Computer Vision

Course 8

Course 5

Color Image Processing Color Image Processing

Color Image Processing

Color is very important characteristic of an image that in most cases simplifies object identification and extraction form a scene. Human eye can discern thousands of color shades and intensities and only two dozen shades of gray.

Color image processing is divided in 2 major areas: *full-color* (images acquired with a full-color sensor) and *pseudo-color* (gray images for which color is assigned) processing.

(https://youtube.com/shorts/pS5Cvpz7-i4?si=rbWW1pT0Hcsejhmo)

The colors that humans can 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 (390nm to750nm). A body that reflects light that is balanced in all visible wavelengths appears white to the observer. A body that favors reflectance in a limited range of the visible spectrum exhibits some shades of color.

For example, blue objects reflect light with wavelengths from 450 to 475 nm, while absorbing most of the energy of other wavelengths.

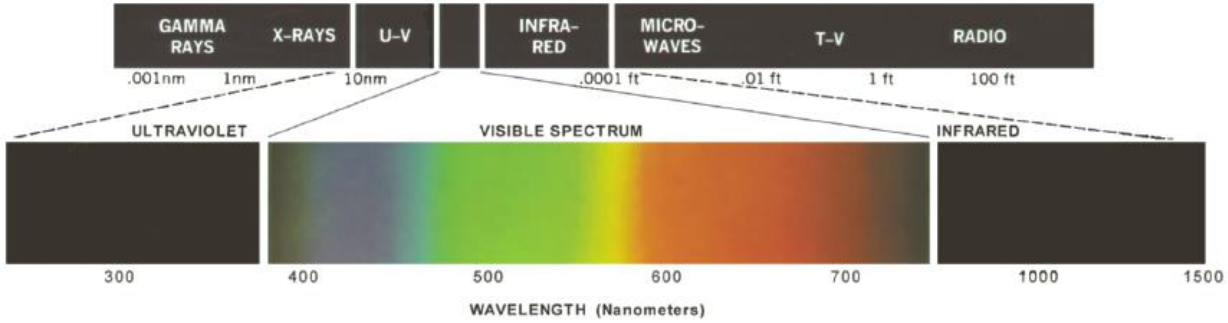


FIGURE 6.2 Wavelengths comprising the visible range of the electromagnetic spectrum. (Courtesy of the General Electric Co., Lamp Business Division.)

Course 5

How to characterize light? If the light is achromatic (void of color) its only attribute is its *intensity* (or amount) – determined by levels of gray (black-grays-white).

Chromatic light spans the electromagnetic spectrum from approximately 400 to 720 nm. Three basic quantities are used to describe the quality of a chromatic light source: radiance, luminance, and brightness.

- *Radiance* is the total amount of energy that flows from the light source (usually measured in watts).
- *Luminance* (measured in lumens – lm) gives a measure of the amount of energy an observer perceives from a light source. For example, the light emitted from a source operating in the infrared region of the spectrum could

have significant energy (radiance), but an observer would hardly perceive it (the luminance is almost zero).

- *Brightness* is a subjective descriptor that cannot be measured, it embodies the achromatic notion of intensity and is a factor describing color sensation.

Cones are the sensors in the eye responsible for color vision. It has been established that the 6 to 7 million cones in the human eye can be divided into three principal sensing categories, corresponding roughly to red, green, and blue. Approximately 65% of all cones are sensitive to red light, 33% are sensitive to green light, an only about 2% are sensitive to blue (but the blue cones are the most sensitive).

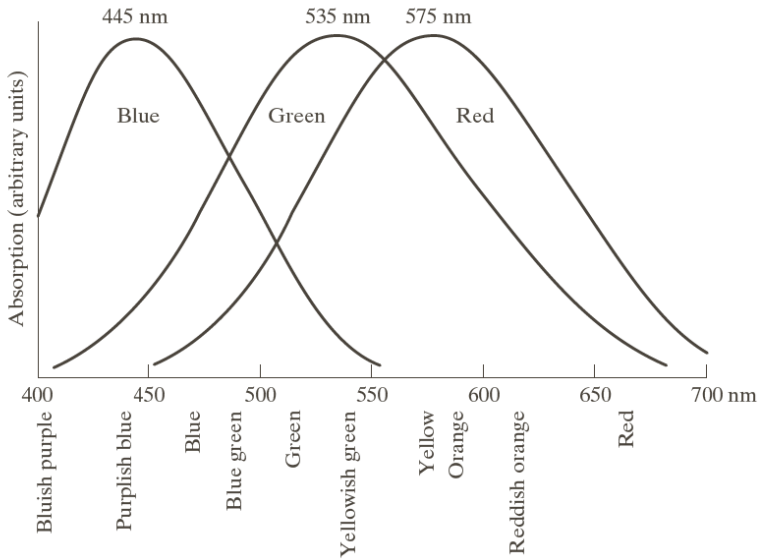


FIGURE 6.3
Absorption of light by the red, green, and blue cones in the human eye as a function of wavelength.

Due to these absorption characteristics of the human eye, colors are seen as variable combinations of the so-called *primary colors*: red (R), green (G), and blue (B).

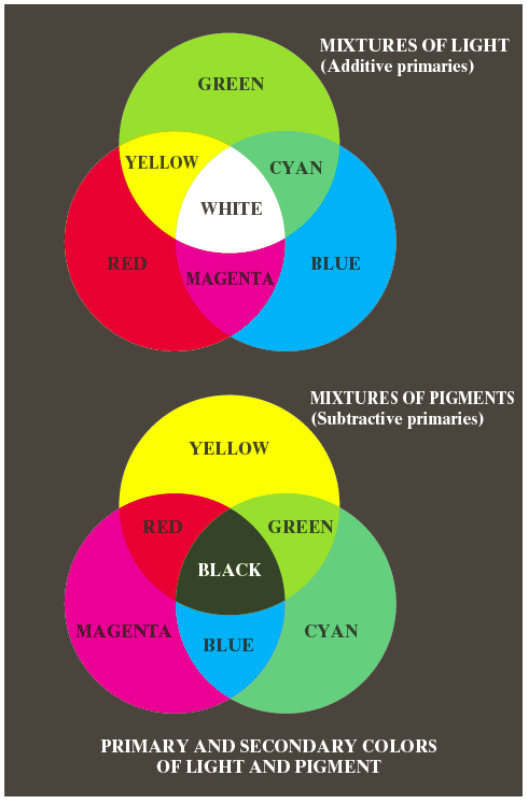
For the purpose of standardization, the CIE (Commission Internationale de l'Eclairage) designated in 1931 the following specific wavelength values to the three primary colors: blue= 435.8 nm, green = 546.1 nm, and red=700 nm. The CIE standards correspond only approximately with experimental data.

These three standard primary colors, when mixed in various intensity proportions, can produce all visible colors.

The primary colors can be added to produce the *secondary* colors of light – magenta (red+blue), cyan (green+blue), and yellow (red+green). Mixing the three primaries, or a secondary with its opposite primary color in the right intensities produces white light.

We must differentiate between the primary colors of light and the primary colors of pigments. A primary color for pigments is one that subtracts or absorb a

primary color of light and reflects or transmits the other two. Therefore, the primary colors of pigments are magenta, cyan, and yellow, and the secondary colors are red, green, and blue.



a
b

FIGURE 6.4 Primary and secondary colors of light and pigments. (Courtesy of the General Electric Co., Lamp Business Division.)

The characteristics usually used to distinguish one color from another are *brightness*, *hue*, and *saturation*.

