1	0
0	1

0	−1
1	0

Roberts

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

Prewitt

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

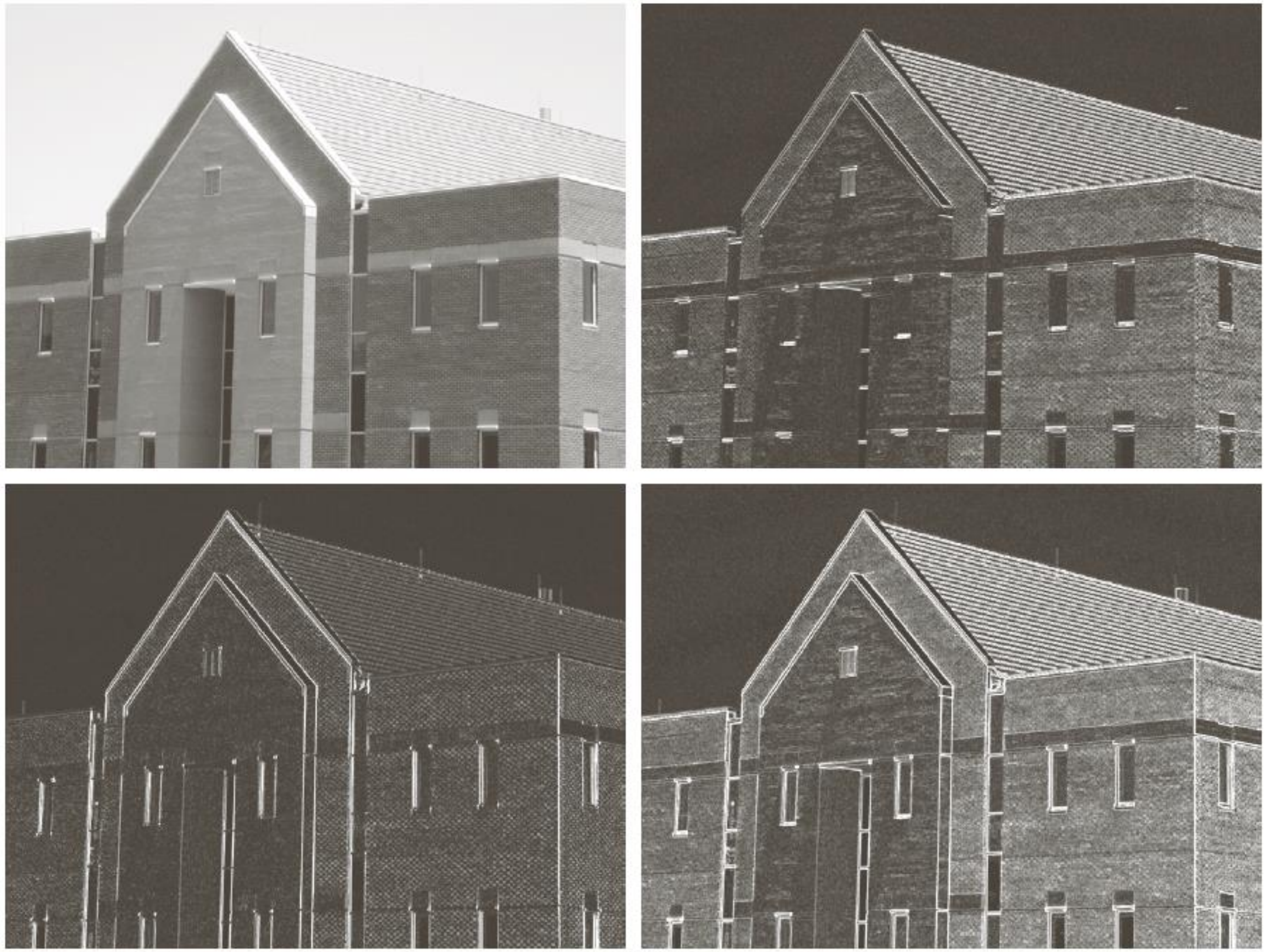
Sobel

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Sobel

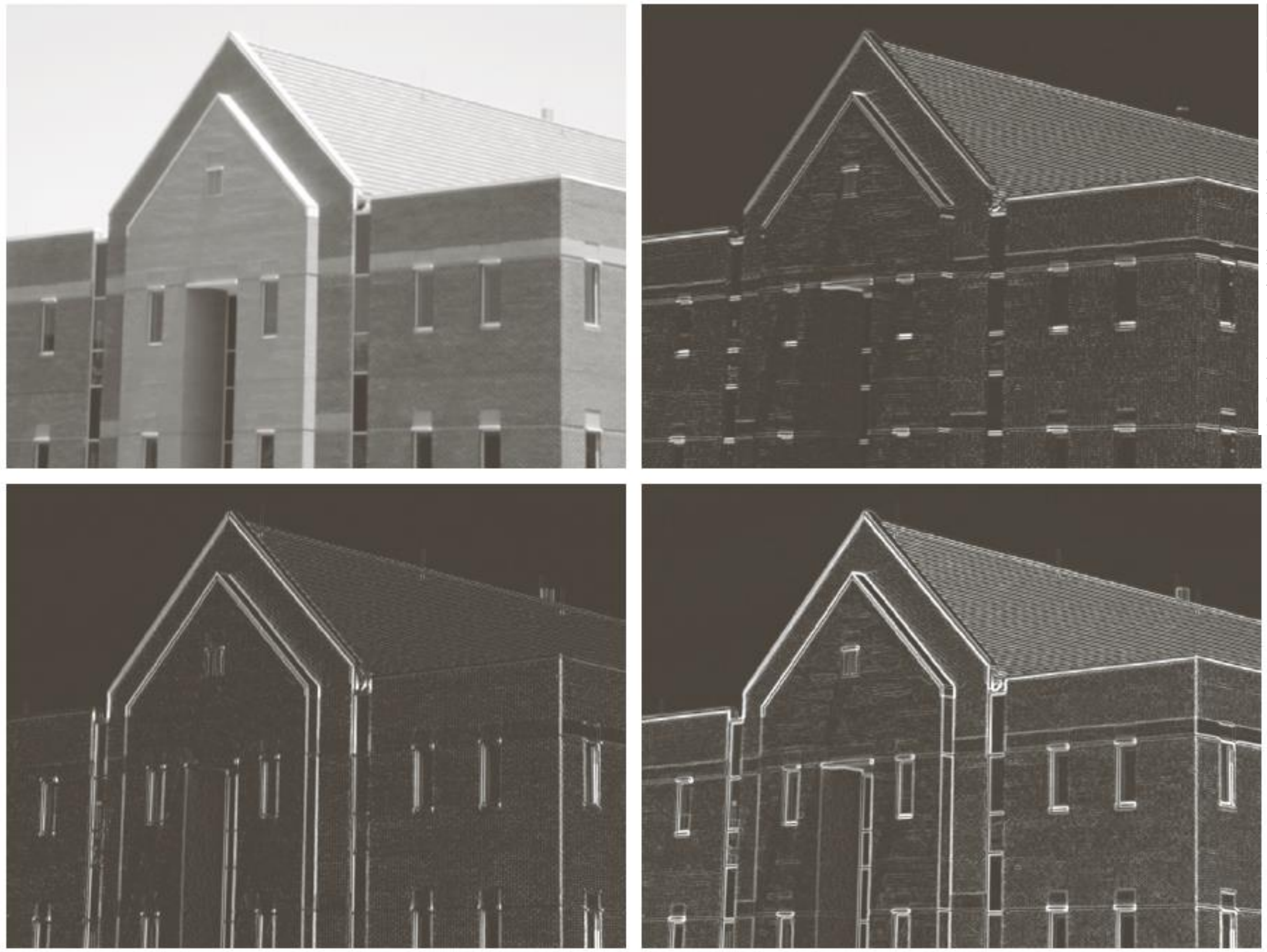
Example



a b
c d

FIGURE 10.16 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) $|g_x|$, the component of the gradient in the x-direction, obtained using the Sobel mask in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the mask in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Smoothed before Gradient



a b
c d

FIGURE 10.18 Same sequence as in Fig. 10.16, but with the original image smoothed using a 5×5 averaging filter prior to edge detection.

