Digital Image Processing

Lecture # 6
Corner Detection & Color Processing

Corners

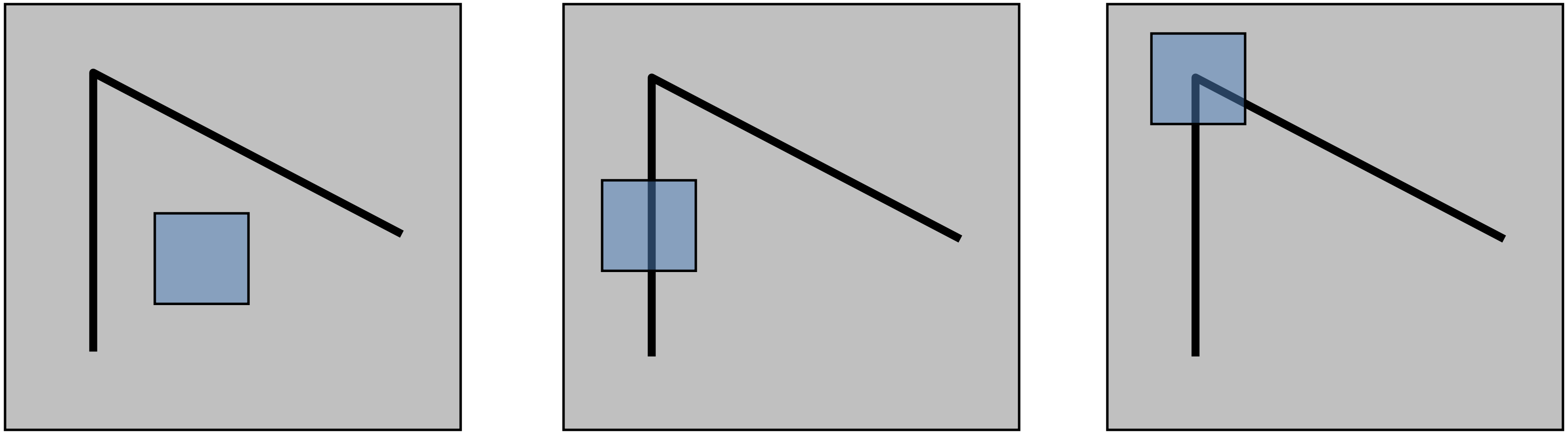
Corners (interest points)

- Unlike edges, corners (patches of pixels surrounding the corner) do not necessarily correspond to the geometric entities of the observed scene
- They capture corner structures in the pattern of intensities
- Prove stable in a sequence of images, hence, help in tracking objects across sequences
- Good for correspondence algorithms in Stereopsis, structure from motion and reconstruction
- Corners are specific locations in the image like, mountain peaks, building corners, and interestingly shaped patches of snow.
- Permit matching even in the presence of occlusion (clutter) and large scale and orientation changes.

Local measures of uniqueness

Suppose we only consider a small window of pixels

- What defines whether a feature is a good or bad candidate?

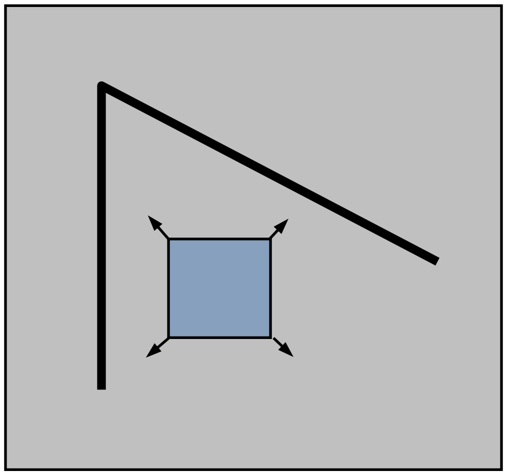


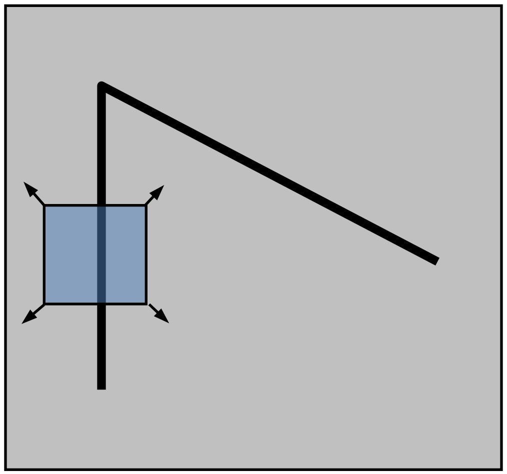
Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

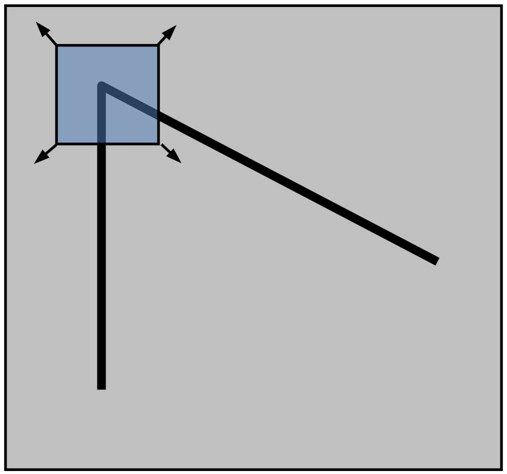
Feature detection

Local measure of feature uniqueness

- How does the window change when you shift it?
- Shifting the window in *any direction* causes a *big change*







“flat” region:
no change in all directions

“edge”:
no change along the edge direction

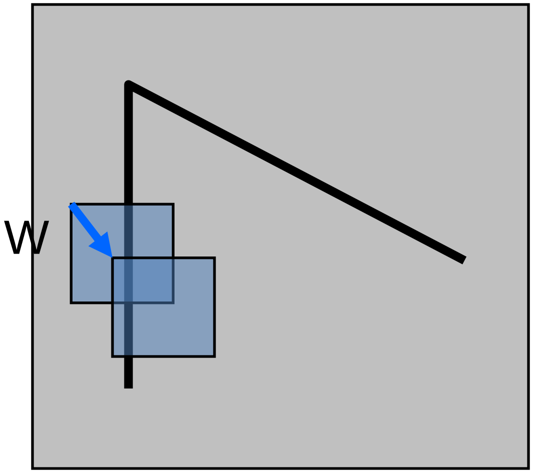
“corner”:
significant change in all directions

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Feature detection: the math

Consider shifting the window W by (u,v)

- how do the pixels in W change?
- compare each pixel before and after by summing up the squared differences (SSD)
- this defines an SSD "error" of *E(u,v)*:



$$E(u,v) = \sum_{(x,y)\in W} \left[I(x+u,y+v) - I(x,y)\right]^2$$

Small motion assumption

Taylor Series expansion of I:

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

If the motion (u,v) is small, then first order approx is good

$$I(x+u, y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

$$\approx I(x,y) + [I_x \ I_y] \begin{bmatrix} u \\ v \end{bmatrix}$$

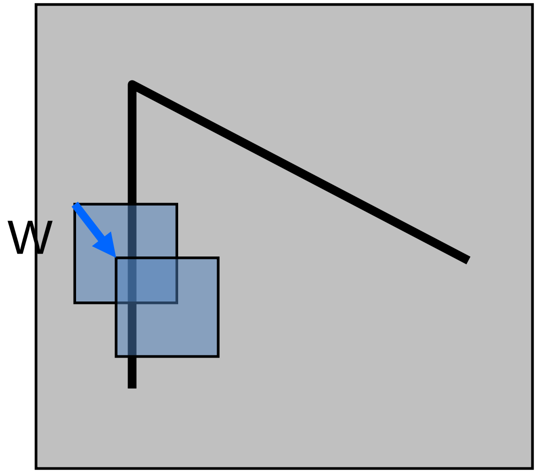
Plugging this into the formula on the previous slide…

shorthand: $I_x = \frac{\partial I}{\partial x}$

Feature detection: the math

Consider shifting the window W by (u,v)

- how do the pixels in W change?
- compare each pixel before and after by summing up the squared differences
- this defines an "error" of E(u,v):



$$E(u,v) = \sum_{(x,y)\in W} \left[I(x+u,y+v) - I(x,y)\right]^2$$

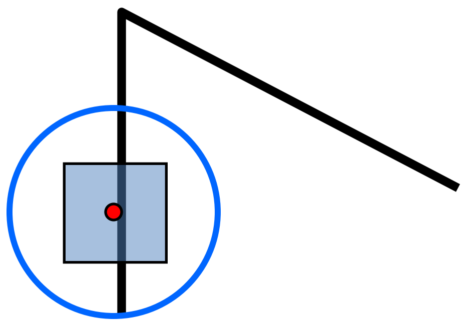
$$\approx \sum_{(x,y)\in W} \left[I(x,y) + [I_x \; I_y]\begin{bmatrix} u \\ v \end{bmatrix} - I(x,y)\right]^2$$

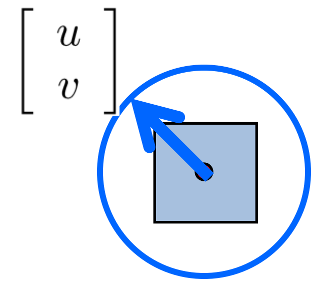
$$\approx \sum_{(x,y)\in W} \left[[I_x \; I_y]\begin{bmatrix} u \\ v \end{bmatrix}\right]^2$$

Feature detection: the math

This can be rewritten:

$$E(u,v) = \sum_{(x,y)\in W} [u\ v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{H} \begin{bmatrix} u \\ v \end{bmatrix}$$





For the example above

- You can move the center of the green window to anywhere on the blue unit circle
- Which directions will result in the largest and smallest E values?
- We can find these directions by looking at the eigenvectors of *H*

Quick eigenvalue/eigenvector review

The **eigenvectors** of a matrix **A** are the vectors **x** that satisfy:

$$Ax = \lambda x$$

The scalar λ is the **eigenvalue** corresponding to **x**

- The eigenvalues are found by solving:

$$det(A - \lambda I) = 0$$

- In our case, ***A*** = ***H*** is a 2x2 matrix, so we have

$$det \begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} = 0$$

- The solution:

$$\lambda_{\pm} = \frac{1}{2} \left[(h_{11} + h_{22}) \pm \sqrt{4 h_{12} h_{21} + (h_{11} - h_{22})^2} \right]$$

Once you know λ, you find **x** by solving

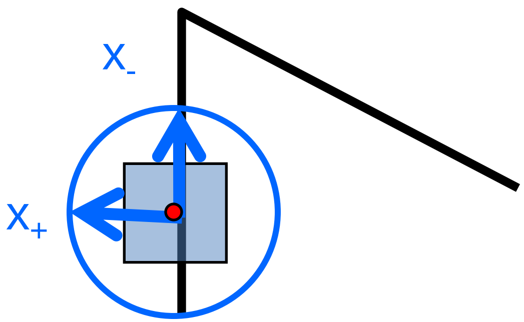
$$\begin{bmatrix} h_{11} - \lambda & h_{12} \\ h_{21} & h_{22} - \lambda \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0$$

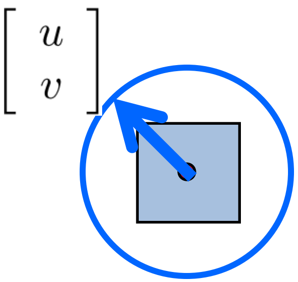
Feature detection: the math

This can be rewritten:

$$E(u,v) = \sum_{(x,y)\in W} [u\ v] \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

H





Eigenvalues and eigenvectors of H

- Define shifts with the smallest and largest change (E value)
- x_+ = direction of largest increase in E.
- λ_+ = amount of increase in direction x_+
- x_- = direction of smallest increase in E.
- λ- = amount of increase in direction x_+

$$\begin{aligned} Hx_+ &= \lambda_+ x_+ \\ Hx_- &= \lambda_- x_- \end{aligned}$$

Feature detection: the math

How are λ_+, x_+, λ_-, and x_+ relevant for feature detection?

- What's our feature scoring function?

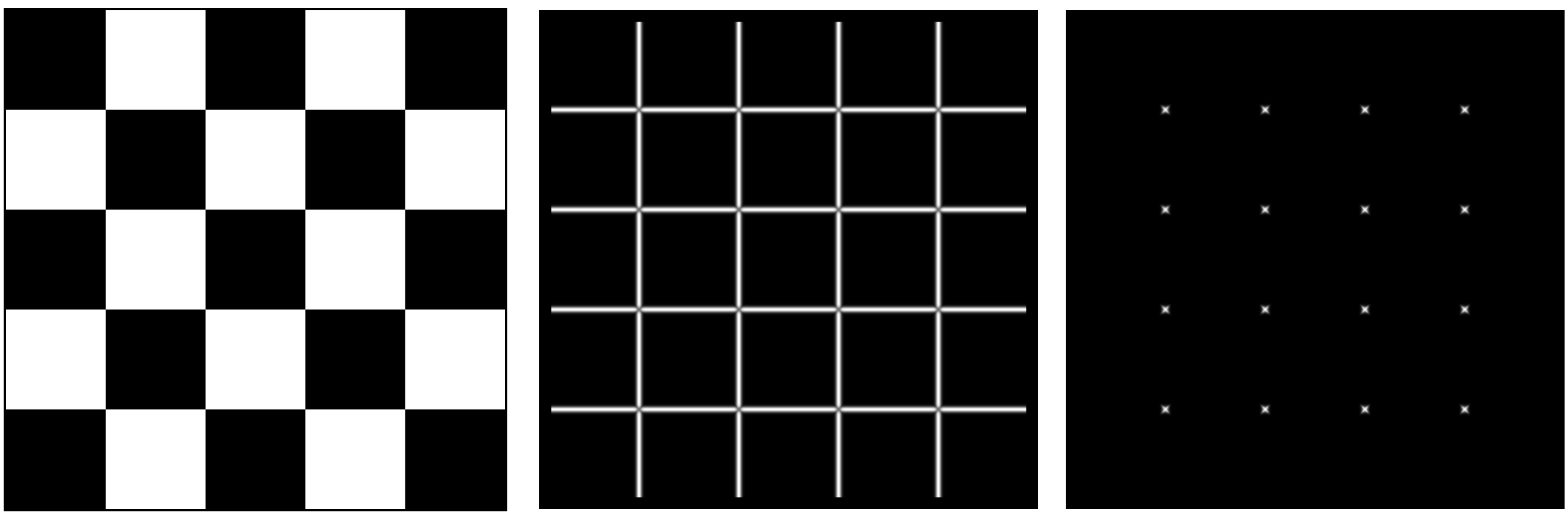
Feature detection: the math

How are λ_+, x_+, λ_-, and x_+ relevant for feature detection?

- What's our feature scoring function?

Want *E(u,v)* to be *large* for small shifts in *all* directions

- the *minimum* of *E(u,v)* should be large, over all unit vectors [u v]
- this minimum is given by the smaller eigenvalue (λ_-) of *H*

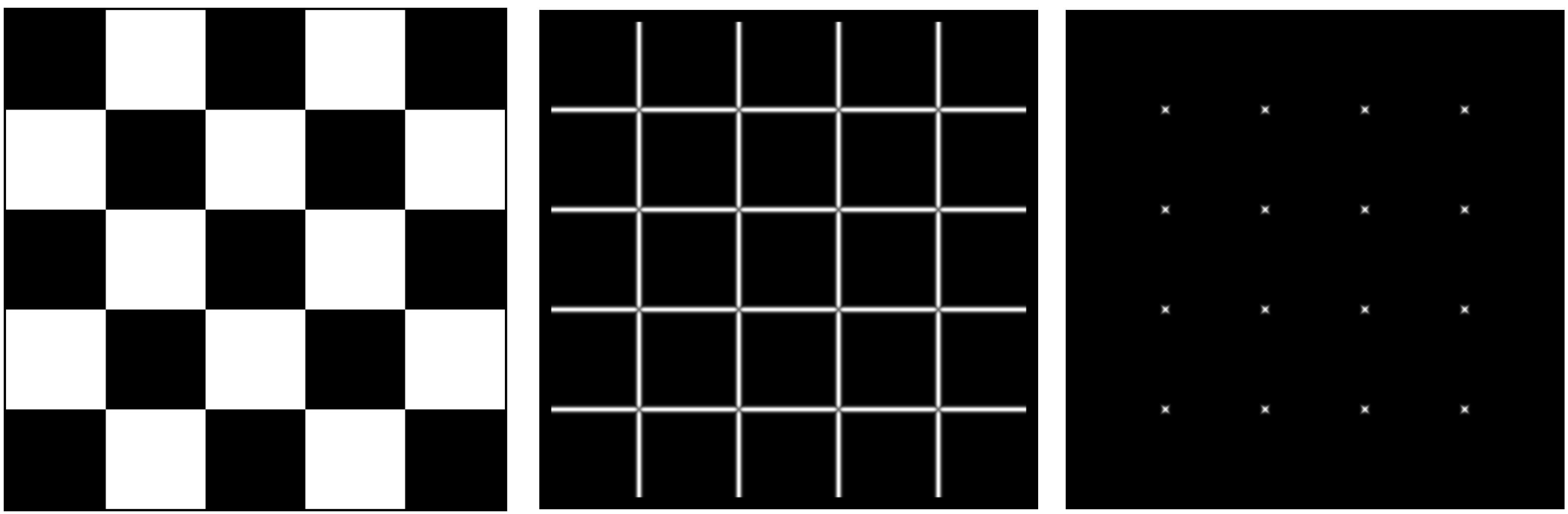


I $\qquad$ λ_+ $\qquad$ λ_-

Feature detection summary

Here's what you do

- Compute the gradient at each point in the image
- Create the *H* matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold)
- Choose those points where λ_- is a local maximum as features