Brightness embodies the achromatic notion of intensity. *Hue* is an attribute associated with the dominant wavelength in a mixture of light waves. Hue represents dominant color as perceived by an observer (when we call an object to be red, orange or yellow we refer to its hue). *Saturation* refers to the relative purity or the amount of white light mixed with a hue. The pure spectrum colors are fully saturated. Color such as pink (red+white) and lavender (violet+white) are less saturated, with the degree of saturation being inversely proportional to the amount of white light added.

Hue and saturation taken together are called *chromaticity*, and therefore a color may be characterized by its brightness and chromaticity.

The amounts of red, green, and blue needed to form any particular color are called the *tristimulus* values and are denoted ***X, Y*** and ***Z***, respectively. A color is specified by its *trichromatic coefficients*, defined as:

$$x = \frac{X}{X + Y + Z}$$

$$y = \frac{Y}{X + Y + Z}$$

$$z = \frac{Z}{X + Y + Z}$$

$$x + y + z = 1$$

For any wavelength of light in the visible spectrum, the tristimulus values needed to produce the color corresponding to that wavelength can be obtained from the existing curves or tables.

Another approach for specifying colors is to use the CIE *chromaticity diagram*, which shows color composition as a function of ***x*** (red) and ***y*** (green); ***z*** (blue) is obtained from relation ***z = 1-x-y***.

Course 5

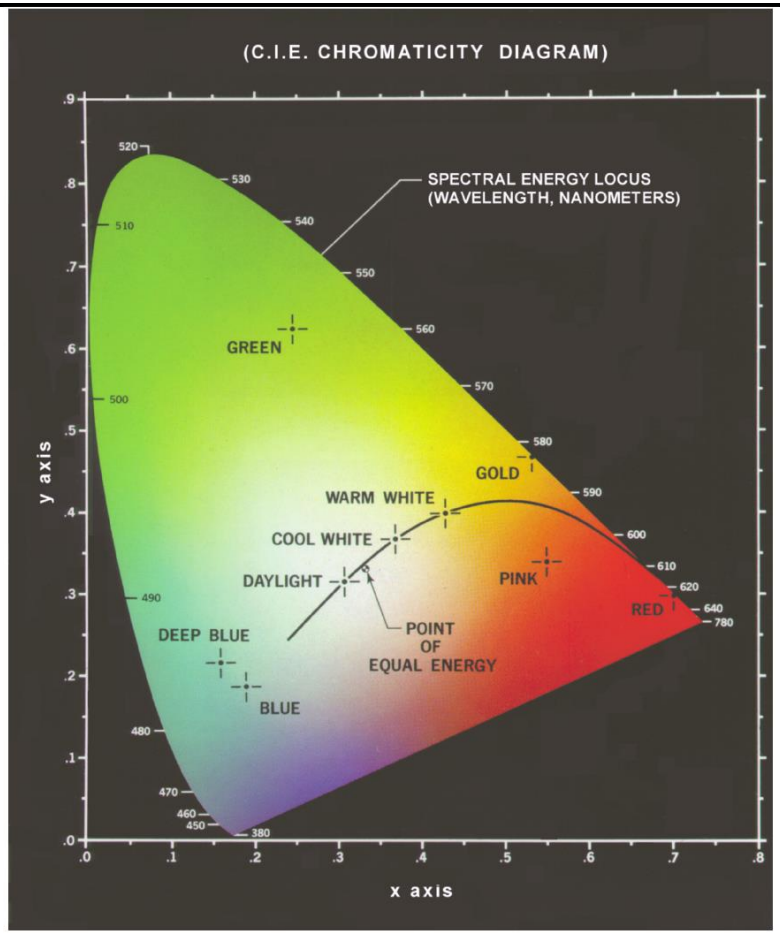


FIGURE 6.5 Chromaticity diagram. (Courtesy of the General Electric Co., Lamp Business Division.)

The positions of the various spectrum colors (from violet at 380 nm to red at 780 nm) are indicated around the boundary of the tongue-shaped chromaticity diagram.

The chromaticity diagram is useful for color mixing because a straight-line segment joining any two points in the diagram defines all the different color variation that can be obtained by combining these two colors. This procedure can be extended to three colors: to triangle determined by the three color-points on the diagram embodies all the possible colors that can be obtained by mixing the three colors.

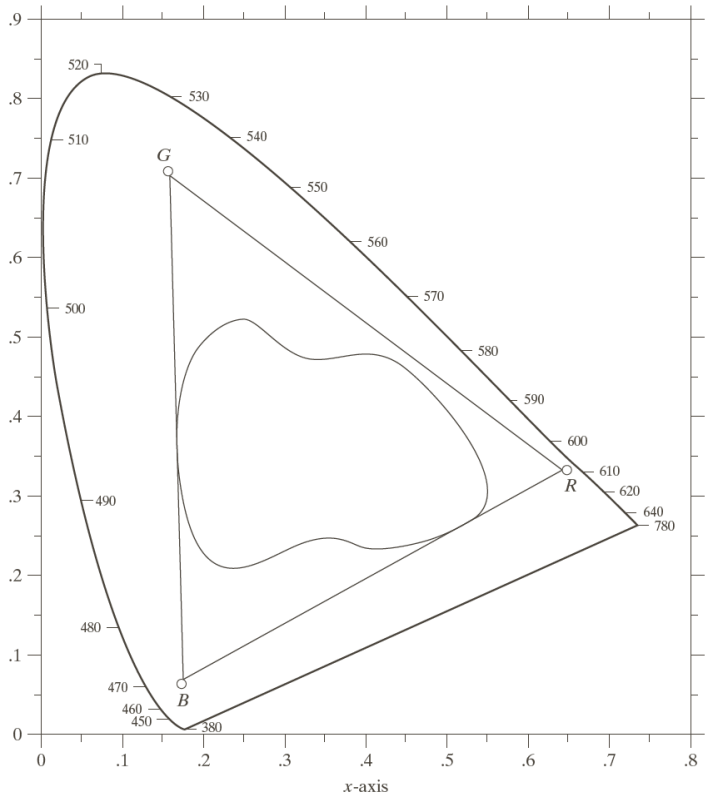


FIGURE 6.6
Typical color gamut of color monitors (triangle) and color printing devices (irregular region).

Course 5

Color Models

A *color model* (*color space* or *color system*) is a specification of a coordinate system and a subspace within that system where each color is represented by a single point.

http://www.colorcube.com/articles/models/model.htm

Most color models in use today are oriented either toward hardware (color monitors or printers) or toward applications where color manipulation is a goal.

The most commonly used hardware-oriented model is **RGB** (red-green-blue) – for color monitors, color video cameras.

The **CMY** (cyan-magenta-yellow) and **CMYK** (cyan-magenta-yellow-black) models are in use for color printing.

The **HSI** (hue-saturation-intensity) model correspond with the way humans describe and interpret colors. The HSI model has the advantage that it decouples the color and gray-scale information in an image, making it suitable for using the gray-scale image processing techniques.

The RGB Color Model

In the RGB model, each color appears decomposed in its primary color components: red, green, blue. This model is based on a Cartesian coordinate system. The color subspace of interest is the unit cube (Figure 6.7), in which the primary and the secondary colors are at the corners; black is at the origin, and white is at the corner farthest from the origin.