Smoothing and Thresholding

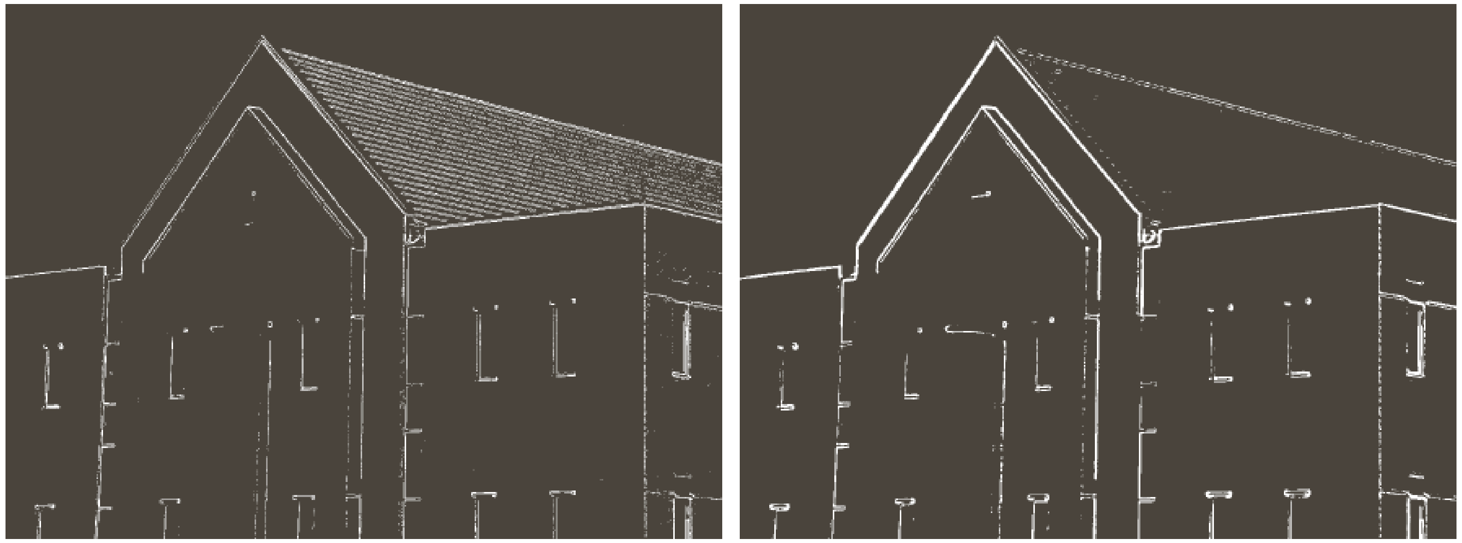


FIGURE 10.20 (a) Thresholded version of the image in Fig. 10.16(d), with the threshold selected as 33% of the highest value in the image; this threshold was just high enough to eliminate most of the brick edges in the gradient image. (b) Thresholded version of the image in Fig. 10.18(d), obtained using a threshold equal to 33% of the highest value in that image.

Gradient in the Presence of Noise

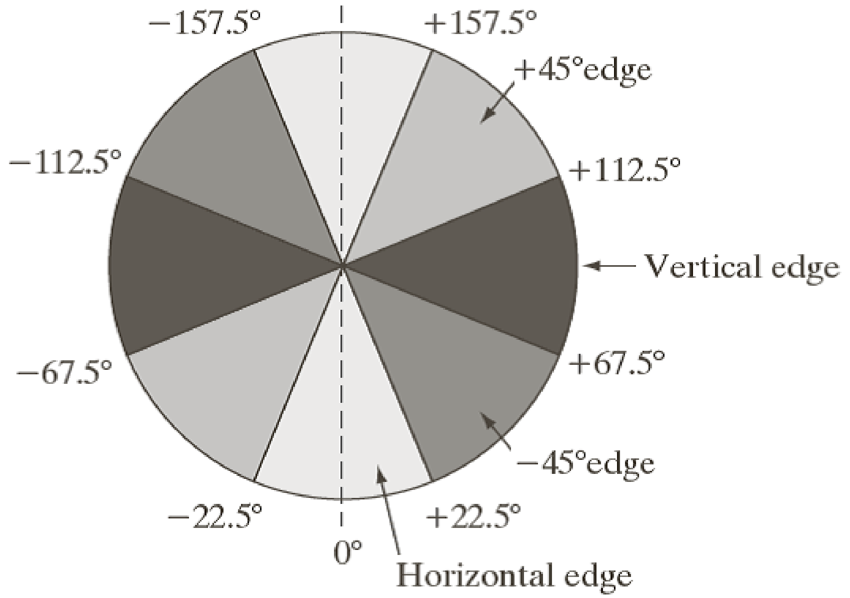
- Differentiation amplifies noise
- Solution: Smooth first
- Edge detection process:
  - Image smoothing for noise reduction
    - Related to scale issue
  - Detection of candidates for edge points
    - Thick band of pixels around an edge
  - Edge localization
    - Select from the candidates only the points that are truly on an edge, i.e. from thick edge, select a 1-pixel thick curve of edge points

Canny Edge Detector

- Three objectives:
  - Low error rate: Find all true edges but no spurious ones.
  - Good localization: Distance between true edges and pixels marked as edges should be minimum.
  - Single edge response: The detector should mark only one point as an edge for every true edge point.
- Low error rate:
  - Optimal edge detector is the 1st derivative of the Gaussian oriented perpendicular to the edge
  - Smooth the image with a 2D Gaussian then use the gradient magnitude and direction.
  - Gradient magnitude M measures strength of the edge
  - Gradient orientation α measures orientation of edge

Nonmaxima Suppression

- Consider four orientations in a 3x3 region: vertical (d_1), horizontal (d_2), +45° (d_3)