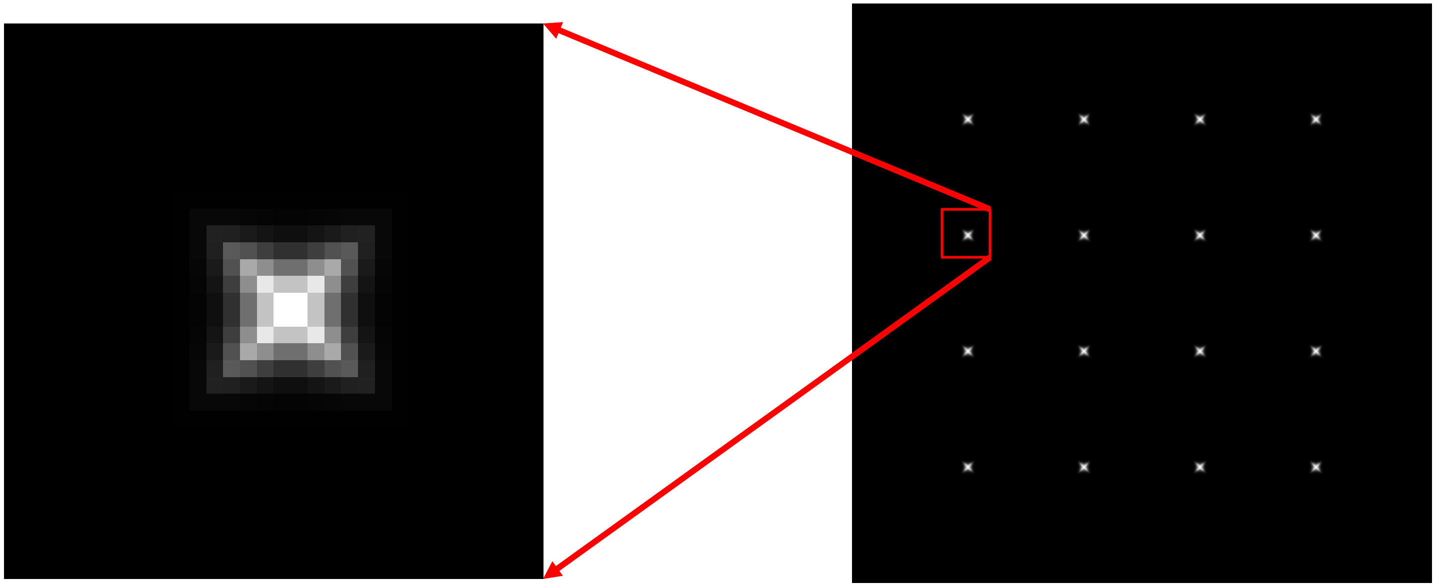


I λ_+ λ_-

Feature detection summary

Here's what you do

- Compute the gradient at each point in the image
- Create the *H* matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold)
- Choose those points where λ_- is a local maximum as features



λ_-

Szeliski 2005 use harmonic mean

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

$$= \frac{determinant(H)}{trace(H)}$$

- The *trace* is the sum of the diagonals, i.e., $trace(H) = h_{11} + h_{22}$
- Very similar to λ_- but less expensive (no square root) . (That method relies on 1/square root of λ_-)
- Lots of other detectors

Harris Detector

- Harris and Stephens (Harris & Stephens, 1988) : Down weights edge like features

Measure of corner response:

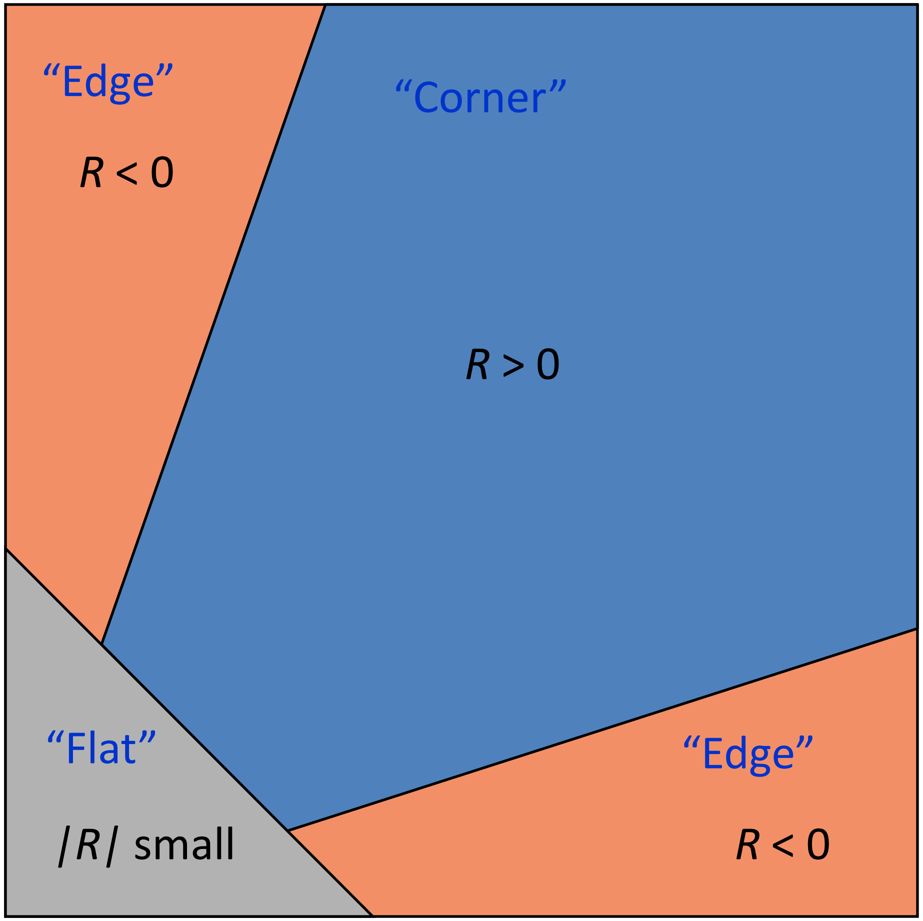
$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

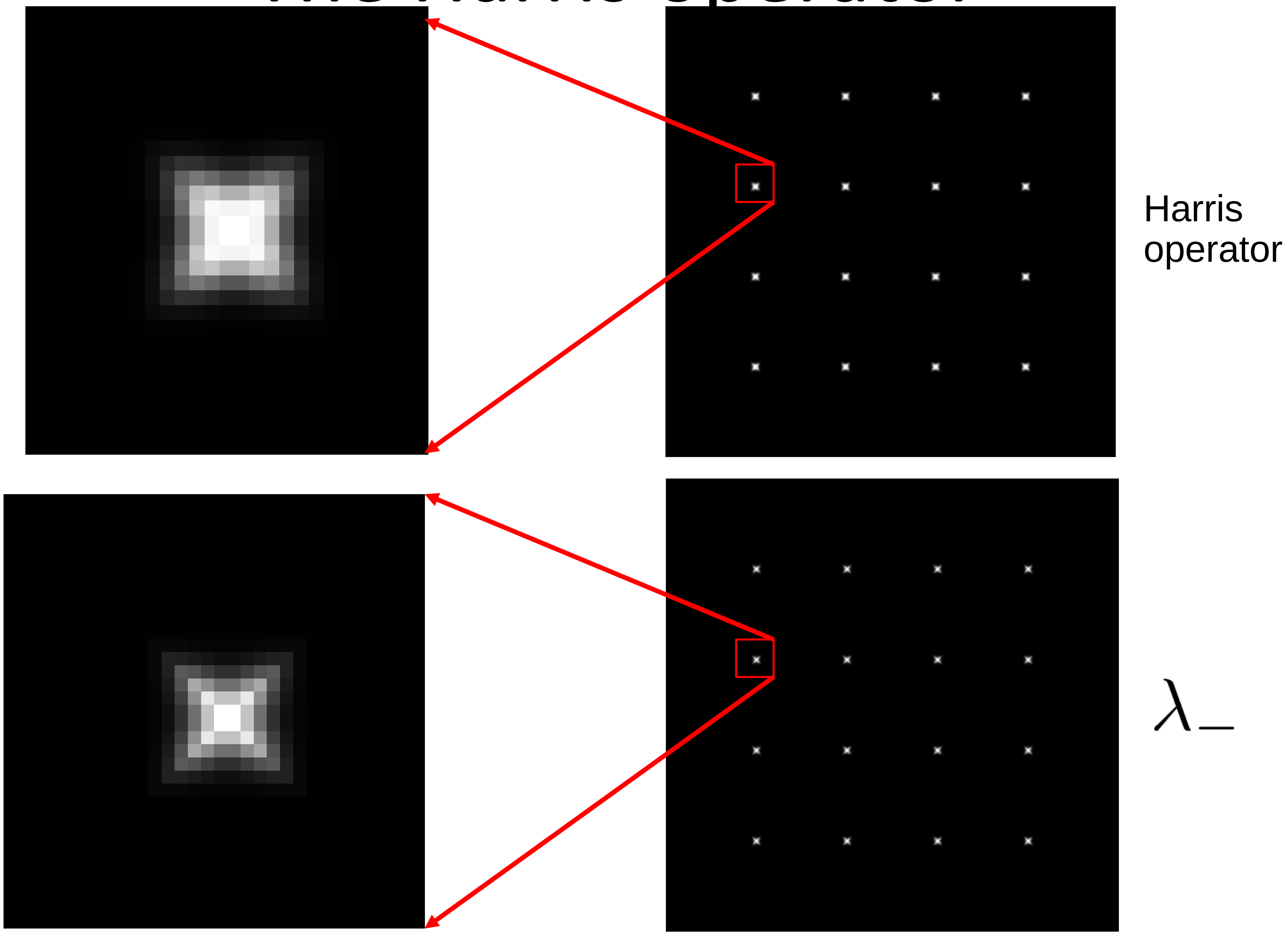
(*k* – empirical constant, *k* = 0.04-0.06)

Harris Detector: Mathematics

- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region

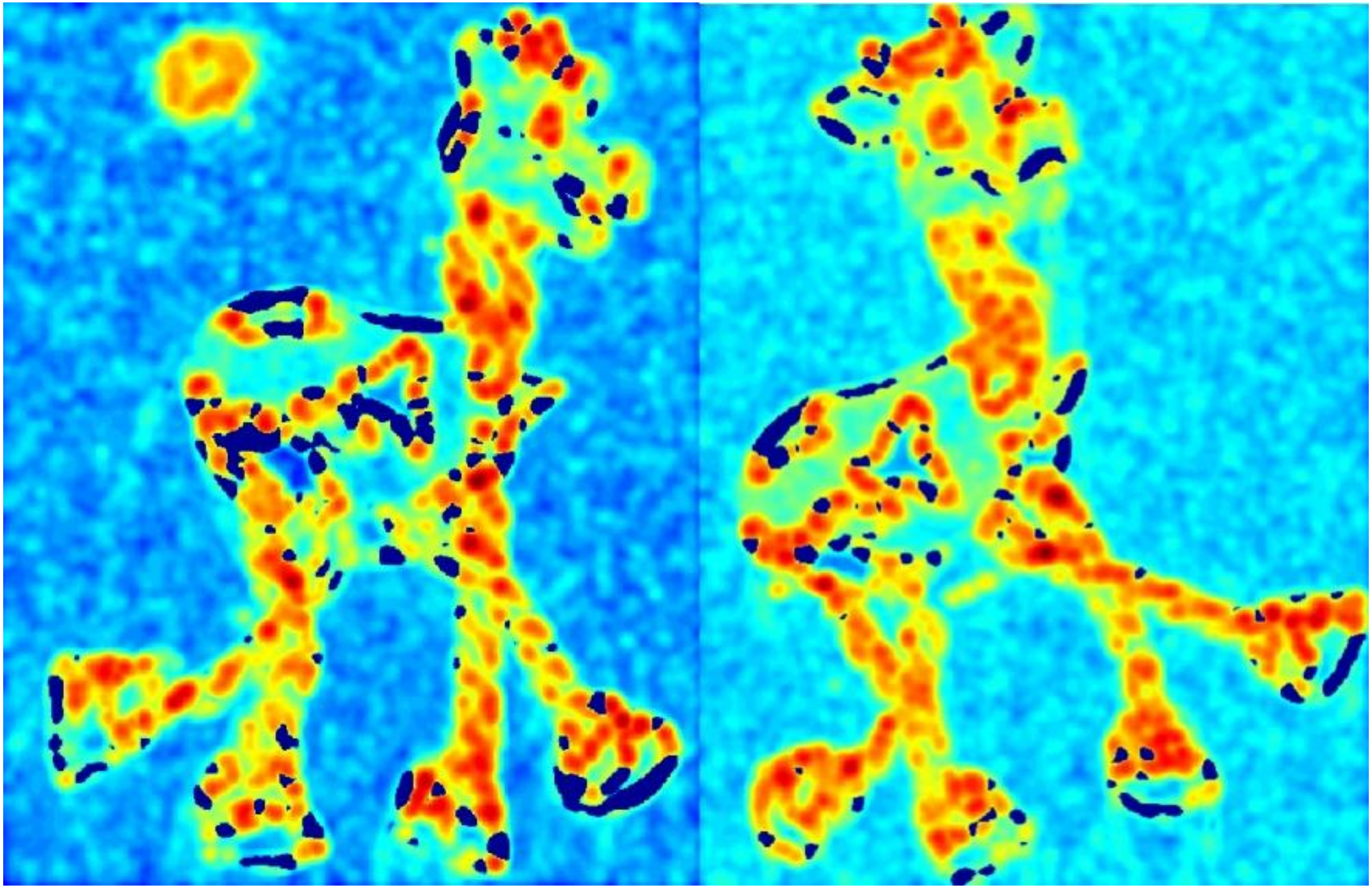


The Harris operator

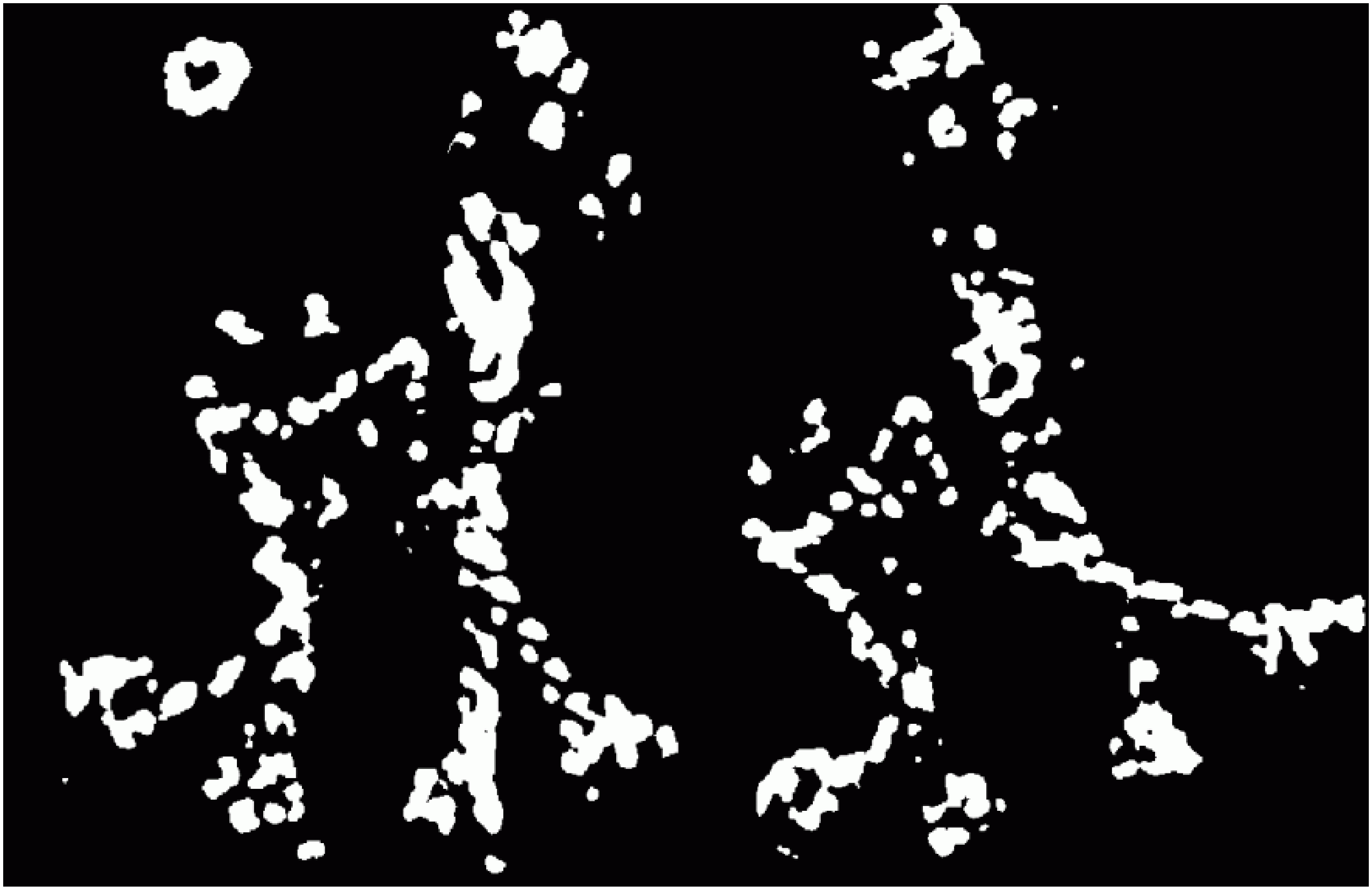


Harris detector example

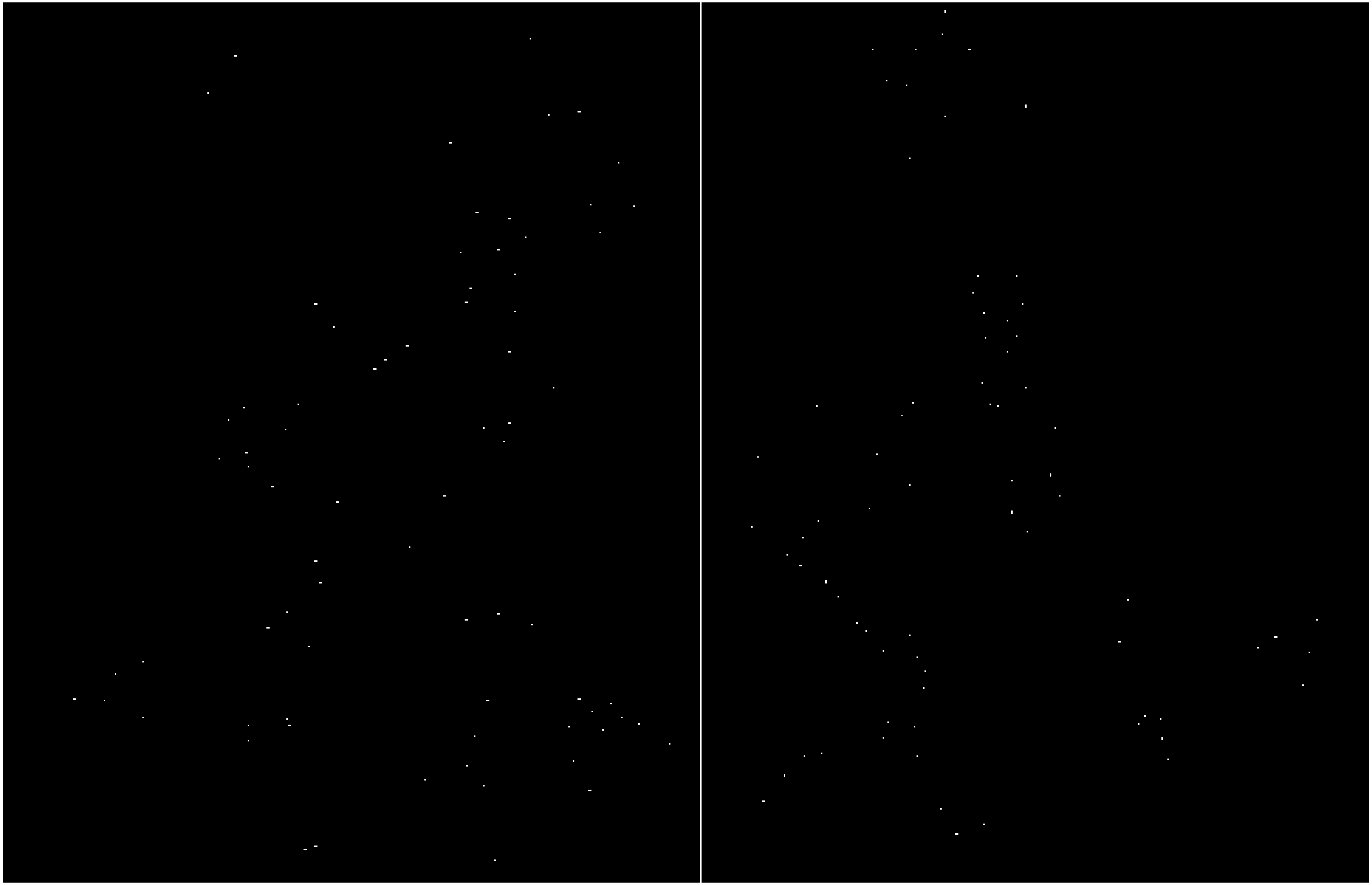
f value (red high, blue low)