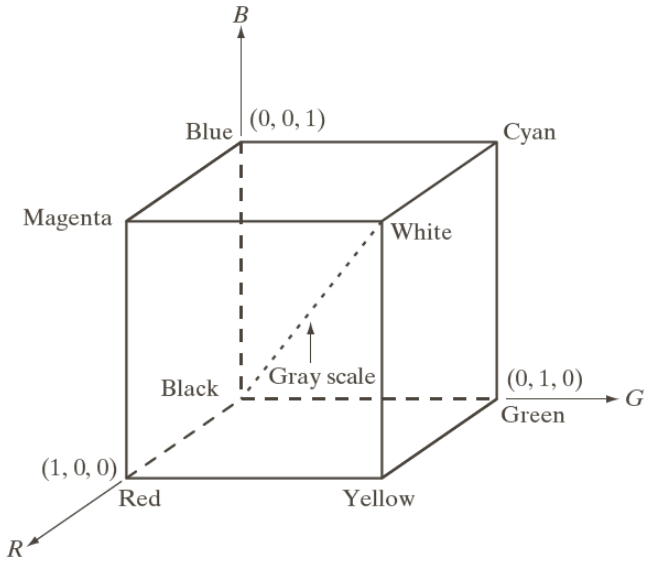


FIGURE 6.7 Schematic of the RGB color cube. Points along the main diagonal have gray values, from black at the origin to white at point (1, 1, 1).

The gray scale (point of equal RGB values) extends from black to white along the line joining these two points. The different colors in this model are points on or inside the cube, and are defined by vectors extending from the origin.

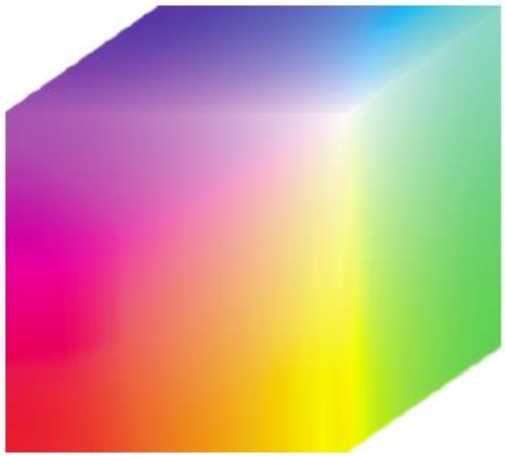
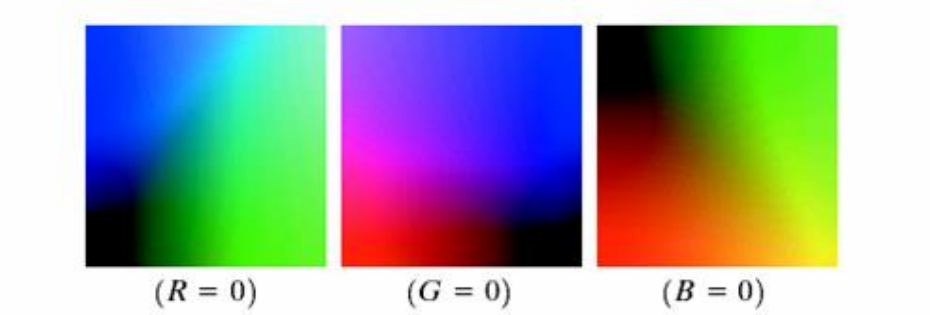


FIGURE 6.8 RGB 24-bit color cube.

Images represented in the RGB color model consist of three component images, one for each primary color. The number of bits used to represent each pixel in RGB space is called the *pixel depth*. Consider an RGB image in which each of the red, green, and blue images are an 8-bit image. In this case, each RGB color pixel has a depth of 24 bits. The term *full-color* image is used often to denote a 24-bit RGB color image. The total number of colors in a 24-bit RGB image is

$$\left(2^8\right)^3 = 16.777.216$$

A convenient way to view these colors is to generate color planes (faces or cross sections of the cube).



A color image can be acquired by using three filters, sensitive to red, green, and blue.

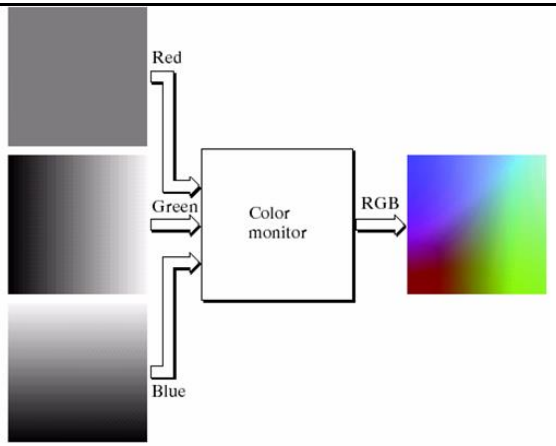


FIGURE 6.9 Generating the RGB image of the cross-sectional color plane (127, *G*, *B*).

Because of the variety of systems in use, it is of considerable interest to have a subset of colors that are likely to be reproduced faithfully, reasonably independently of viewer hardware capabilities. This subset of colors is called the set of *safe RGB colors*, or the set of *all-systems-safe colors*. In Internet applications, they are called *safe Web colors* or *safe browser colors.*

We assume that 256 colors is the minimum number of colors that can be reproduced faithfully by any system. Forty of these 256 colors are known to be

processed differently by various operating system. We have 216 colors that are common to most systems, and are the safe colors, especially in Internet applications. Each of the 216 safe colors has a RGB representation with:

$$R,G,B \in \{0,51,102,153,204,255\}$$

We have $(6)^3=216$ possible color values. It is customary to express these values in the hexagonal number system.

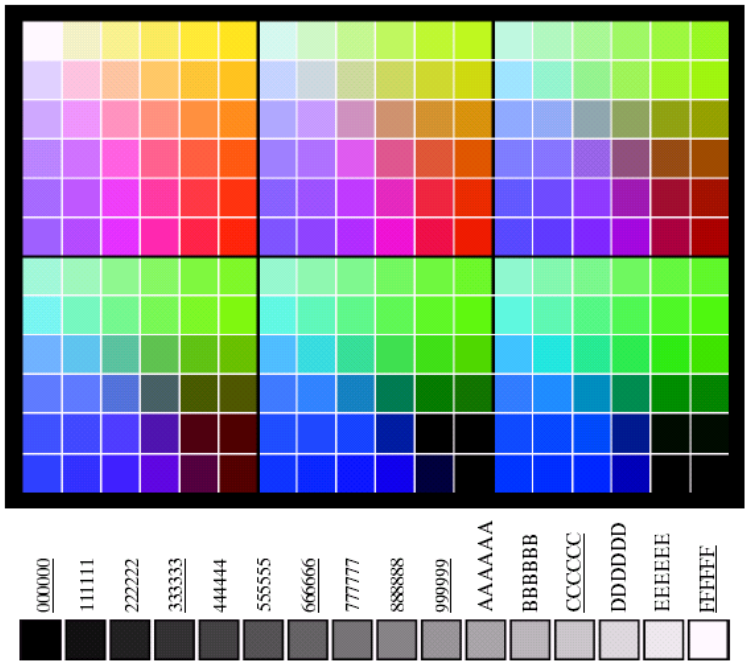
Number System	Color Equivalents					
Hex	00	33	66	99	CC	FF
Decimal	0	51	102	153	204	255

TABLE 6.1 Valid values of each RGB component in a safe color.

Each safe color is formed from three of the two digit hex numbers from the above table. For example purest red if FF0000. The values 000000 and FFFFFF represent black and white respectively.

Figure 6.10(a) shows the 216 safe colors, organized in descending RGB values. Figure 6.10(b) shows the hex codes for all the possible gray colors in the 216 safe color system.

Figure 6.11 shows the RGB safe-color cube.



a
b

FIGURE 6.10
(a) The 216 safe RGB colors. (b) All the grays in the 256-color RGB system (grays that are part of the safe color group are shown underlined).