1. Find d_k that is closest to α
2. If $M(x_i,y_i)$ less than at least one of its neighbor along d_k, then suppress: $g_N(x,y)=0$; otherwise $g_N(x,y)=M(x,y)$
   - Makes thin edges

3. Apply a hysteresis thresholding:
   - Strong edges: $g_{NH} = (g_N >= T_H)$
   - Weak edges: $g_{NL} = (g_N >= T_L)$
   - $g_{NL} = g_{NL} - g_{NH}$
   - Fewer pixels in g_{NH}
4. Fill gaps in g_{NL}
   - Locate edge pixel, p, in g_{NH}
   - Set all connected pixels to p in g_{NL}

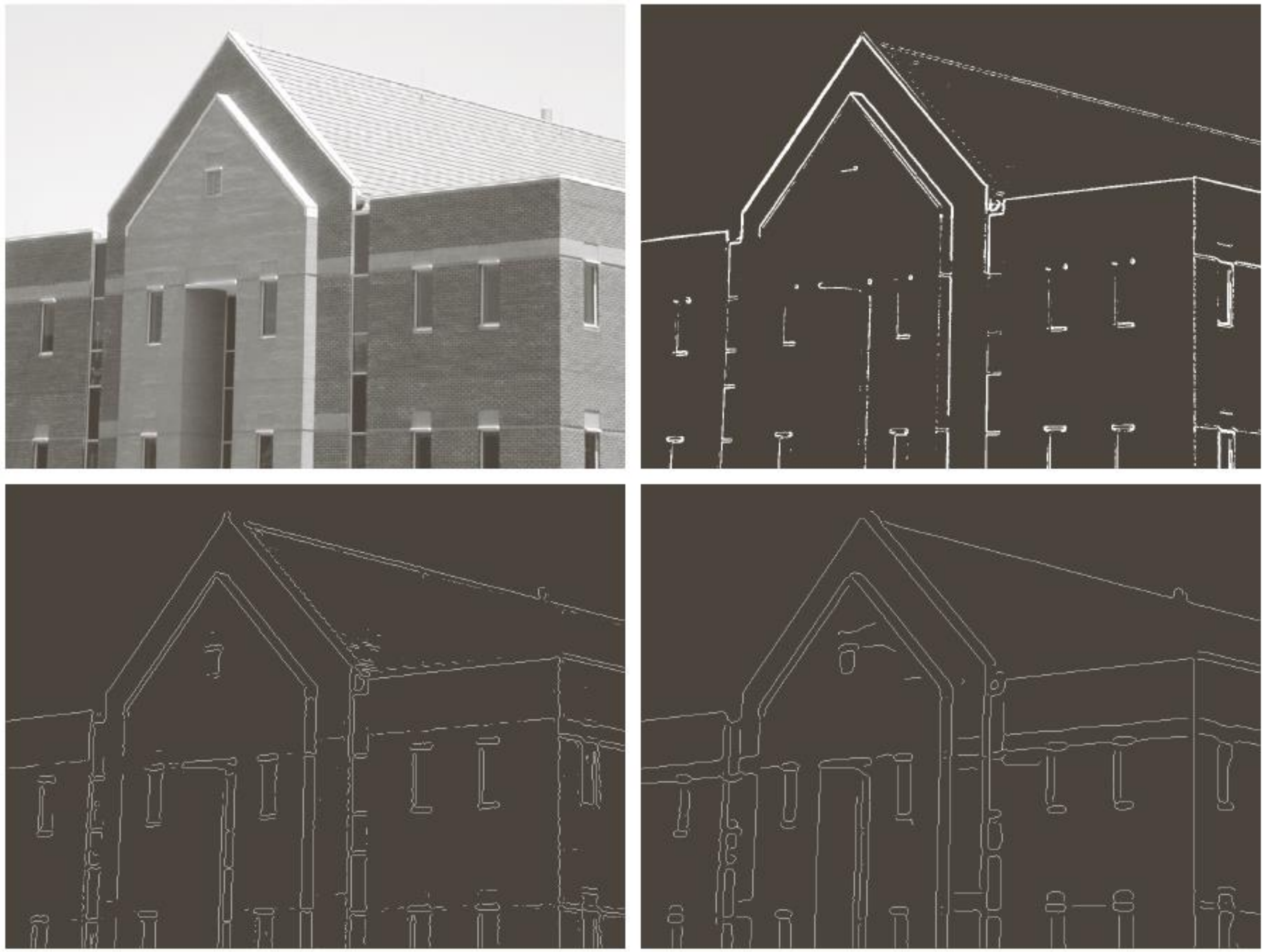

a b
c d

FIGURE 10.25 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Thresholded gradient of smoothed image. (c) Image obtained using the Marr-Hildreth algorithm. (d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny image compared to the other two.