Threshold (f > value)



Find local maxima of f



Harris features (in red)

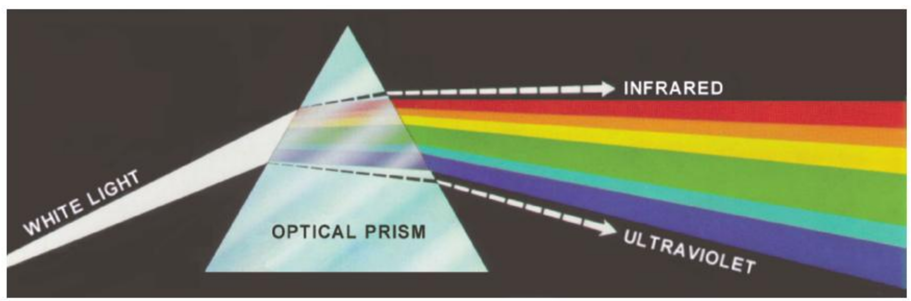
COLOR IMAGE PROCESSING

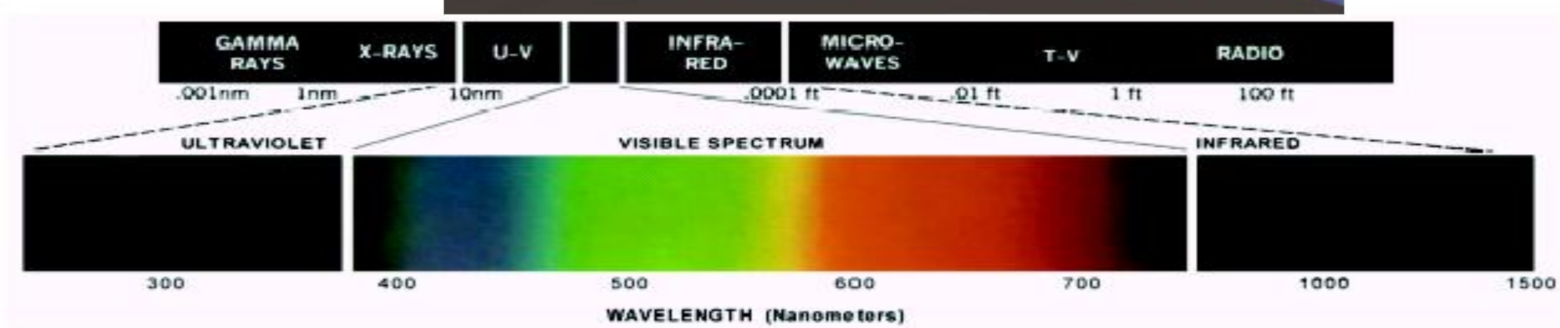
COLOR IMAGE PROCESSING

- Color Importance
  - Color is an excellent descriptor
    - Suitable for object Identification and Extraction
  - Discrimination
    - Humans can distinguish thousands of color shades and intensities but few shades of gray levels
- Color Image Processing
  - Full-Color Processing
    - Color is acquired with a full-color sensor
  - Pseudo-Color Processing
    - Assigning colors to monochrome images

COLOR FUNDAMENTALS

- Colors that humans perceive in an object are determined by the nature of the light reflected from the object
- Visible light is composed of a relatively narrow band of frequencies in the electromagnetic spectrum
- A body that reflects light that is balanced in all visible wavelengths appears white to the observer
- A body that favours reflectance in a limited range of the visible spectrum exhibits some shades of color
- Green objects reflect light with wavelengths primarily in the 500 to 570 nm range while absorbing most of the energy at other wavelengths





HUMAN PERSCEPTION OF COLOR

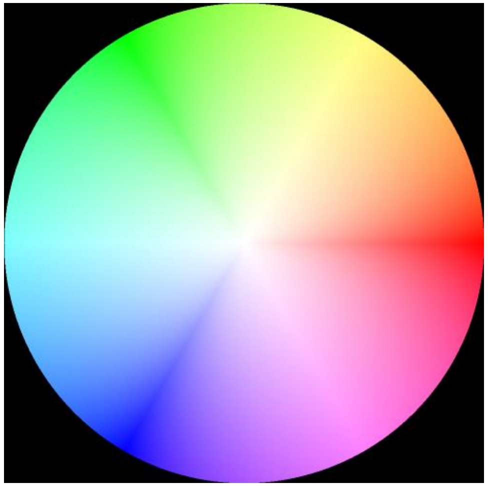
- Retina contains receptors
  - Cones
    - Day vision, can perceive color tone
    - Red, green, and blue cones
  - Rods
    - Night vision, perceive brightness only
- Color sensation
  - Luminance (brightness)
  - Chrominance
    - Hue (color tone)
    - Saturation (color purity)

Monochromatic images

- Image processing - static images -
- Monochromatic static image - continuous image function f(x,y)
  - arguments - two co-ordinates (x,y)
- Digital image functions - represented by matrices
  - co-ordinates = integer numbers
  - Cartesian (horizontal x axis, vertical y axis)
  - OR (row, column) matrices
- Monochromatic image function range
  - lowest value - black
  - highest value - white
- Limited brightness values = gray levels

Chromatic images

- Colour
  - Represented by vector not scalar
    - Red, Green, Blue (RGB)
    - Hue, Saturation, Value (HSV)
    - luminance, chrominance (Yuv , Luv)



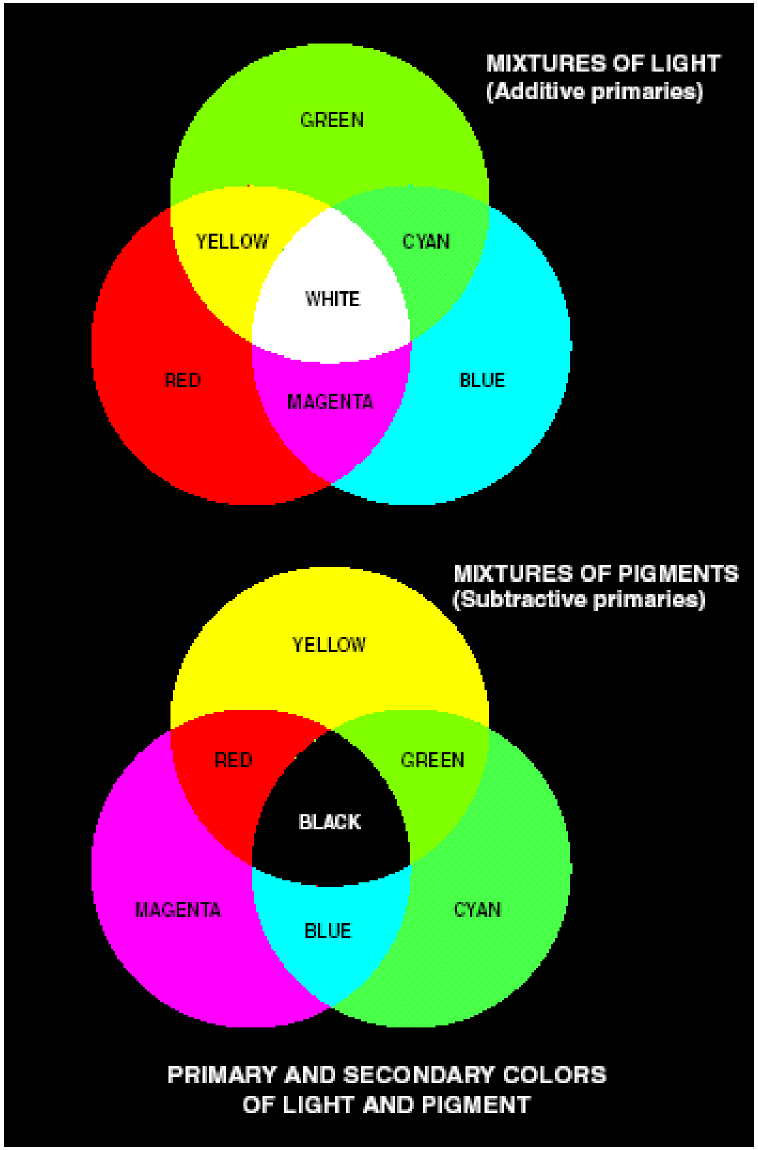
PRIMARY AND SECONDARY COLORS OF LIGHT AND PIGMENTS

- The primary colors can be added to produce the secondary colors of light
- The primary colors of light and primary colors of pigments are different

Magenta = Red + Blue
Cyan = Blue + Green
Yellow = Green + Red

- For pigments, a primary color is defined as one that absorbs a primary color of light and reflects the other two
- Therefore, the primary colors of pigments are magenta, cyan, and yellow

Magenta = White - Green
Cyan = White - Red
Yellow = White - Blue



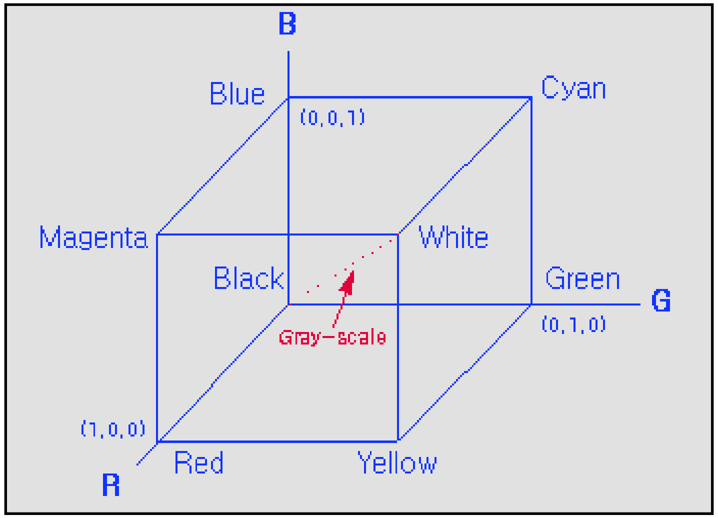
COLOR MODELS

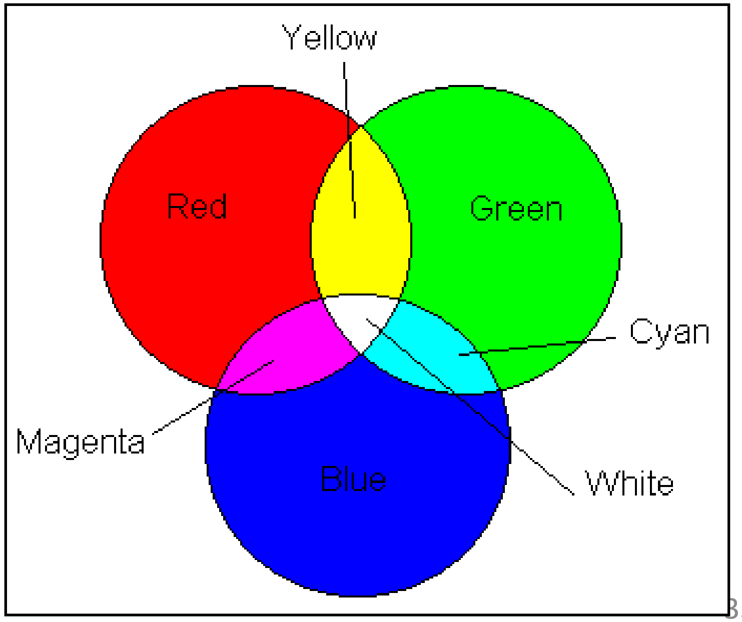
- Color Model
  - Specify colors in a standard way
  - A coordinate system that each color is represented by a single point.
- Most used models:
  - RGB model (Monitor/TV)
  - CMY model (3-color Printers)
  - HSI model (Color Image Processing and Description)

RGB COLOR MODEL

- Pixel Depth: The number of bits used to represent each pixel in RGB space.
- Full-color image: 24-bit RGB color image.
  - (R, G, B) = (8 bits, 8 bits, 8 bits)
  - Number of colors:

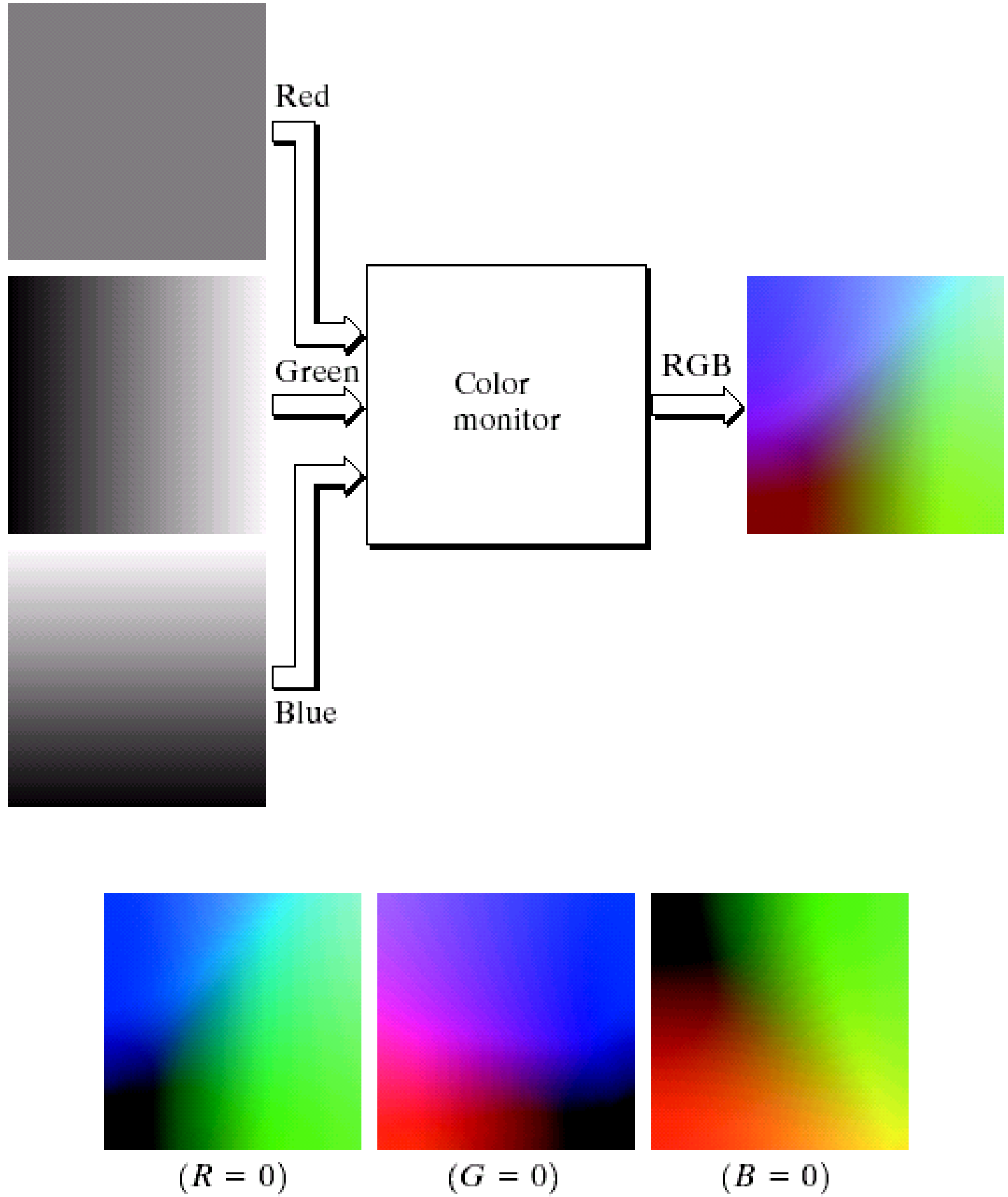
$$\left(2^8\right)^3 = 16,777,216$$





a
b

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



COLOR IMAGE - RGB

Color Image

a b
c d

FIGURE 6.38
(a) RGB image.
(b) Red component image.
(c) Green component.
(d) Blue component.

R-Channel

G-Channel

B-Channel

CMY Model

– Color Printer, Color Copier
– RGB data to CMY

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

HSI COLOR MODEL

- Human description of color is Hue, Saturation and Brightness:
- Hue
  - represents dominant color as perceived by an observer. It is an attribute associated with the dominant wavelength.
- Saturation
  - refers to the relative purity or the amount of white light mixed with a hue. The pure spectrum colors are fully saturated.
    - Pure colors are fully saturated.
    - Pink is less saturated.
- Intensity
  - reflects the brightness.