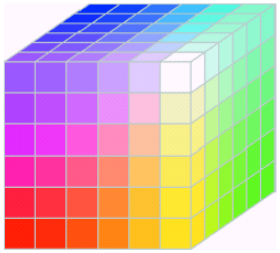


FIGURE 6.11 The RGB safe-color cube.

http://www.techbomb.com/websafe/

The CMY and CMYK Color Models

Cyan, magenta, and yellow are the secondary colors of light but the primary color of pigments. For example, when a surface coated with yellow pigment is illuminated with white light, no blue light is reflected from the surface. Yellow subtracts blue light from reflected white light (which is composed of equal amounts of red, green, and blue light).

Most devices that deposit color pigments on paper, such as color printers and copiers, require CMY data input and perform RGB to CMY conversion. Assuming that the color values were normalized to range ***[0,1]***, this conversion is:

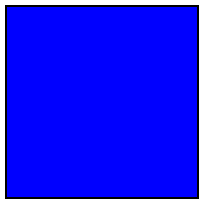
$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

From this equation we can easily deduce, that pure cyan does not reflect red, pure magenta does not reflect green, and pure yellow does not reflect blue.

Equal amount of pigments primary, cyan, magenta, and yellow should produce black. In practice, combining these colors for printing produces a muddy-looking black. In order to produce true black (which is the predominant color in printing), a fourth color, *black*, is added, giving rise to the **CMYK** color model.

Computer Vision

Course 5



Course 5

The HSI Color Model

The RGB, CMY, and other similar color models are not well suited for describing colors in terms that are practical for human interpretation.

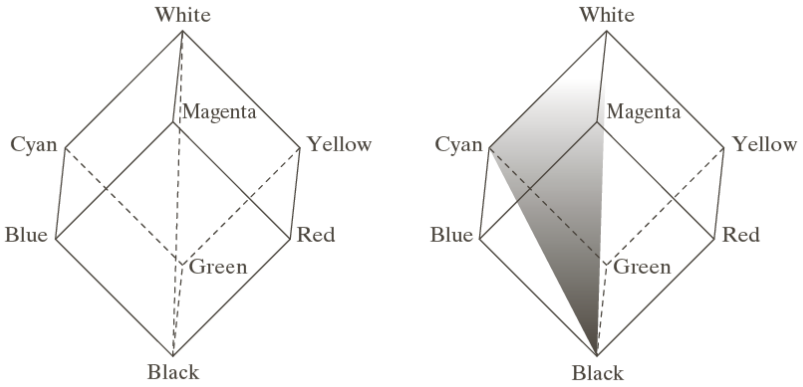
We (humans) describe a color by its hue, saturation and brightness. Hue is a color attribute that describes a pure color, saturation gives a measure of the degree to which a pure color is diluted by white light and brightness is a subjective descriptor that embodies the achromatic notion of intensity.

The HSI (hue, saturation, intensity) color model, decouples the intensity component from the color information (hue and saturation) in a color image.

What is the link between the RGB color model and HSI color model? Consider again the RGB unit cube. The intensity axis is the line joining the black and the white vertices. Consider a color point in the RGB cube. Let ***P*** be a plane

perpendicular to the intensity axis and containing the color point. The intersection of this plane with the intensity axis gives us the intensity of the color point. The saturation (purity) of the considered color point increases as a function of distance from the intensity axis (the saturation of the point on the intensity axis is zero).

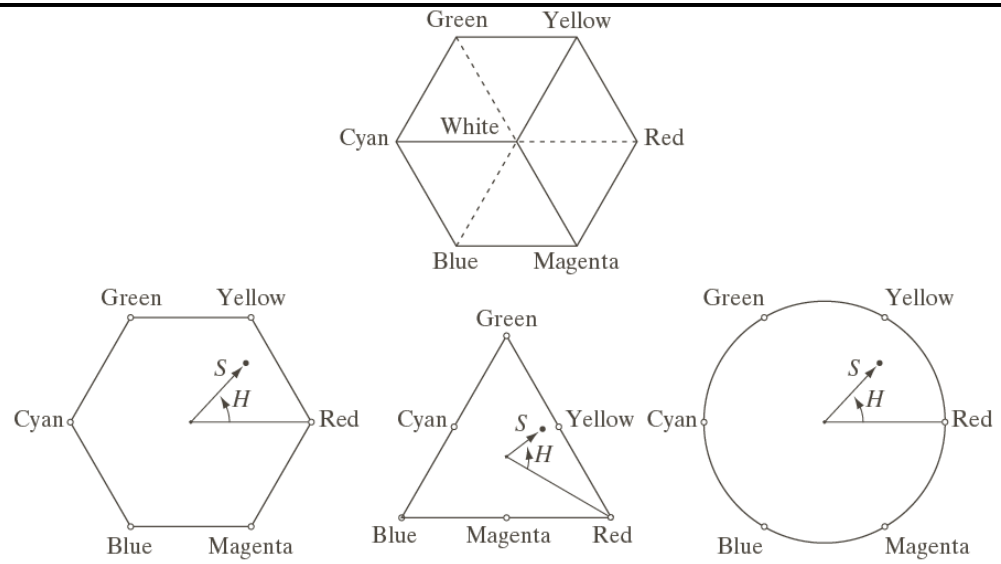
In order to determine how hue can be linked to a given RGB point, consider a plane defined by black, white and cyan. The intensity axis is also included in this plane. The intersection of this plane with the RGB-cube is a triangle. All point contained in this triangle would have the same hue (i.e. cyan).



a b

FIGURE 6.12 Conceptual relationships between the RGB and HSI color models.

The HSI space is represented by a vertical intensity axis and the locus of color points that lie on planes perpendicular to this axis. As the planes move up and down the intensity axis, the boundary defined by the intersection of this plane with the faces of the cube have either triangular or hexagonal shape.



a
b c d

FIGURE 6.13 Hue and saturation in the HSI color model. The dot is an arbitrary color point. The angle from the red axis gives the hue, and the length of the vector is the saturation. The intensity of all colors in any of these planes is given by the position of the plane on the vertical intensity axis.

In the plane shown in Figure 6.13(a) primary colors are separated by 120º. The secondary colors are 60º from the primaries. The hue of the point is determined

by an angle from some reference point. Usually (but not always) an angle of 0º from the red axis designates 0 hue, and the hue increases counterclockwise from there. The saturation (distance from the vertical axis) is the length of the vector from the origin to the point. The origin is defined by the intersection of the color plane with the vertical intensity axis.

Converting colors from RGB to HSI

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

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

$$S = 1 - \frac{3}{R+G+B}\min\{R,G,B\}$$

$$I = \frac{1}{3}\left(R+G+B\right)$$

It is assumed that the RGB values have been normalized to the range ***[0,1]*** and that angle θ is measured with respect to the red axis of the HSI space in Figure 6.13. Hue can be normalized to the range ***[0,1]*** by dividing it to 360º. The other two HSI components are in this range if the RGB values are in the interval ***[0,1].***

R=100, G=150, B=200 → H=210º, S=1/3, I=150/255=0.588

Converting colors from HSI to RGB

Given values of HSI we now want to find the corresponding RGB values in the same range.