HSI Color Model

- The HSI model uses three measures to describe colors:
  - Hue: A color attribute that describes a pure color (pure yellow, orange or red)
  - Saturation: Gives a measure of how much a pure color is diluted with white light
  - Intensity: Intensity is the same achromatic notion that we have seen in grey level images

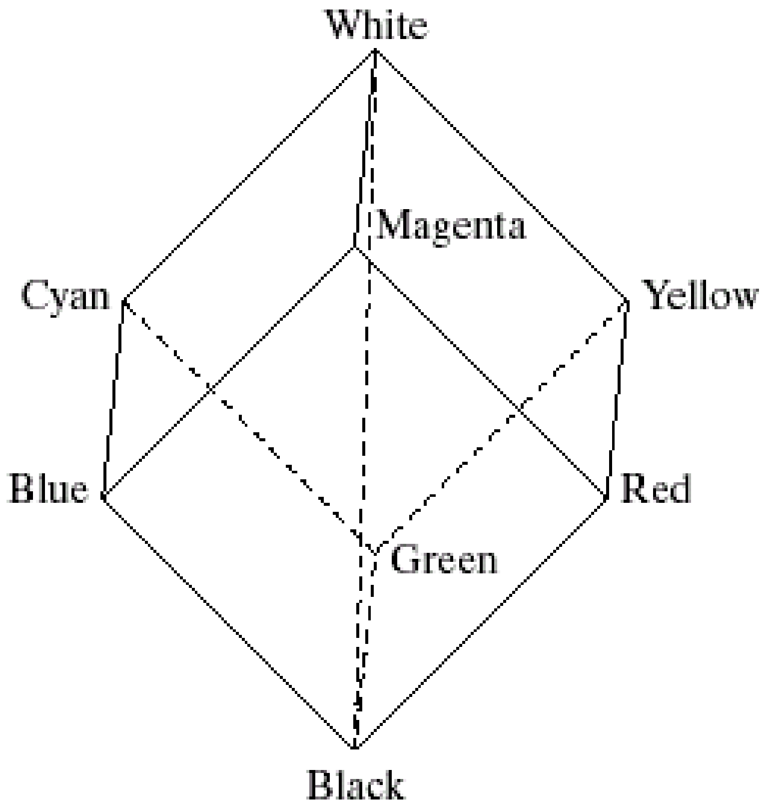
HSI Color Model

- Intensity can be extracted from RGB images
- Remember the diagonal on the RGB color cube that we saw previously ran from black to white
- Now consider if we stand this cube on the black vertex and position the white vertex directly above it

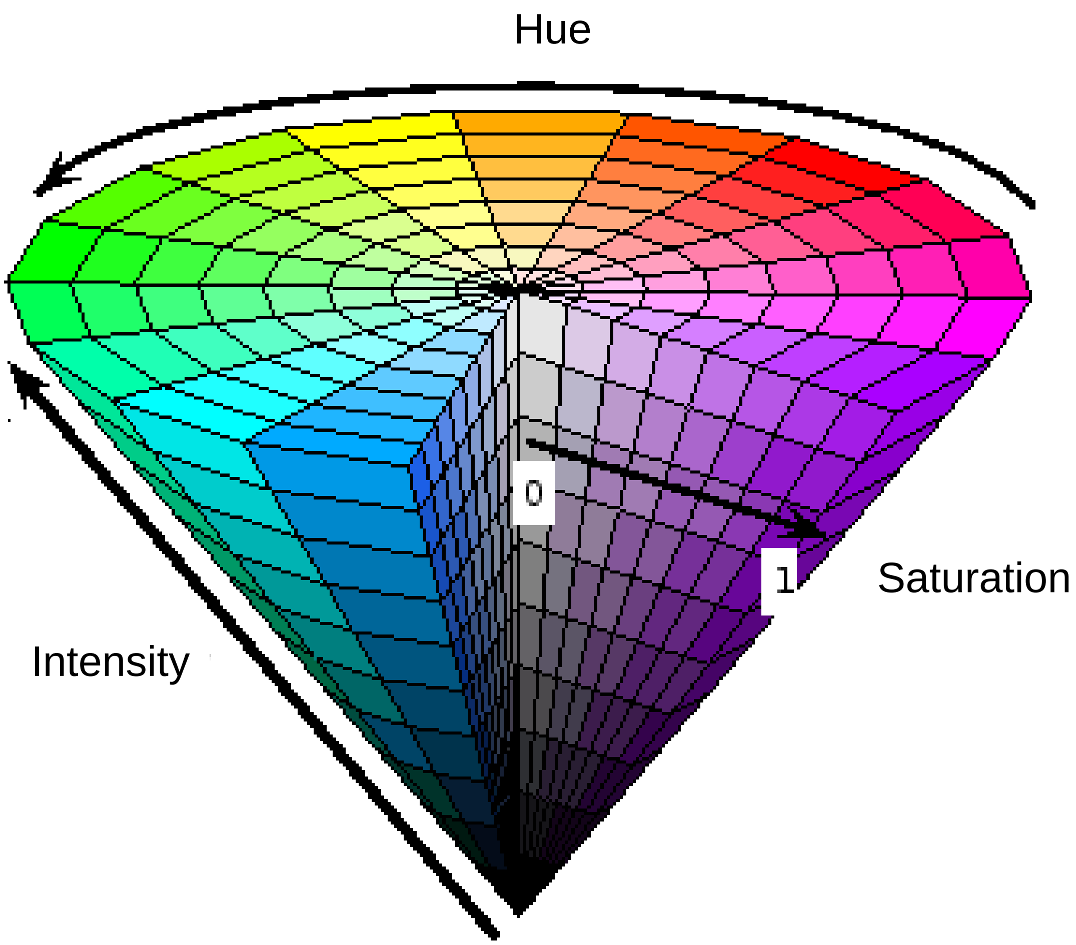
HSI Color Model

- The intensity component of any color can be determined by passing a plane *perpendicular* to the intensity axis and containing the color point

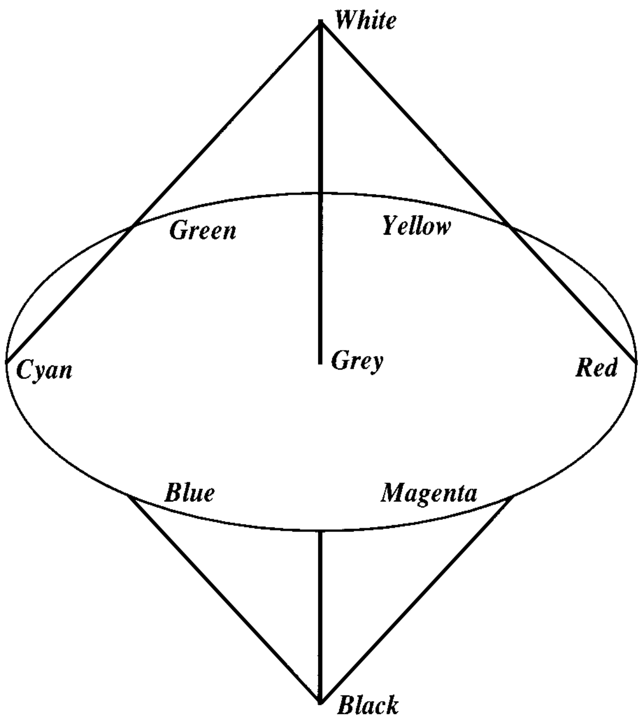
- The intersection of the plane with the intensity axis gives us the intensity component of the color

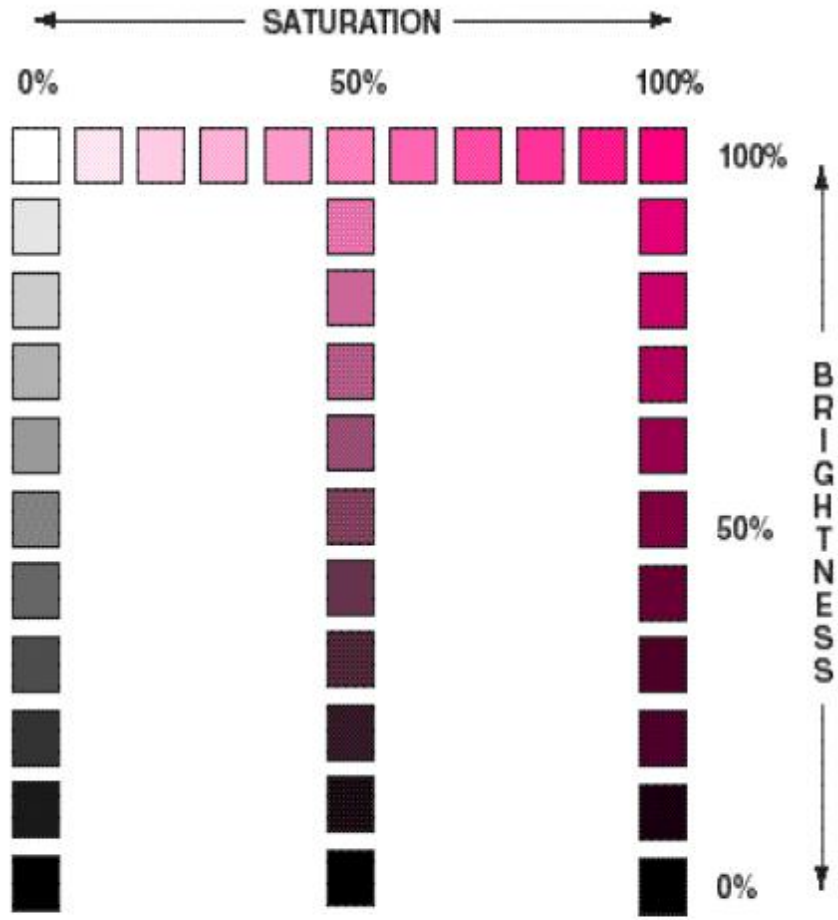


HSI Color Model



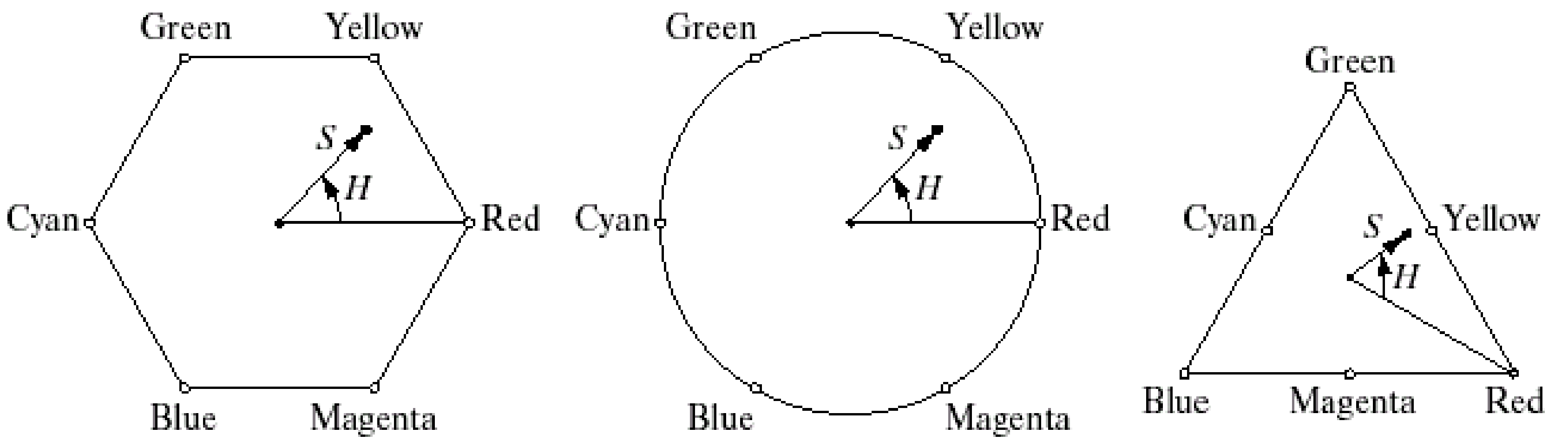
HSI COLOR MODEL- SINGLE HUE



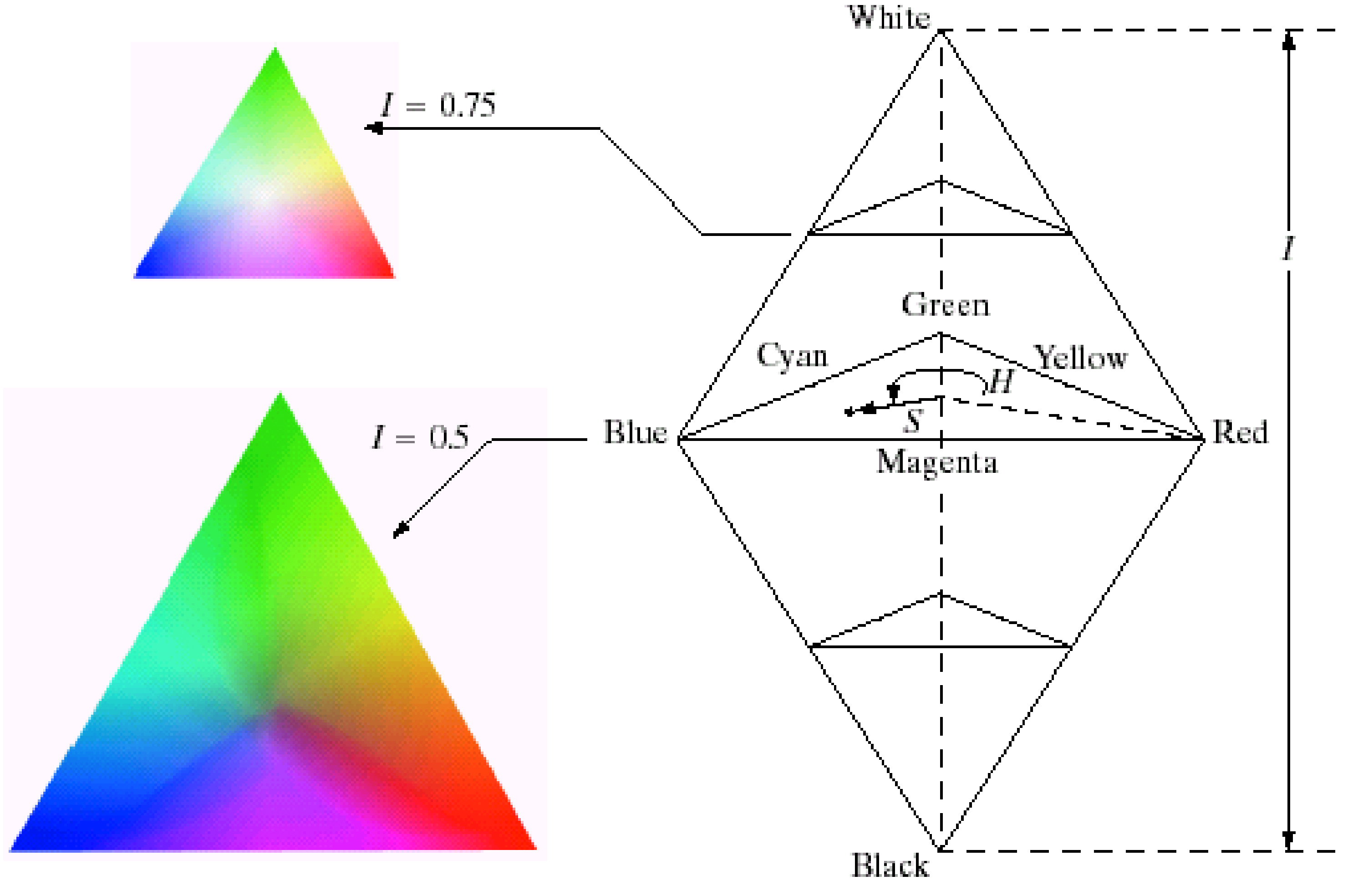


HSI Color Model

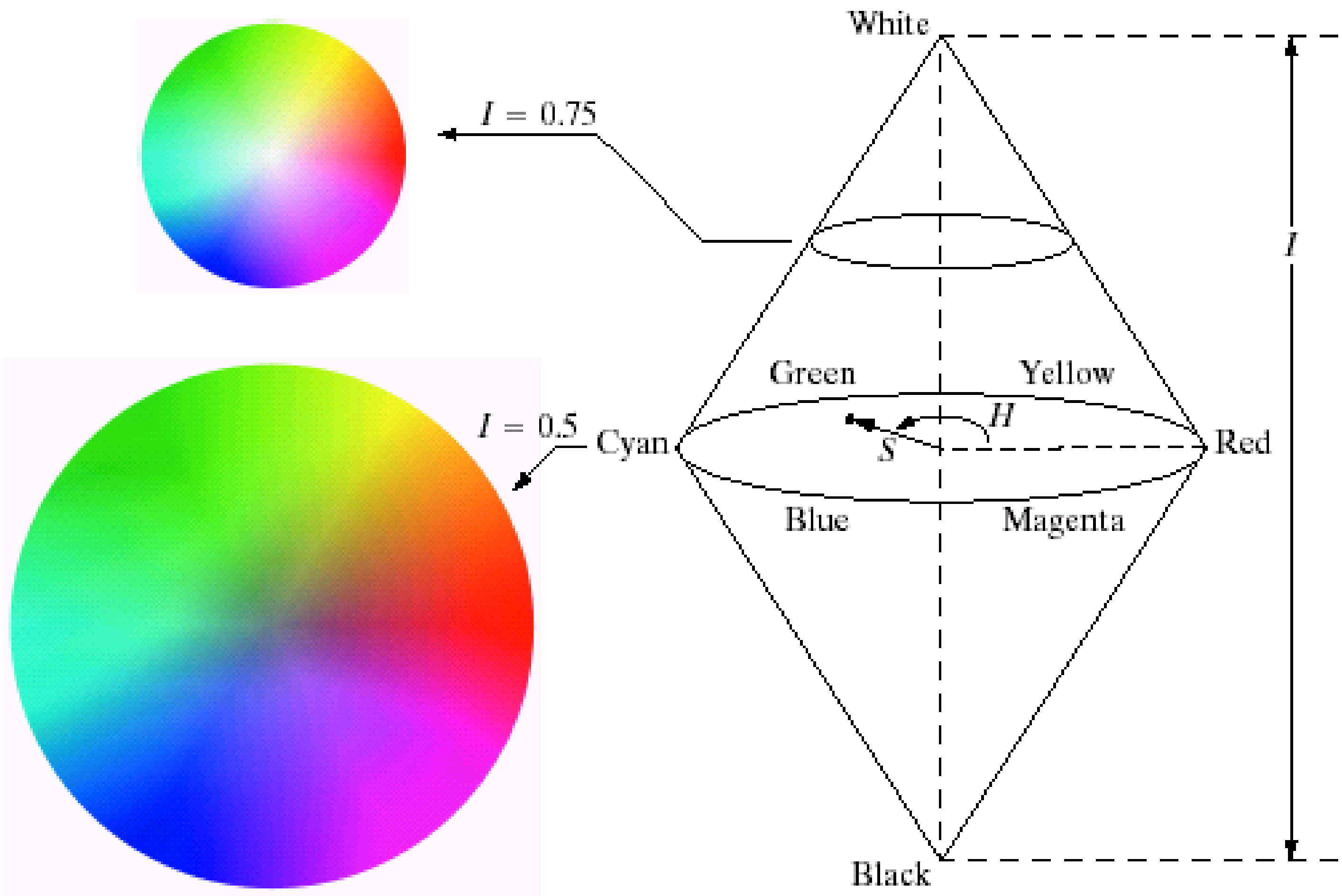
Because the only important things are the angle and the length of the saturation vector this plane is also often represented as a circle or a triangle



HSI Color Model



HSI Color Model



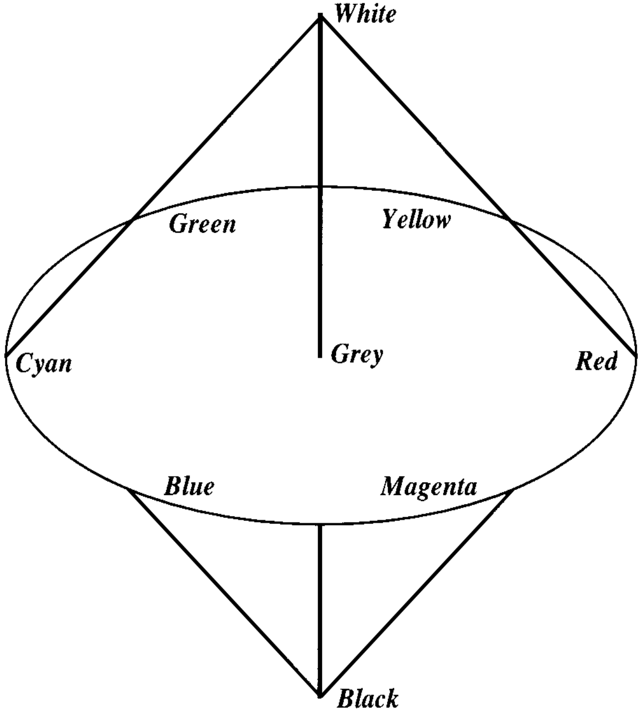
Converting from RGB to HSI

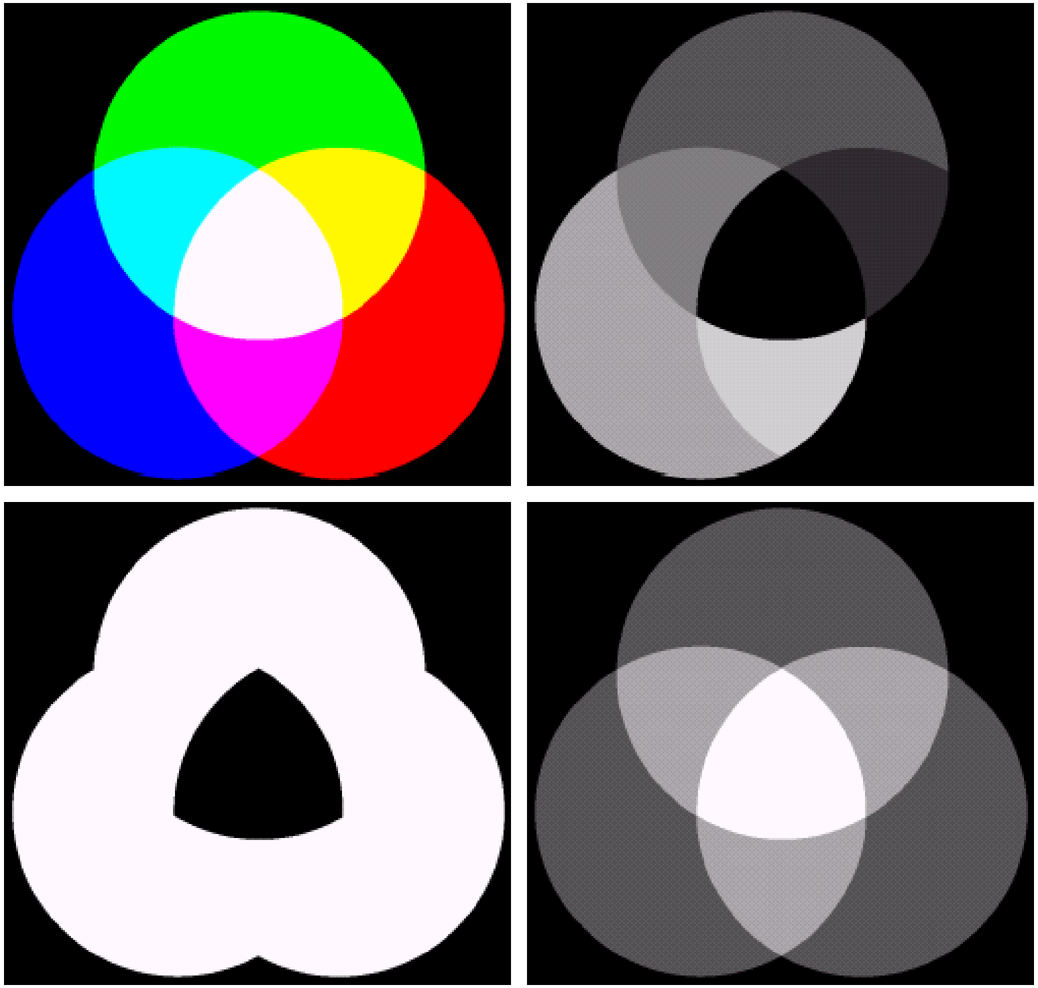
$$\theta = \cos^{-1}\left\{ \frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\left[(R-G)^2+(R-B)(G-B)\right]^{\frac{1}{2}}} \right\}$$

$$H = \begin{cases} \theta & if\, B \le G \\ 360-\theta & if\, B > G \end{cases}$$

$$S = 1 - \frac{3}{(R+G+B)}\left[\min(R,G,B)\right]$$

$$I = \tfrac{1}{3}(R+G+B)$$





a b
c d

FIGURE 6.16 (a) RGB image and the components of its corresponding HSI image: (b) hue, (c) saturation, and (d) intensity.

COLOR IMAGE - HSI

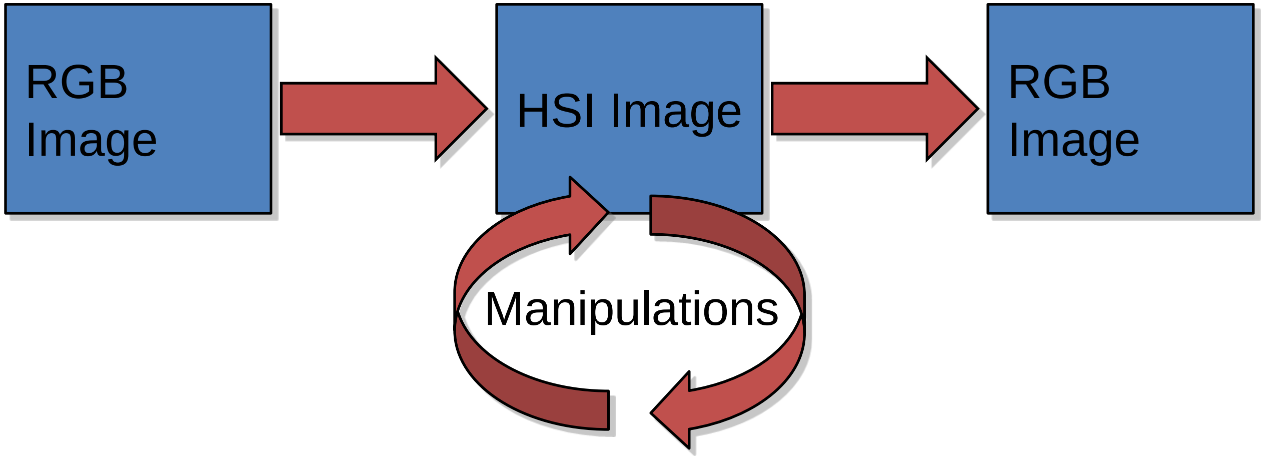
H-Channel S-Channel I-Channel

a b c

FIGURE 6.39 HSI components of the RGB color image in Fig. 6.38(a). (a) Hue. (b) Saturation. (c) Intensity.

Manipulating Images In The HSI Model

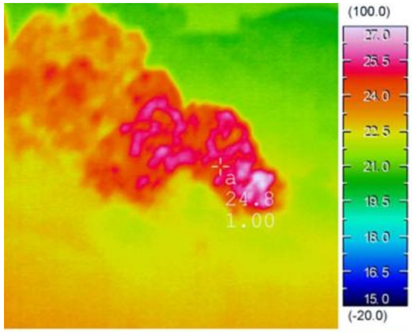
- In order to manipulate an image under the HSI model we:
  - First convert it from RGB to HSI
  - Perform our manipulations under HSI
  - Finally convert the image back from HSI to RGB

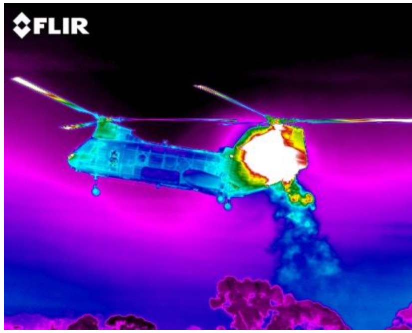


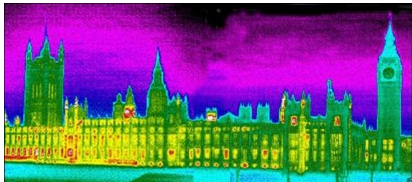
Pseudocolor Image Processing

- Pseudocolor (also called false color) image processing consists of assigning colors to grey values based on a specific criterion

- The principle use of pseudocolor image processing is for human visualization



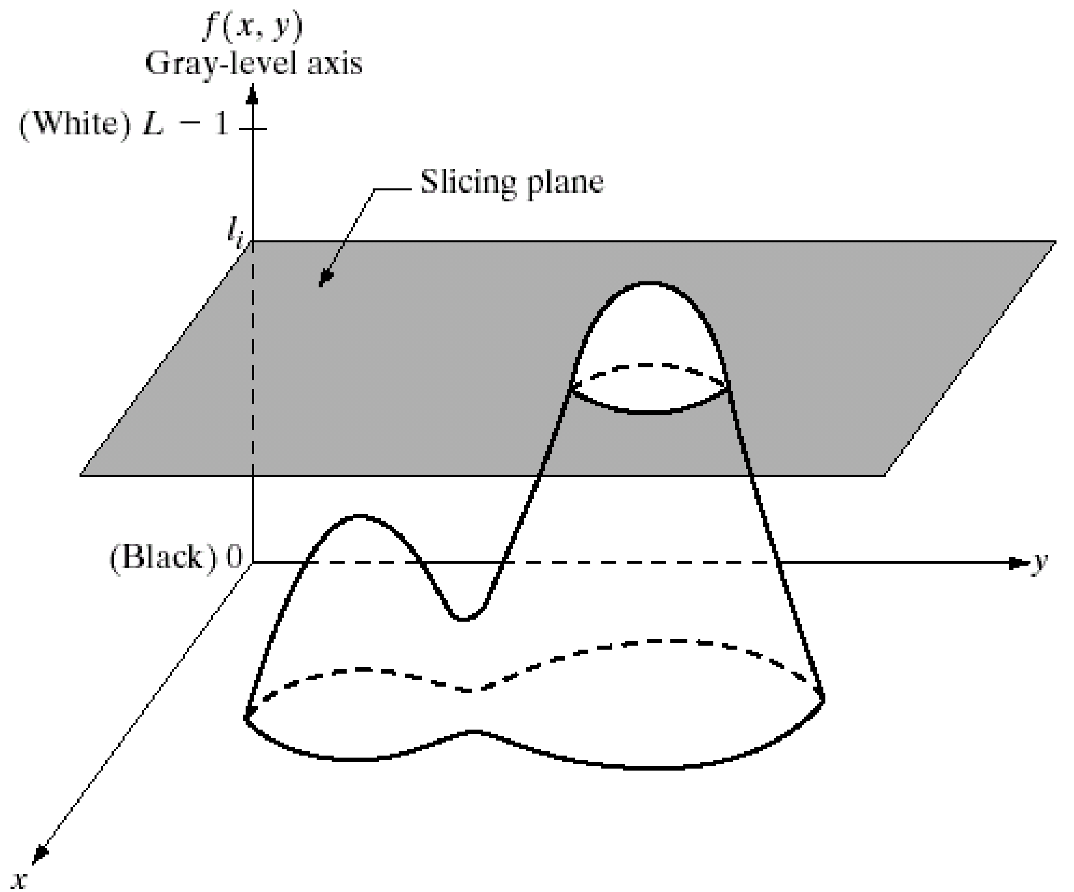




Pseudo Color Image Processing – Intensity Slicing

- Intensity slicing and color coding is one of the simplest kinds of pseudocolor image processing
- First we consider an image as a 3D function mapping spatial coordinates to intensities (that we can consider heights)
- Now consider placing planes at certain levels parallel to the coordinate plane
- If a value is one side of such a plane it is rendered in one color, and a different color if on the other side

Pseudo Color Image Processing – Intensity Slicing



Pseudo Color Image Processing – Intensity Slicing

- In general intensity slicing can be summarized as:
  - Let [$0, L-1$] represent the grey scale
  - Let l_0 represent black [$f(x, y) = 0$] and let l_{L-1} represent white [$f(x, y) = L-1$]
  - Suppose P planes perpendicular to the intensity axis are defined at levels $l_1, l_2, \ldots, l_p$
  - Assuming that $0 < P < L-1$ then the P planes partition the grey scale into $P + 1$ intervals $V_1, V_2, \ldots, V_{P+1}$

Pseudo Color Image Processing – Intensity Slicing

– Grey level color assignments can then be made according to the relation:

$$f(x,y) = c_k \qquad \text{if } f(x,y) \in V_k$$

– where c_k is the color associated with the k^{th} intensity level V_k defined by the partitioning planes at $l = k - 1$ and $l = k$

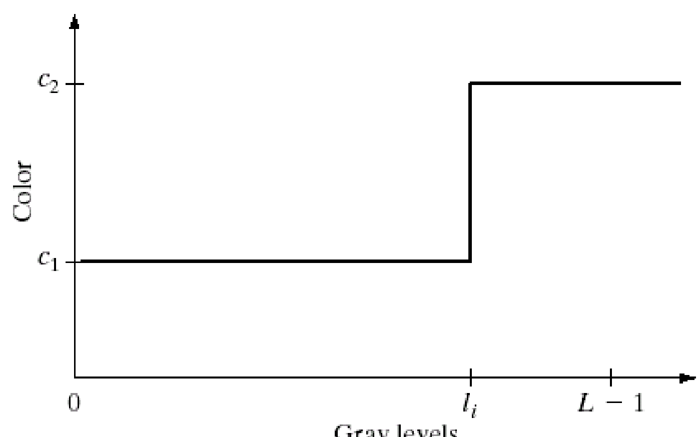
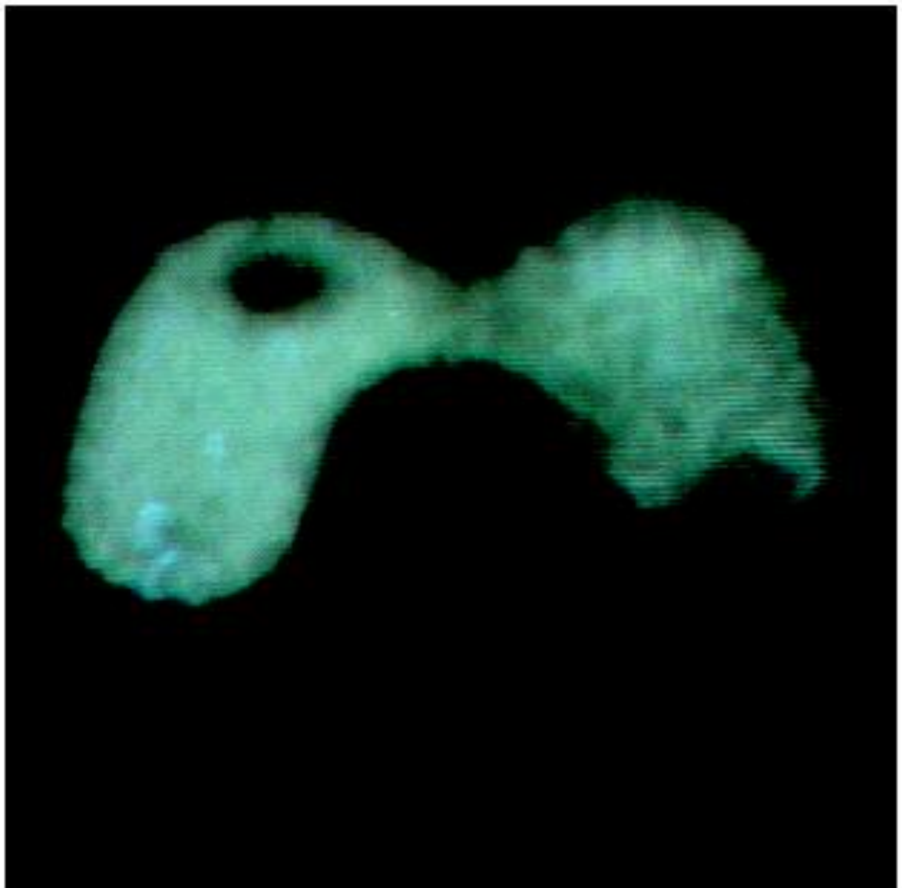
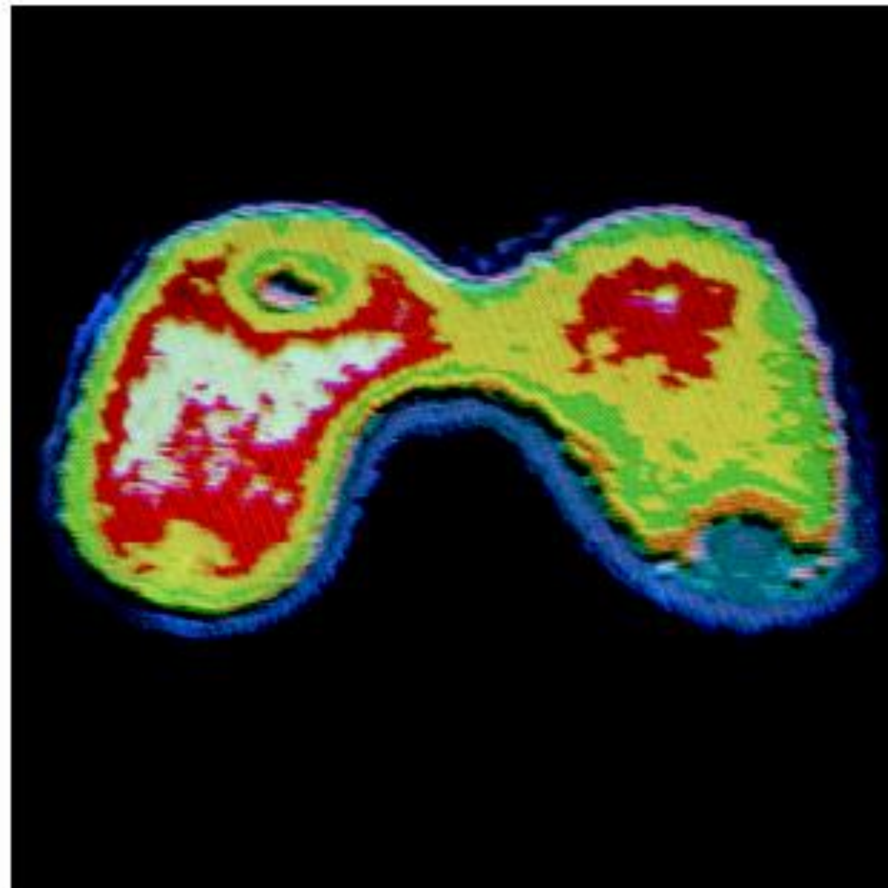


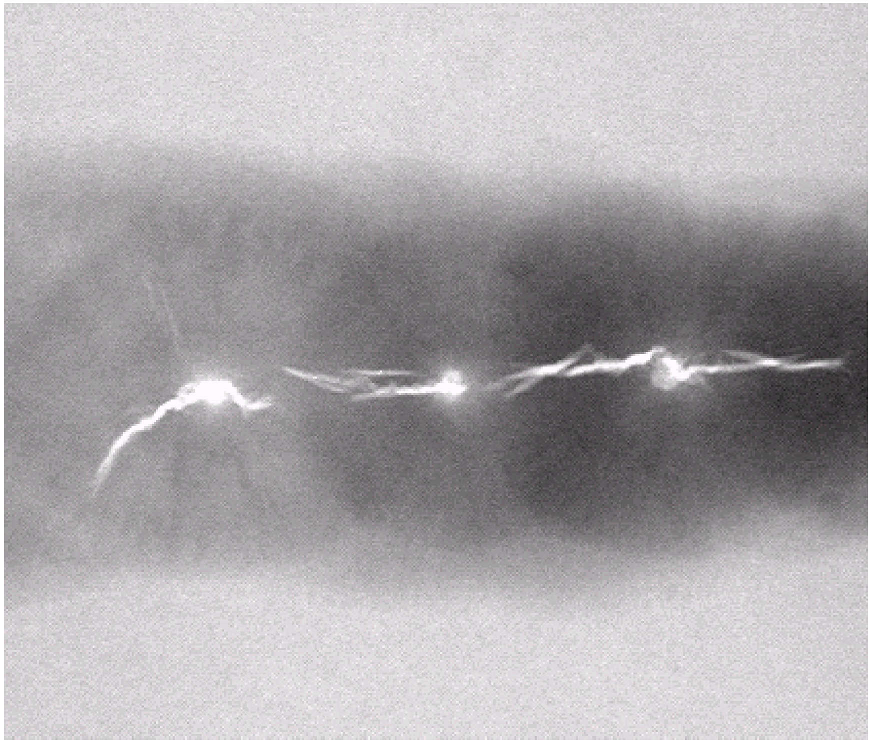
FIGURE 6.19 An alternative representation of the intensity-slicing technique.