RG sector ($0^o \leq H < 120^o$)

$$B = I(1 - S)$$

$$R = I\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right]$$

$$G = 3I - (R + B)$$

GB sector ($120^o \le H < 240^o$)

$$R = I(1-S)$$

$$H = H - 120^\circ,\ G = I\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right]$$

$$B = 3I - (R+G)$$

BR sector ($120^o \le H < 240^o$)

$$G = I(1-S)$$

$$H = H - 240^\circ,\ B = I\left[1 + \frac{S\cos H}{\cos(60^\circ - H)}\right]$$

$$R = 3I - (G+B)$$

Lab color space

The effectiveness of such transformations is judged ultimately in print. The transformations are developed and evaluated on monitors. It is necessary to have a high degree of consistency between the monitors and the output devices. This is best accomplished with a *device-independent color model* that relates the color gamut of the monitors and output devices, as well as any other devices being used, to one another. The model of choice for many *color management systems* (CMS) is the CIE L*a*b* model, also called CIELAB. The L*a*b* color components are given by the following equations:

$$L^* = 116 \cdot h\left(\frac{Y}{Y_W}\right) - 16$$

$$a^* = 500 \cdot \left[h\left(\frac{X}{X_W}\right) - h\left(\frac{Y}{Y_W}\right)\right]$$

$$b^* = 200 \cdot \left[h\left(\frac{Y}{Y_W}\right) - h\left(\frac{Z}{Z_W}\right)\right]$$

$$h(q) = \begin{cases} \sqrt[3]{q} & q > 0.008856 \\ 7.787q + \dfrac{16}{116} & q \leq 0.008856 \end{cases}$$

X_W, Y_W, Z_W are reference tristimulus values – typically the white of a perfectly reflecting diffuser under CIE standard D65 illumination ($x = 0.3127$, $y = 0.33290$, $z = 1 - x - y$).

The *L*a*b** color space is *colorimetric* (i.e. colors perceived as matching are encoded identically), *perceptually uniform* (i.e. color differences among various hues are perceived uniformly), and *device independent*. Like the HSI system, the *L*a*b** system is an excellent decoupler of intensity (represented by lightness ***L****) and color (represented by ***a**** for red minus green and ***b**** for green minus blue), making it useful in both image manipulation (tone and contrast editing) and image compression applications.

Course 5

Other color spaces

YIQ – for the NTSC (National Television System Committee) television system in US
Y – luminance
I (in-phase), Q (quadrature) – chrominance

YUV – for the PAL (Phase Alternation Line) and SECAM (Séquentiel Couleur à Mémoire) television system in Europe
(I, Q) – obtained by rotating (U,V)

YC_bC_r – digital video transmission

More about color spaces in: Andreas Koschan, Mongi Abidi, *Digital Color Image Processing*, Wiley, 2008

Color Difference

RGB, CMY, Lab – Euclidean distance

HSI - $F_1=(H_1, S_1, I_1)$, $F_2=(H_2, S_2, I_2)$

$$d_{\text{HSI}}(F_1,F_2)=\sqrt{(\Delta I)^2+(\Delta C)^2}\ ,\ \Delta I=\left|I_1-I_2\right|$$

$$\Delta C=\sqrt{S_1^2+S_2^2-2S_1S_2\cos\theta}$$

$$\theta=\begin{cases}\left|H_1-H_2\right| & \text{if } \left|H_1-H_2\right|\le\pi \\ 2\pi-\left|H_1-H_2\right| & \text{if } \left|H_1-H_2\right|>\pi\end{cases}$$

Course 5

Pseudo-color Image Processing

Pseudo-color (also called *false color*) image processing consists of assigning colors to gray values based on a specified criterion. The main use of pseudo-color is for human visualization and interpretation of gray-scale events in an image or sequence of images.

Intensity (Density) Slicing

If an image is viewed as a 3-D function, the method can be described as one of placing planes parallel to the coordinate plane of the image; each plane then "slices" the function in the area of intersection.

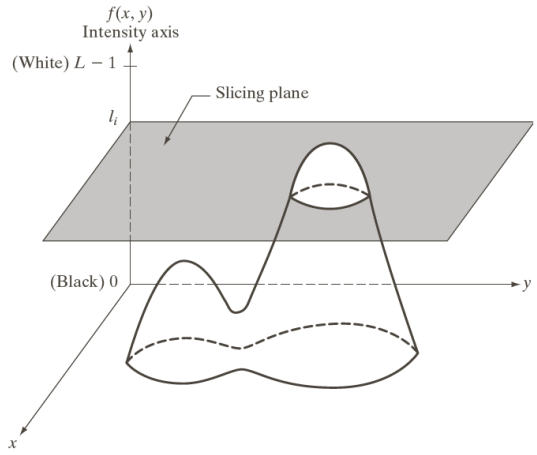


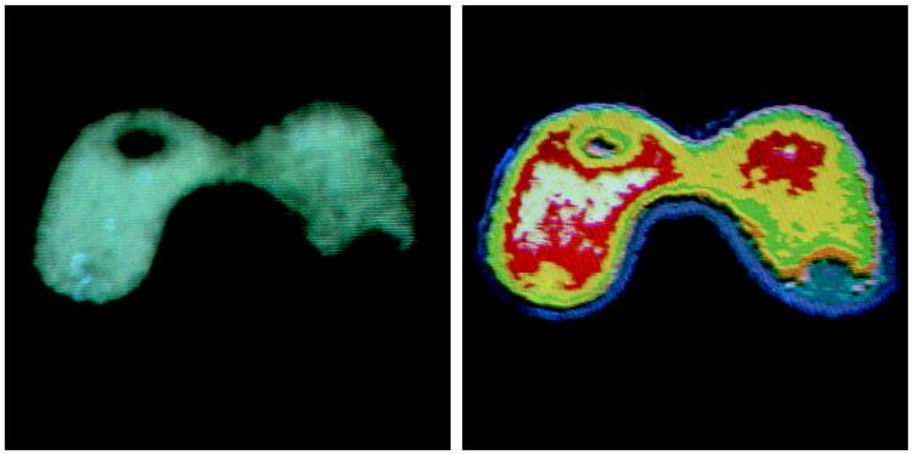
FIGURE 6.18 Geometric interpretation of the intensity-slicing technique.

The plane at $f(x,y)=l_i$ slices the image function into two levels. If a different color is assigned to each side of the plane, any pixel whose intensity level is above the plane will be coded with one color and any pixel below the plane will be coded with other color. Levels that lie on the plane itself may be arbitrarily assigned one of the two colors. The result is a two color image whose

relative appearance can be controlled by moving the slicing plane up and down the intensity axis.

Let ***[0, L-1]*** represent the gray scale, let l_0 represent black (***f(x,y)=0***) and level l_{L-1} represent white (***f(x,y)=L-1***). Suppose that ***P*** planes perpendicular to the intensity axis are defined at levels $l_1, l_2, \ldots, l_P$, $0<P<L-1$. The ***P*** planes partition the gray scale into ***P+1*** intervals, $V_1, V_2, \ldots, V_{P+1}$. Intensity to color assignments are made according to the relation:

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

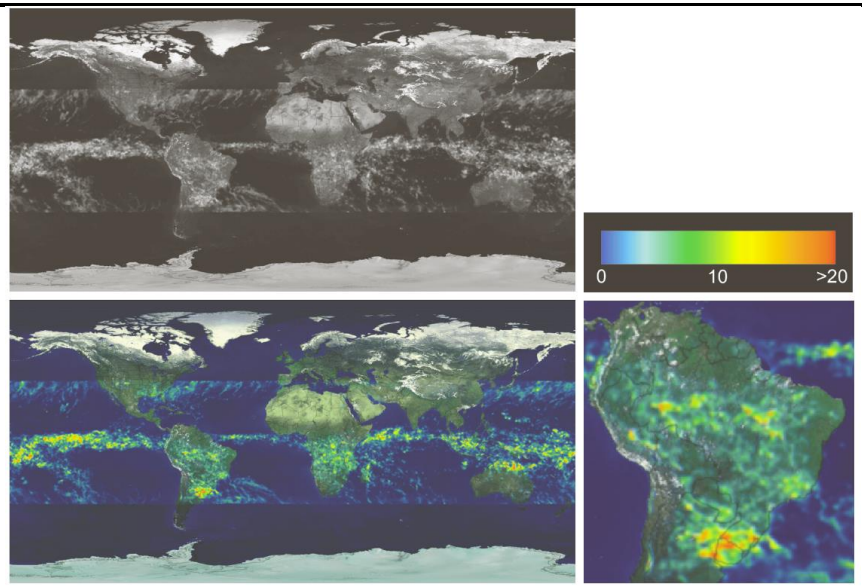


a b

FIGURE 6.20 (a) Monochrome image of the Picker Thyroid Phantom. (b) Result of density slicing into eight colors. (Courtesy of Dr. J. L. Blankenship, Instrumentation and Controls Division, Oak Ridge National Laboratory.)