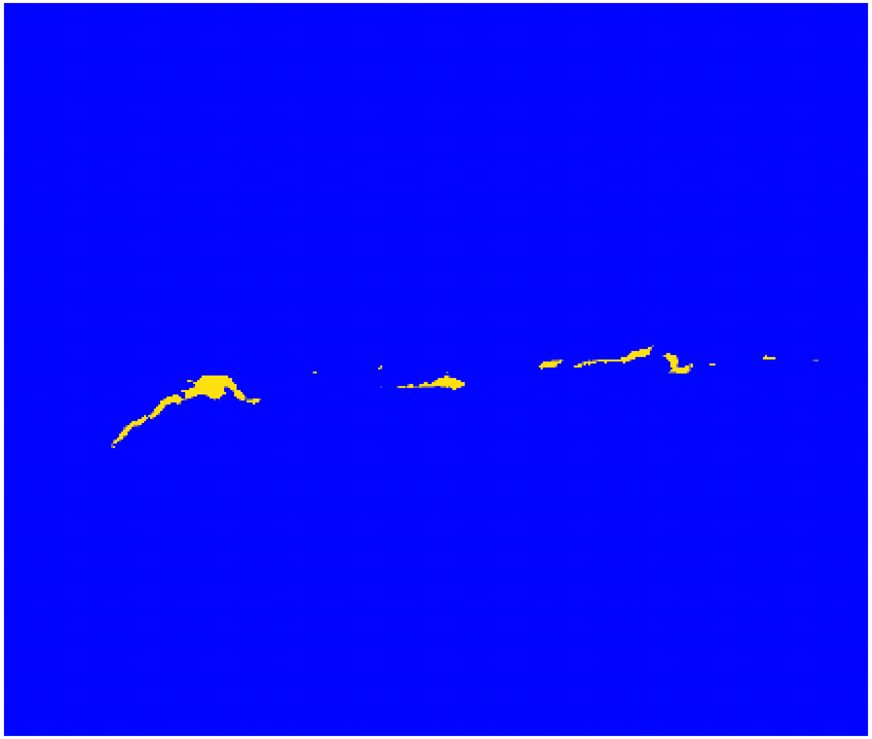
Pseudo Color Image Processing – Intensity Slicing





Pseudo Color Image Processing – Intensity Slicing





COLOR IMAGE - SMOOTHING

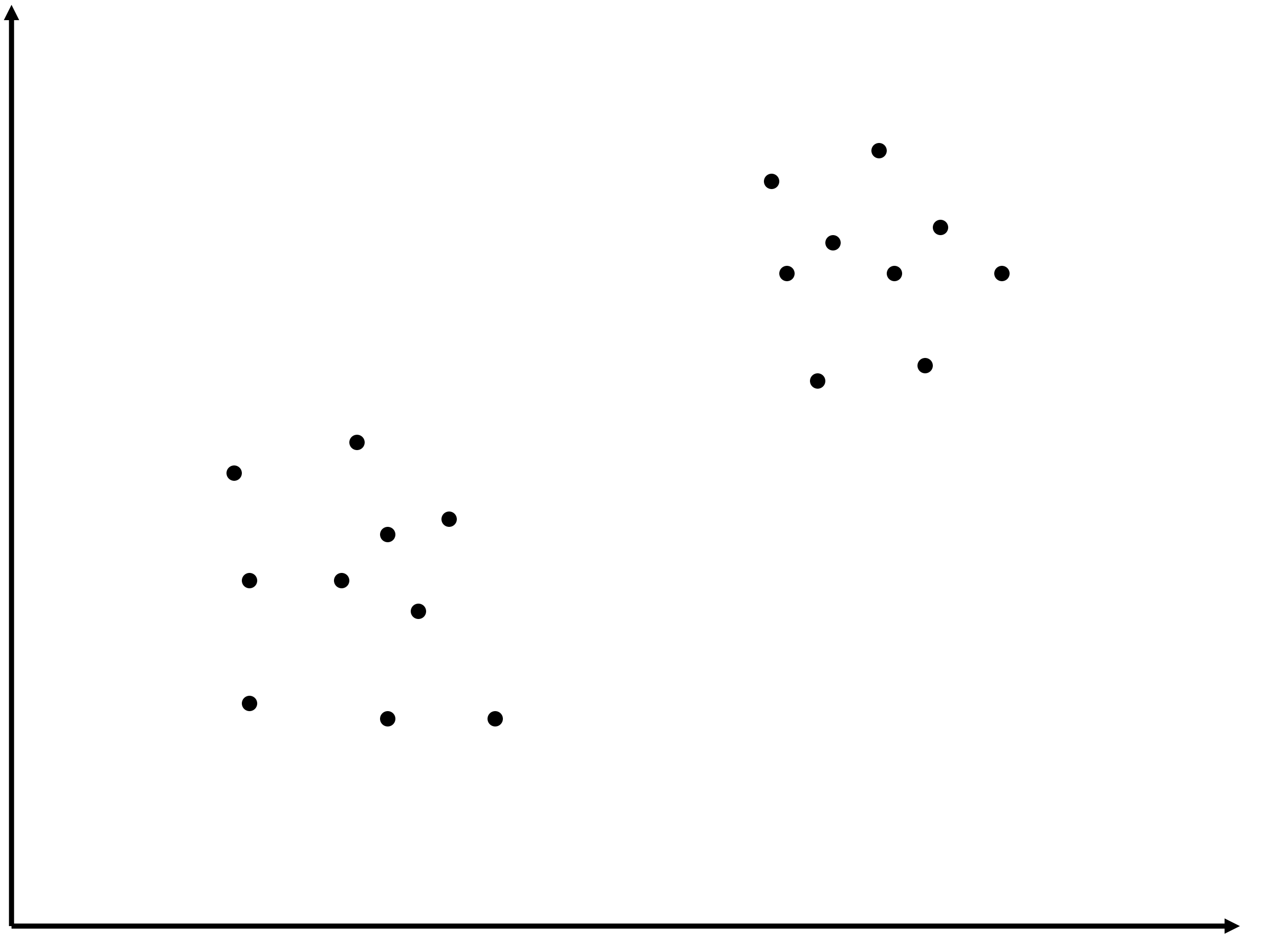
- Smoothing can be viewed as a spatial filtering operation in which the coefficients of the filtering mask are all 1's
- This concept can be easily extended to the processing of full-color images
- Simply smooth each of the RGB color planes and then combine the processed planes to form a smoothed full-color result

$$\hat{\mathbf{c}}(x,y) = \frac{1}{MN}\begin{bmatrix} \sum_{(x,y)\in S_{xy}} R(x,y) \\ \sum_{(x,y)\in S_{xy}} G(x,y) \\ \sum_{(x,y)\in S_{xy}} B(x,y) \end{bmatrix}$$

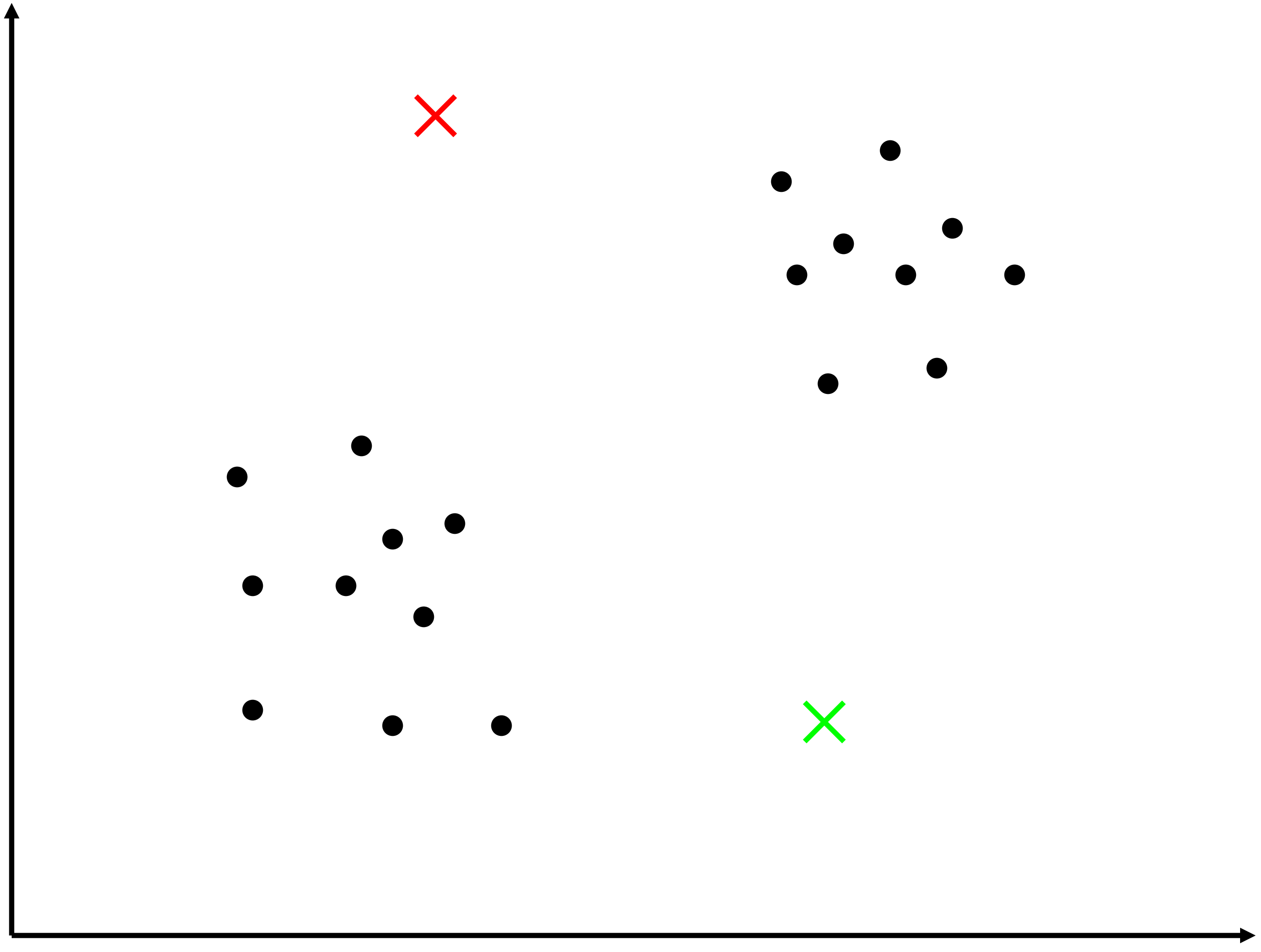
K-Means Clustering

1. Chose the number (K) of clusters and randomly select 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).

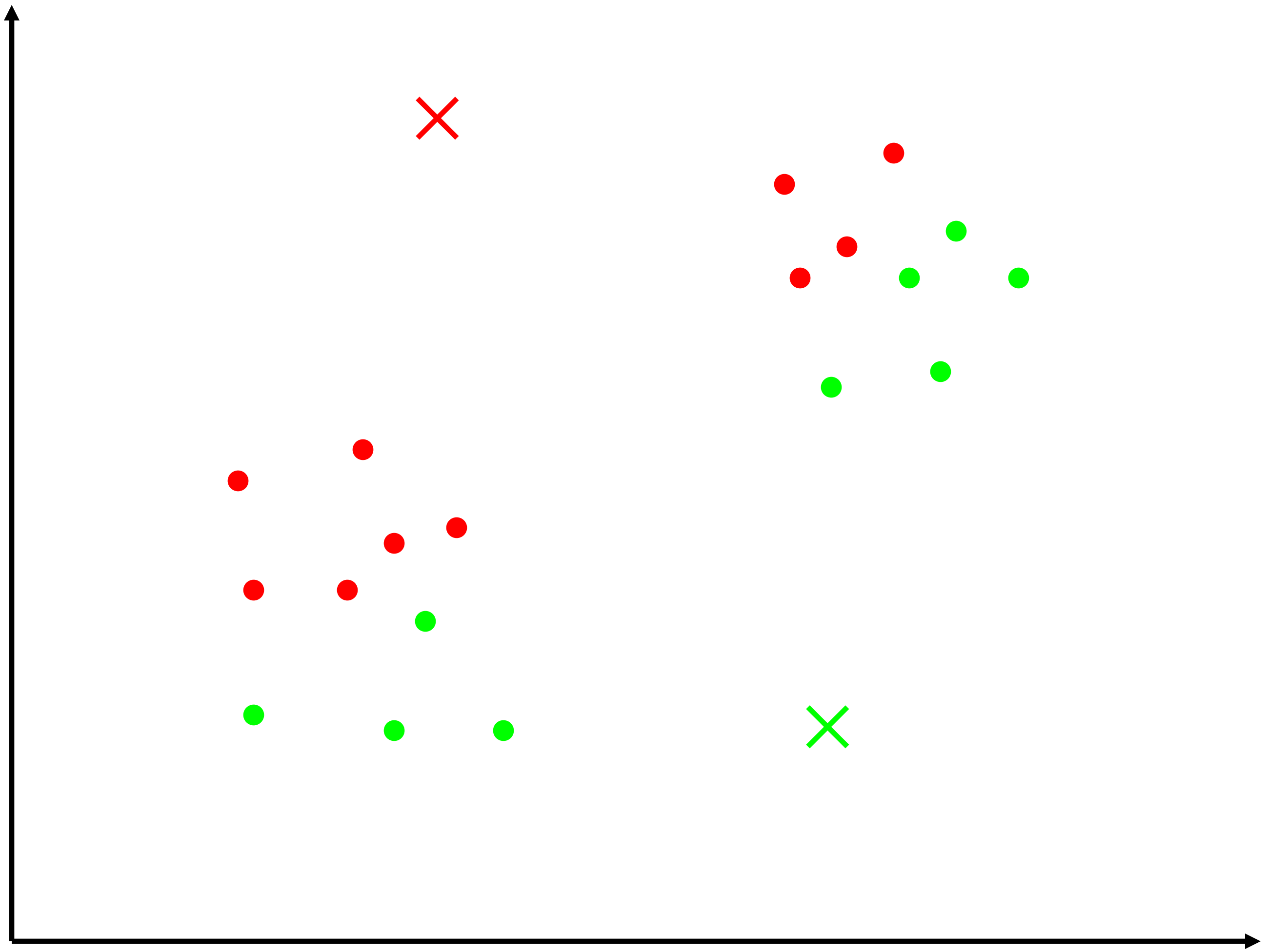
K-Means Clustering



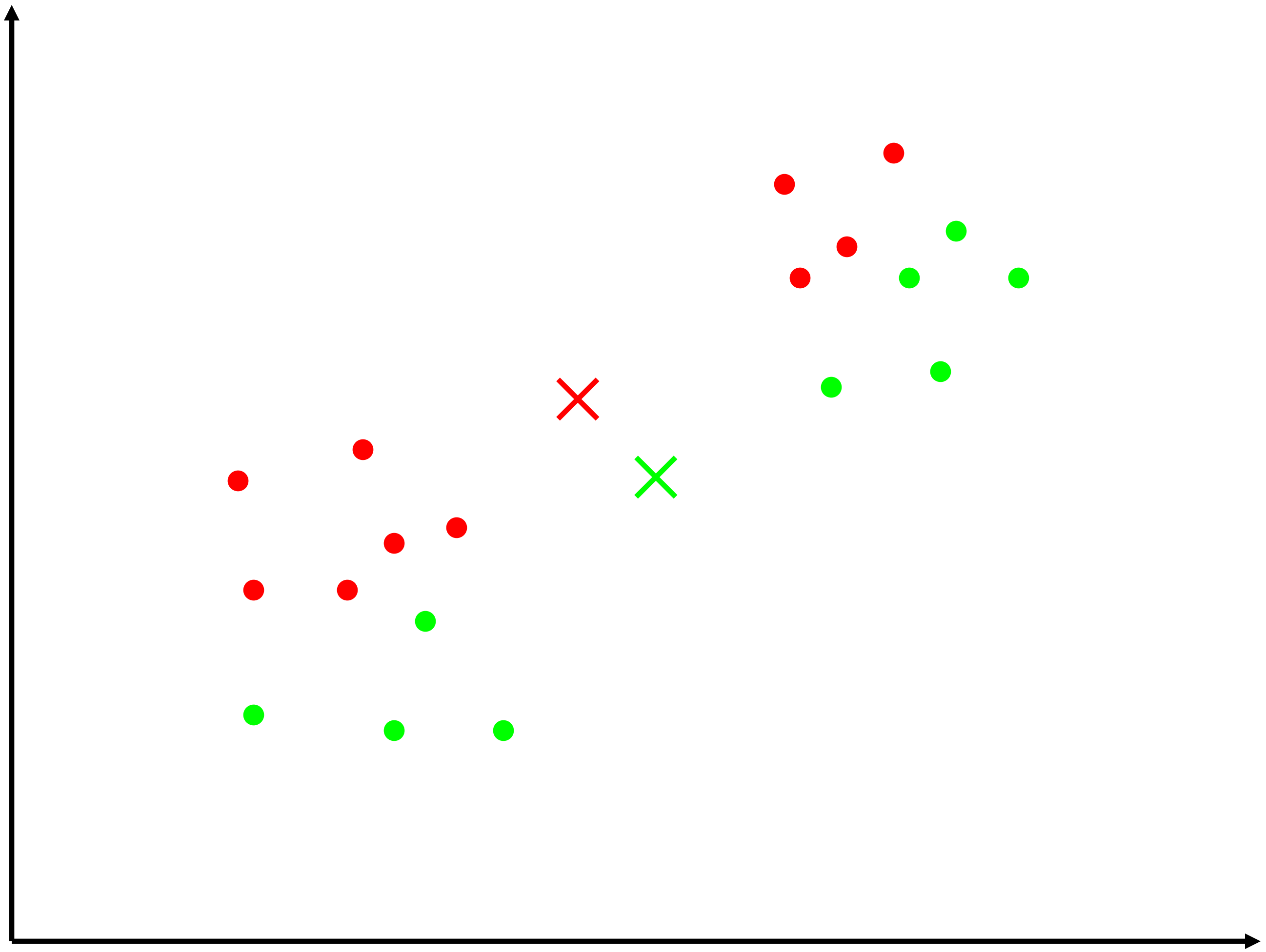
K-Means Clustering



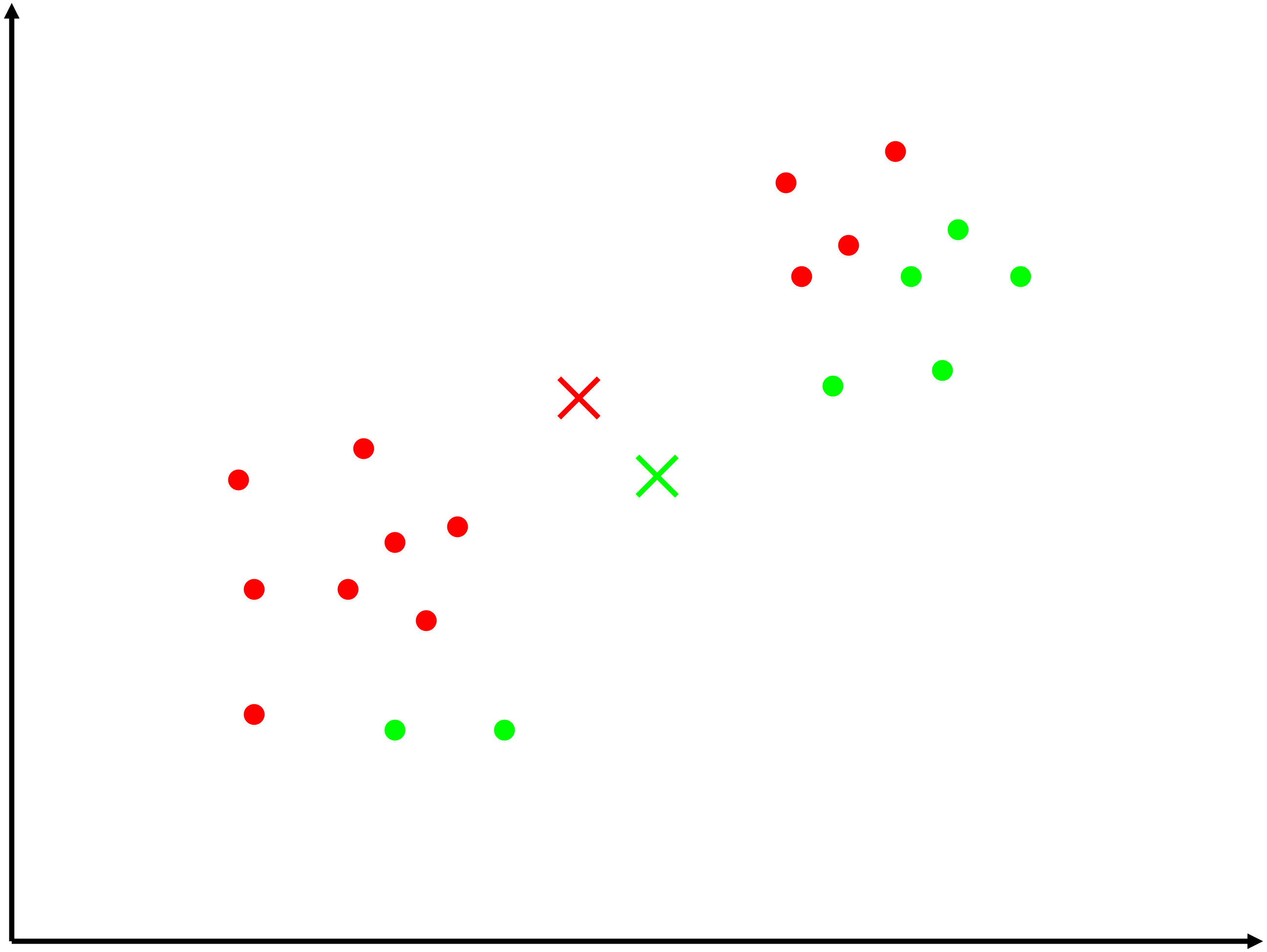
K-Means Clustering



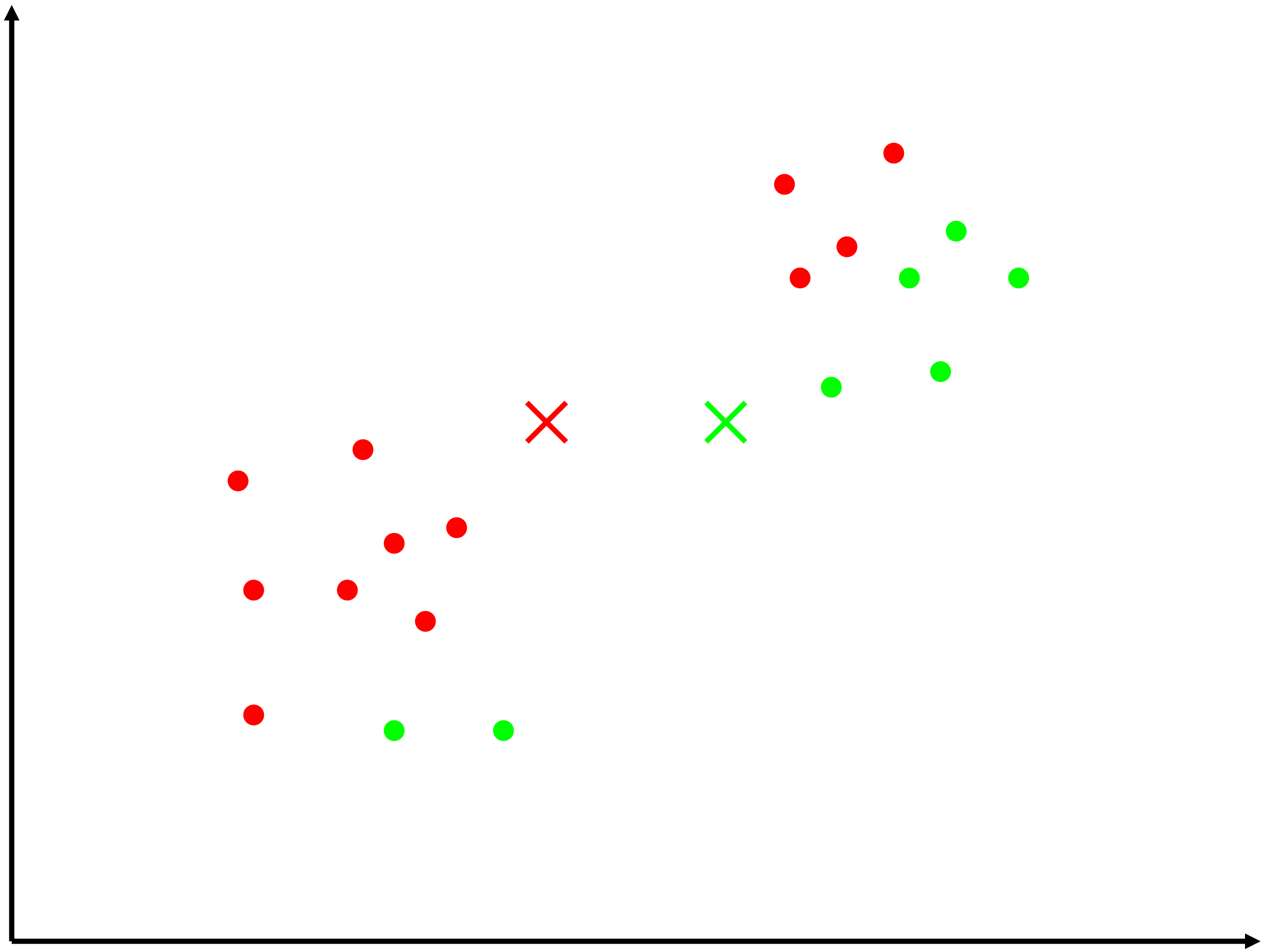
K-Means Clustering



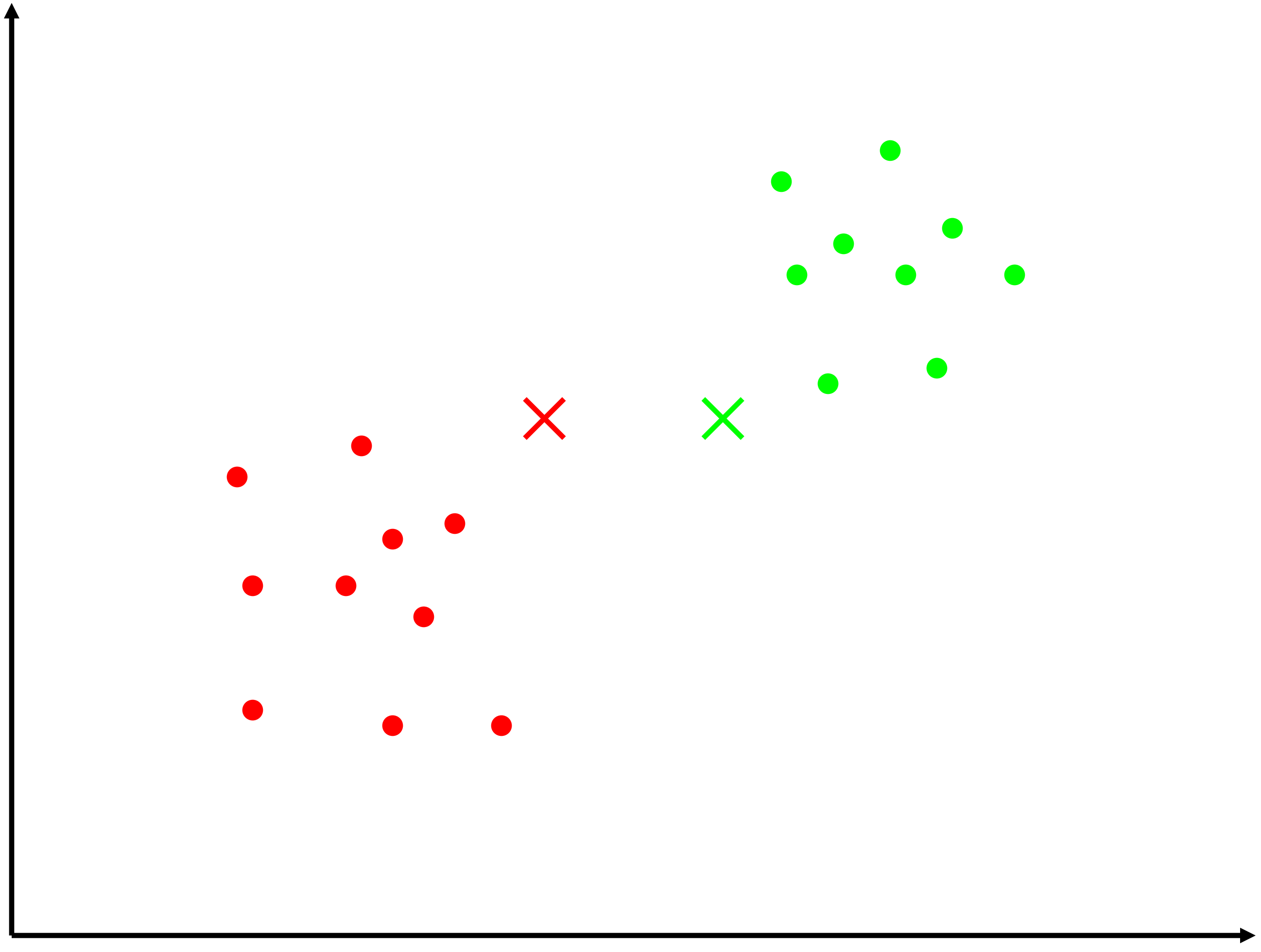
K-Means Clustering



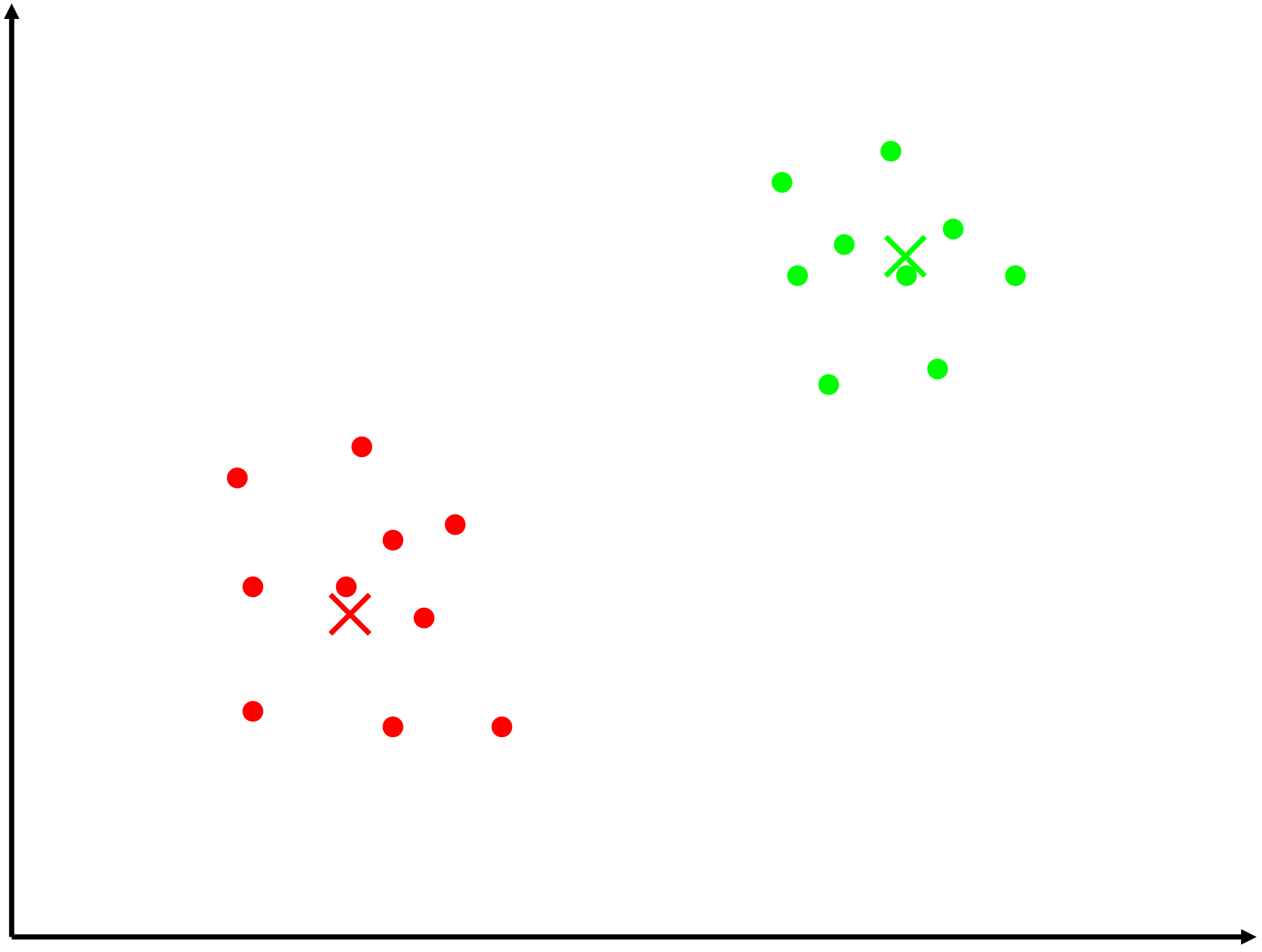
K-Means Clustering



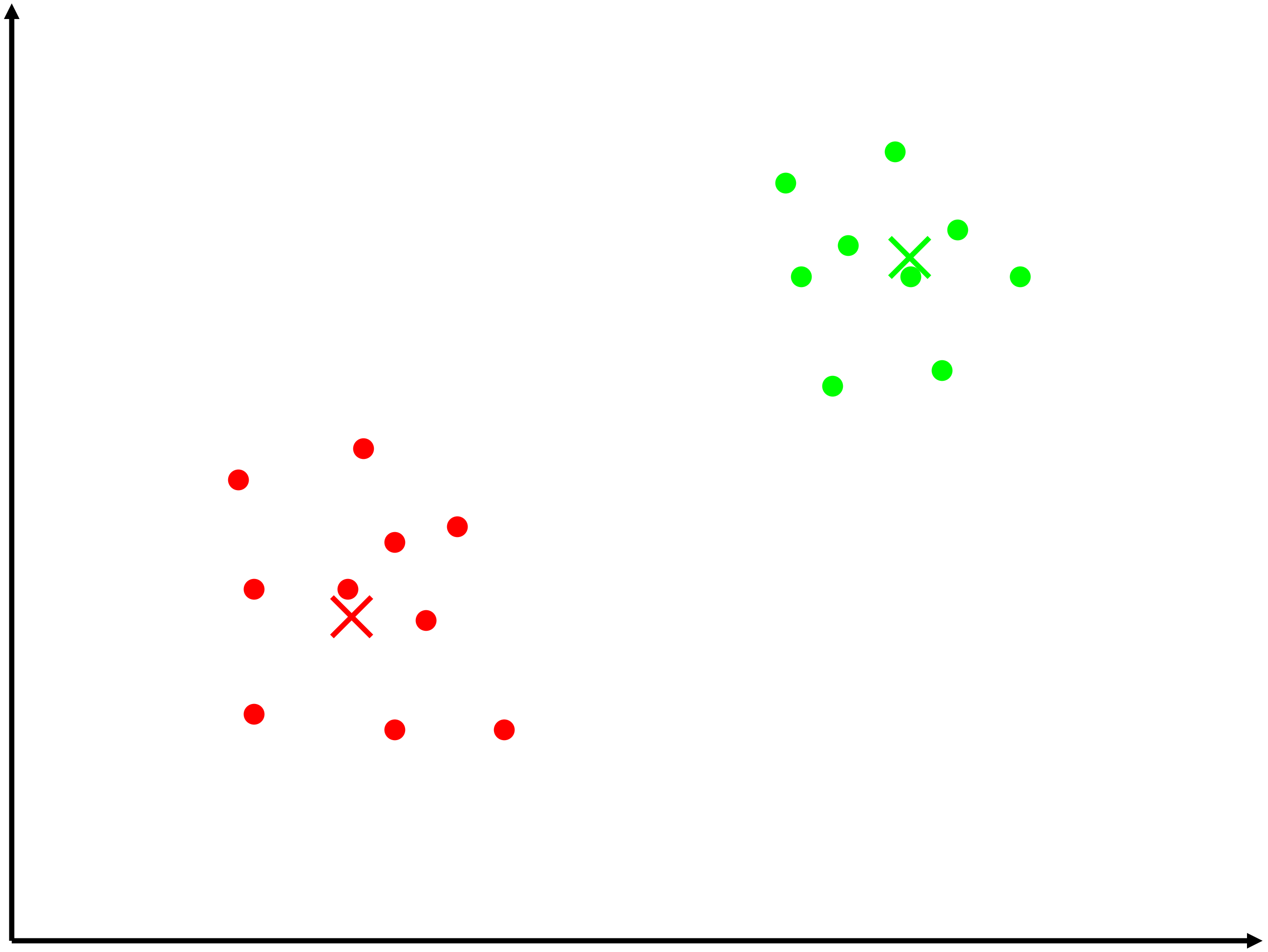
K-Means Clustering



K-Means Clustering

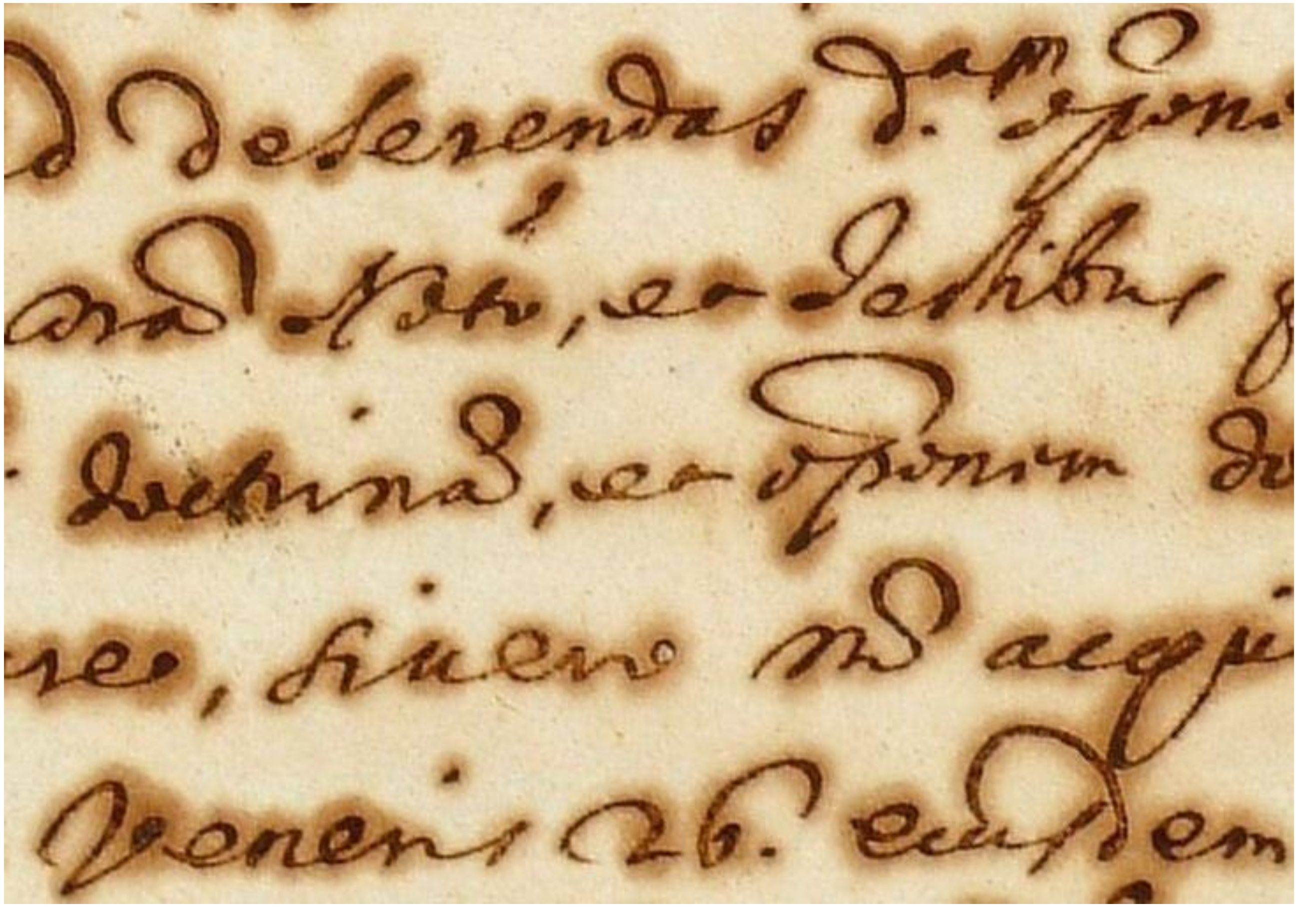


K-Means Clustering



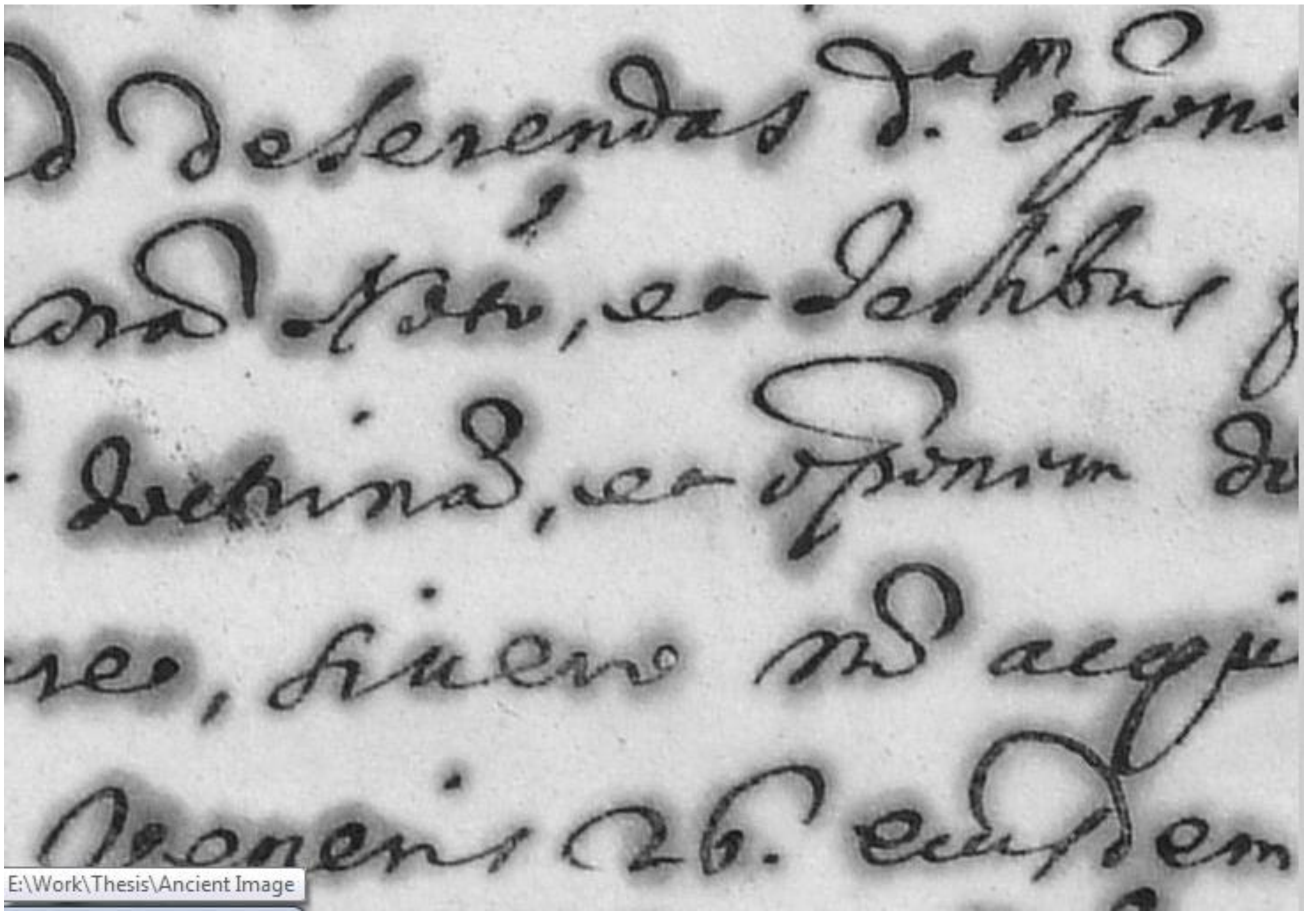
Clustering

- Example



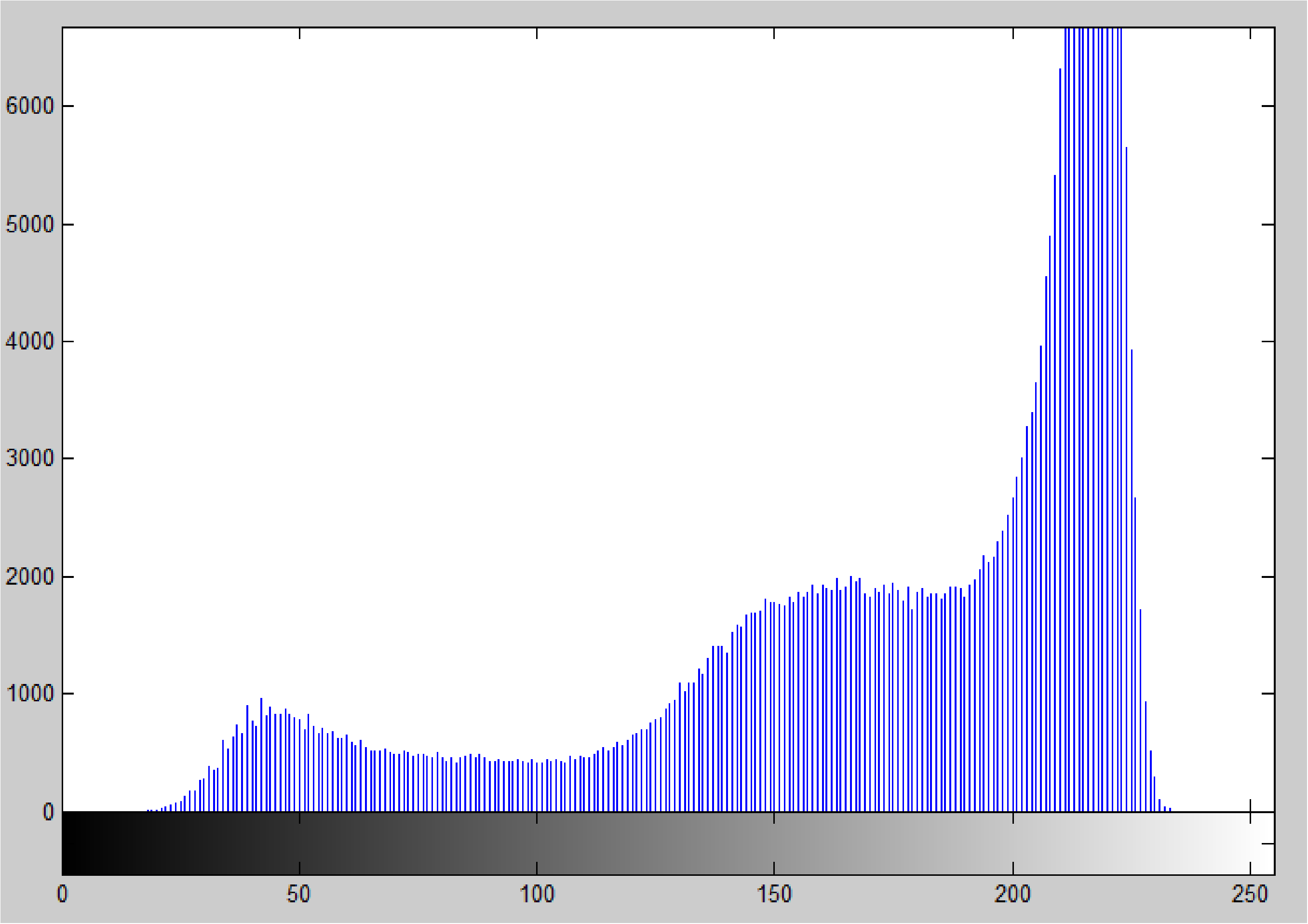
Clustering

- Example



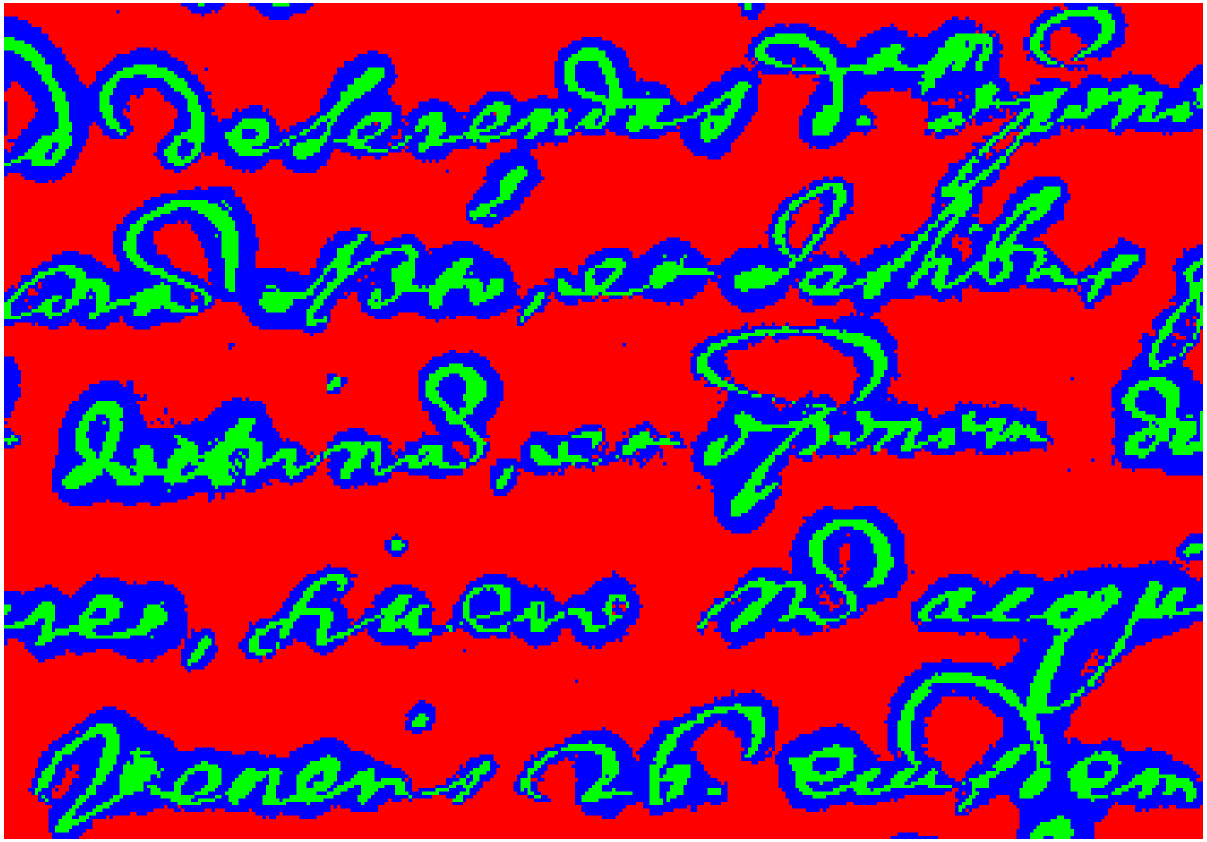