Measurements of rainfall levels with ground-based sensors are difficult and expensive, and total rainfall figures are even more difficult to obtain because a significant portion of precipitations occurs over the ocean. One way to obtain these figures is to use a satellite. The TRMM (Tropical Rainfall Measuring

Mission) satellite utilizes, among others, three sensors specially designed to detect rain: a precipitation radar, a microwave imager, and a visible and infrared scanner. The results from the various rain sensors are processed, resulting in estimates of average rainfall over a given time period in the area monitored by the sensors. From these estimates, it is not difficult to generate gray-scale images whose intensity values correspond directly to rainfall, with each pixel representing a physical land area whose size depends on the resolution of the sensors.



a b
c d

FIGURE 6.22 (a) Gray-scale image in which intensity (in the lighter horizontal band shown) corresponds to average monthly rainfall. (b) Colors assigned to intensity values. (c) Color-coded image. (d) Zoom of the South American region. (Courtesy of NASA.)

Course 5

Color Slicing

Highlighting a specific range of colors in an image is useful for separating objects from their surroundings. The basic idea is either to:

- display the colors of interest so they stand out from the background
- use the region defined by the colors as a mask for further processing.

One of the simplest ways to "slice" a color image is to map the colors outside some range of interest to a neutral color. If the colors of interest are enclosed by a cube (or hypercube, if $n>3$) of width W and centered at a prototypical (e.g. average) color with components $(a_1, a_2, ..., a_n)$ the set of transformations is:

$$s_i = \begin{cases} 0.5 & \text{if } \left| r_j - a_j \right| > \dfrac{W}{2} \ , \ 1 \le j \le n \\ r_i & \text{otherwise} \end{cases} \qquad , i = 1, 2, ..., n$$

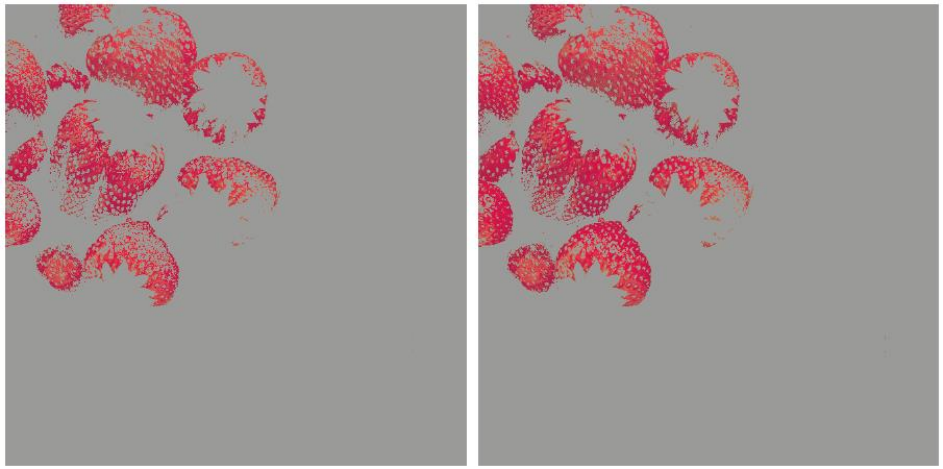
These transformations highlight the colors around the prototype by forcing all other colors to the midpoint of the reference color space (an arbitrarily chosen neutral point).

For the RGB color space, for example, a suitable neutral point is middle gray or color (***0.5, 0.5, 0.5***).

If a sphere is used to specify the colors of interest, the transformations are:

$$s_i = \begin{cases} 0.5 & \text{if } \sum_{j=1}^{n} (r_j - a_j)^2 > R_0^2 \\ r_i & \text{otherwise} \end{cases} \quad , i = 1, 2, ..., n$$

where R_0 is the radius of the enclosing sphere and $(a_1, a_2, ..., a_n)$ are the components of its center.



a b

FIGURE 6.34 Color-slicing transformations that detect (a) reds within an RGB cube of width $W = 0.2549$ centered at (0.6863, 0.1608, 0.1922), and (b) reds within an RGB sphere of radius 0.1765 centered at the same point. Pixels outside the cube and sphere were replaced by color (0.5, 0.5, 0.5).

Image Segmentation

Segmentation subdivides an image into its constituent regions and objects. The level of detail to which the subdivision is carried depends on the problem being solved. Segmentation should stop when the objects or regions of interest in an application have been detected. For example, in the automated inspection of electronic assemblies, interest lies in analysing images of products with the objective of determining the presence or absence of specific anomalies,

such as missing components or broken connection paths. There is no point in carrying segmentation past the level of detail required to identify those elements.

Segmentation of nontrivial images is one of the most difficult tasks in image processing. Segmentation accuracy determines the eventual success or failure of computerized analysis procedure.

https://darwin-public.s3.eu-west-1.amazonaws.com/splash_page/v7-vs-other-tools.mp4

Image segmentation tasks:

(https://www.v7labs.com/blog/image-segmentation-guide#h4)

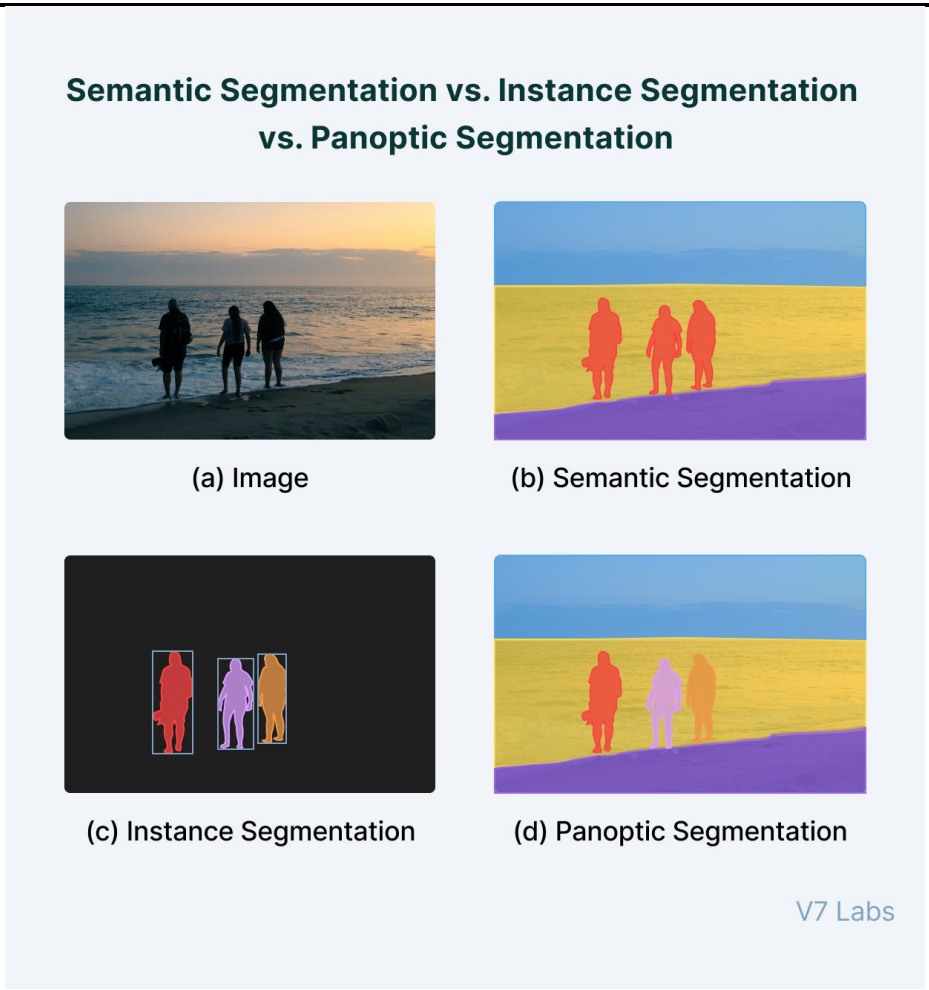
1. **Semantic segmentation** segments out a broad boundary of objects belonging to a particular class. It classifies pixels in an image into semantic classes. Pixels belonging to a particular class are simply classified to that class with no other information or context taken into consideration.

2. **Instance segmentation** provides a segment map for each object it views in the image, without any idea of the class the object belongs to. Pixels are classified into categories on the basis

of “instances” rather than classes. Overlapping or very similar object regions are distinguished on the basis of their boundaries.

3. **Panoptic segmentation** is the conjugation of instance and semantic segmentation tasks. Panoptic segmentation gives us the segment maps of all the objects of any particular class present in the image.

Course 5


Semantic Segmentation vs. Instance Segmentation vs. Panoptic Segmentation
(a) Image
(b) Semantic Segmentation
(c) Instance Segmentation
(d) Panoptic Segmentation
V7 Labs

Course 5

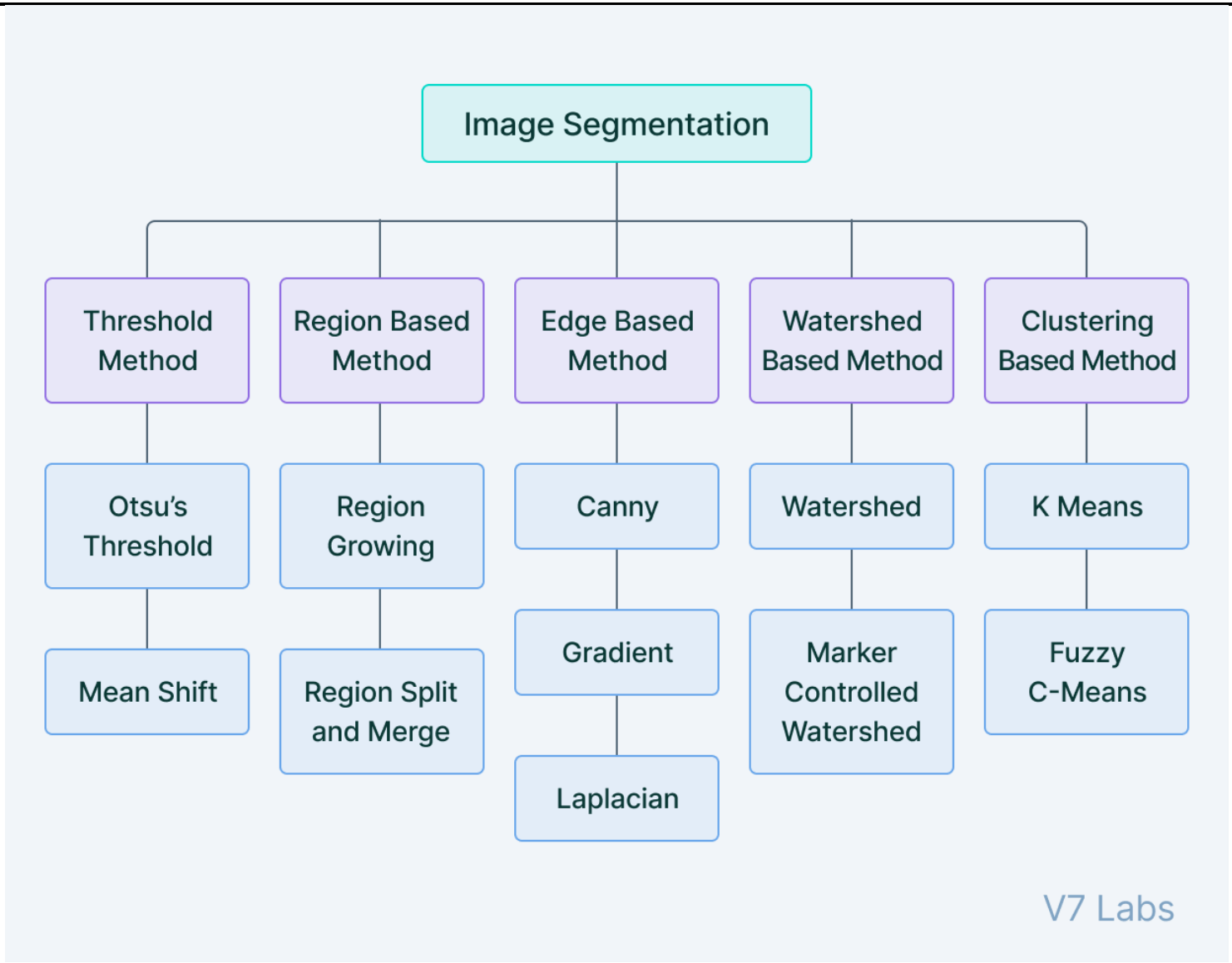


Image Segmentation
Threshold Method
Region Based Method
Edge Based Method
Watershed Based Method
Clustering Based Method
Otsu's Threshold
Region Growing
Canny
Watershed
K Means
Mean Shift
Region Split and Merge
Gradient
Marker Controlled Watershed
Fuzzy C-Means
Laplacian
V7 Labs

Fundamentals

Let $\boldsymbol{R}$ represent the entire spatial region occupied by an image. Image segmentation can be viewed as a process that partitions $\boldsymbol{R}$ into $\boldsymbol{n}$ subregions $\boldsymbol{R_1, R_2, \ldots, R_n}$ such that:

(a) $\bigcup_{i=1}^{n} R_i = R$

(b) R_i is a connected set, $i = 1,2,\ldots,n$

(c) $R_i \cap R_j = \varnothing$ for all i and j, $i \neq j$

(d) $Q(R_i) = TRUE$ for $i = 1,2,\ldots,n$

(e) $Q(R_i \cup R_j) = FALSE$ for any adjacent regions R_i , R_j

$Q(R)$ is a logical predicate defined over the points in set R.

Condition (a) indicates that the segmentation must be complete; that every pixel must be in a region. Condition (b) requires that points in a region be connected in some predefined sense (e.g. the points must be ***4***- or ***8***-connected). Condition (c) indicates that the regions must be disjoint. Condition (d) deals with the properties that must be satisfied by the pixels in a segmented region (for example $Q(R_i)$ is true

if all pixels have the same intensity level). Finally, condition (e) indicates that two adjacent regions $\boldsymbol{R_i}$ and $\boldsymbol{R_j}$ must be different in the sense of predicate $\boldsymbol{Q}$.