Clustering

- Example



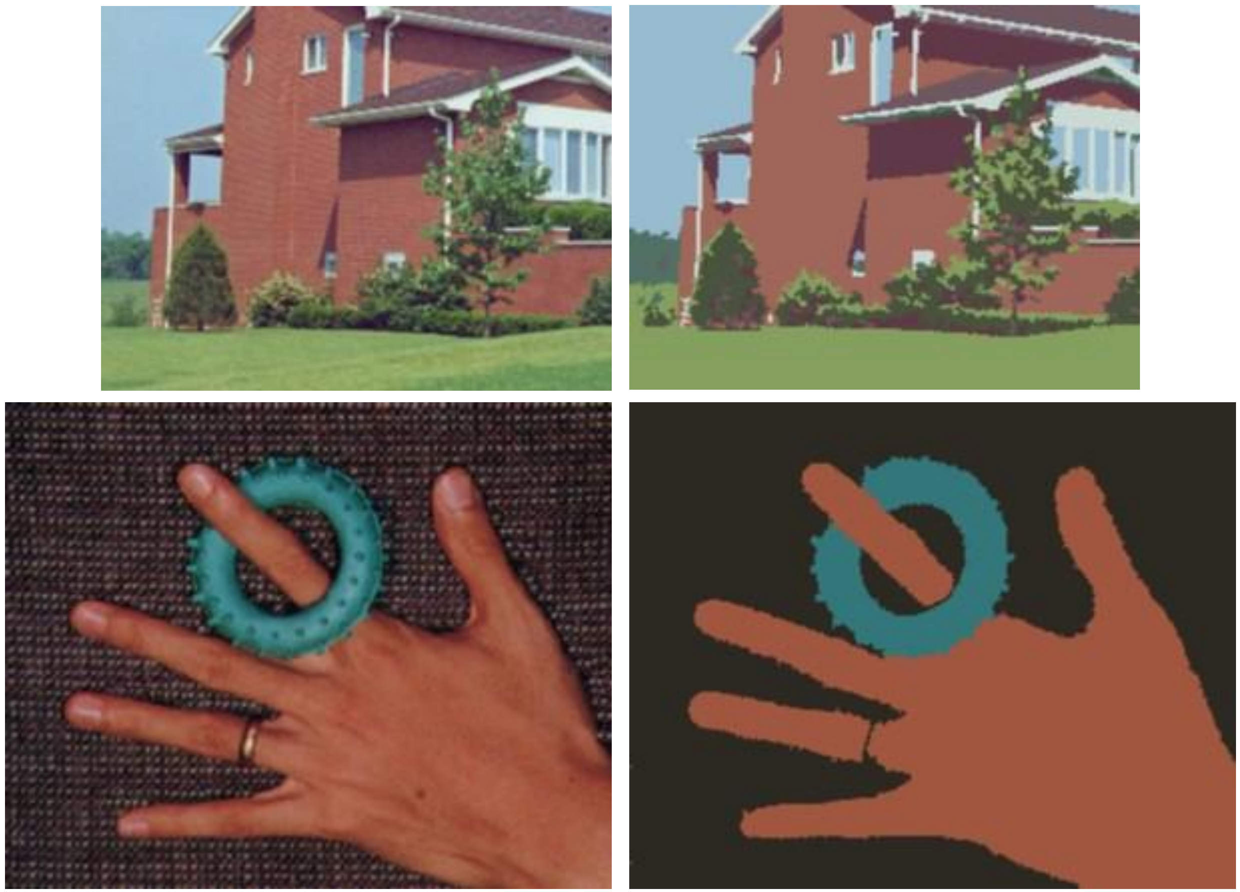
Clustering

- Example



Clustering

- Example



D. Comaniciu and P. Meer, *Robust Analysis of Feature Spaces: Color Image Segmentation*, 1997.

K-Means Clustering

- Example

Original

K=5

K=11

Mean Shift Segmentation Results:

Figure 5.18 Mean-shift color image segmentation with parameters $(h_s, h_r, M) = (16, 19, 40)$ (Comaniciu and Meer 2002) © 2002 IEEE.

Readings from Book (3rd Edn.)

- Color Processing Chapter-6
- Corner Detection: Chapter-4, "Computer Vision: Algorithms and Applications" by Richard Szeliski

Acknowledgements

- Digital Image Processing”, Rafael C. Gonzalez & Richard E. Woods, Addison-Wesley, 2002
- Computer Vision: Algorithms and Applications Richard Szeliski

Material in these slides has been taken from, the following resources