The fundamental problem in segmentation is to partition an image into regions that satisfy the preceding conditions. Segmentation algorithms for monochrome images generally are based on one of two basic categories dealing with properties of intensity values: discontinuity and similarity. In the first category, the assumption is that boundaries of regions

are sufficiently different from each other and from the background to allow boundary detection based on local discontinuities in intensity.

Edge-based segmentation is the principal approach used in this category. *Region-based segmentation* approaches from the second category are based on partitioning an image into regions that are similar according to a set of predefined criteria.

Course 5

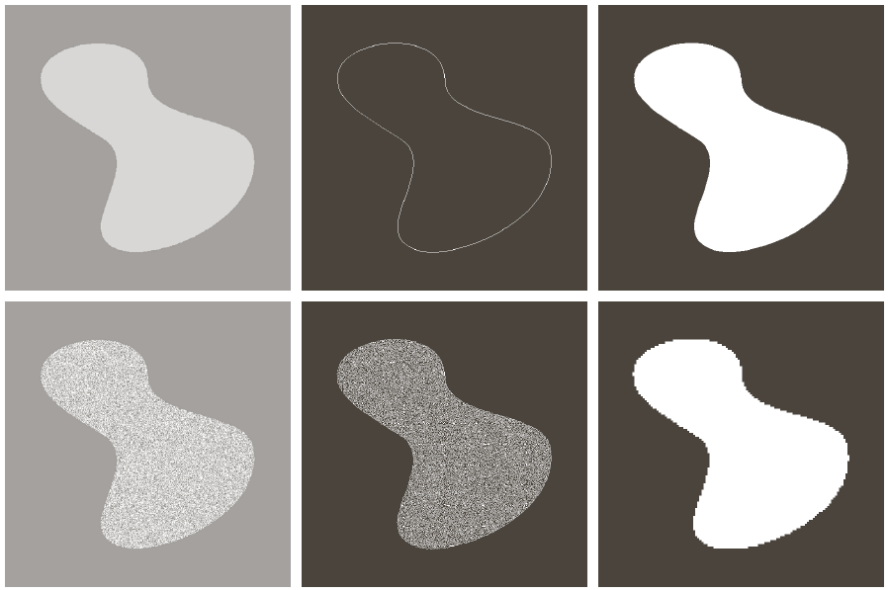


FIGURE 10.1 (a) Image containing a region of constant intensity. (b) Image showing the boundary of the inner region, obtained from intensity discontinuities. (c) Result of segmenting the image into two regions. (d) Image containing a textured region. (e) Result of edge computations. Note the large number of small edges that are connected to the original boundary, making it difficult to find a unique boundary using only edge information. (f) Result of segmentation based on region properties.

Point, Line, and Edge Detection

Edge pixels are pixels at which the intensity of an image function changes abruptly, and *edges* (or *edge segments)* are sets of connected edge pixels. *Edge detectors* are local image processing methods designed to detect edge pixels. A line may be viewed as an edge segment in which the intensity of the background on either side of the line is either much higher or much lower than the intensity of the line pixels.

Background

Abrupt, local changes in intensity can be detected using derivatives, usually first- and second-order derivatives which are defined in terms of differences.

Any approximation for a first derivative must be:

(1) zero in areas of constant intensity

(2) non-zero at the onset of an intensity step or ramp

(3) non-zero at points along an intensity ramp.

An approximation for a second derivative must be:

(1) zero in areas of constant intensity

(2) nonzero at the onset and end of an intensity step or ramp

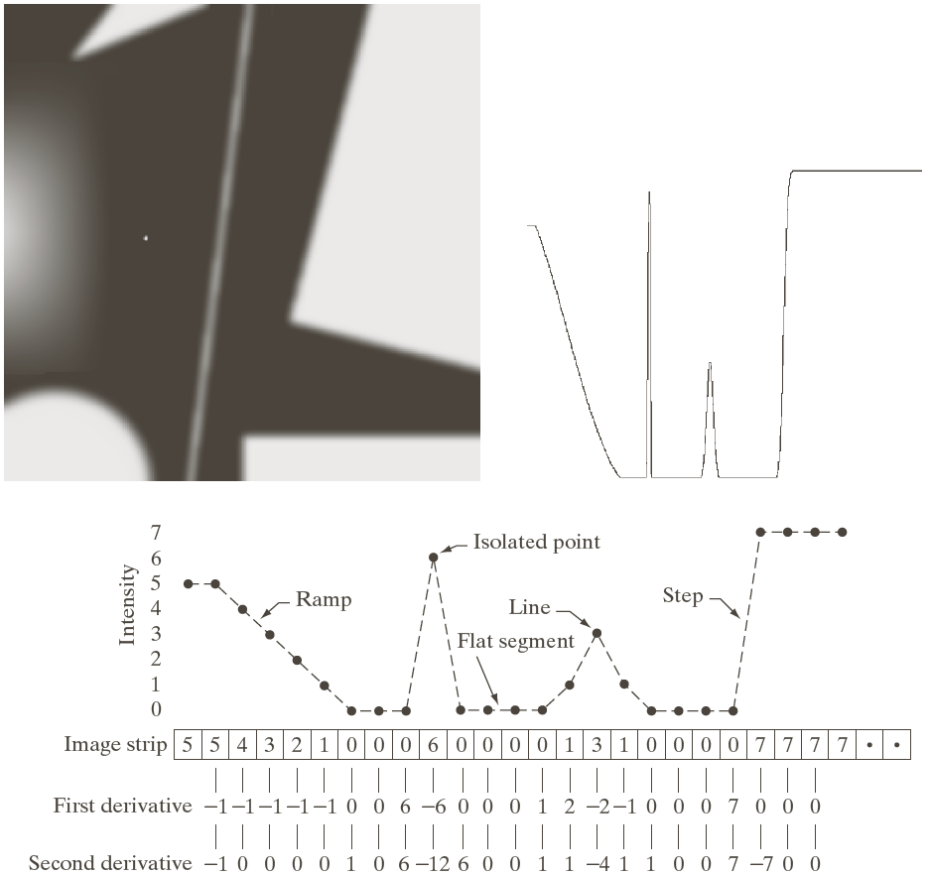
(3) zero along intensity ramps.

$$\frac{\partial f}{\partial x} = f'(x) \approx f(x+1) - f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial f'(x)}{\partial x} \approx f'(x+1) - f'(x) \approx f(x+2) - 2f(x+1) + f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = f''(x) \approx f(x+1) + f(x-1) - 2f(x)$$

Course 5



a b
c

FIGURE 10.2 (a) Image. (b) Horizontal intensity profile through the center of the image, including the isolated noise point. (c) Simplified profile (the points are joined by dashes for clarity). The image strip corresponds to the intensity profile, and the numbers in the boxes are the intensity values of the dots shown in the profile. The derivatives were obtained using Eqs. (10.2-1) and (10.2-2).

(a) First-order derivatives generally produce thicker edges in an image

(b) Second-order derivatives have a stronger response to fine detail, such as thin lines, isolated points, and noise

(c) Second-order derivatives produce a double-edge response at ramp and step transitions in intensity

(d) The sign of the second-order derivative can be used to determine whether a transition into an edge is from light to dark or dark to light.

Computing first and second derivatives at every pixel location is done using spatial filters.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

FIGURE 10.3
A general 3×3 spatial filter mask.

The *response* of the mask at the center point of the region is:

$$\boldsymbol{R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum_{k=1}^{9} w_k z_k} \qquad (1)$$

where $\boldsymbol{z_k}$ is the intensity of the pixel whose spatial location corresponds to the location of the $\boldsymbol{k}$-th coefficient in the mask.

Detection of isolated points

Point detection is based on computation of the second derivative of the image.

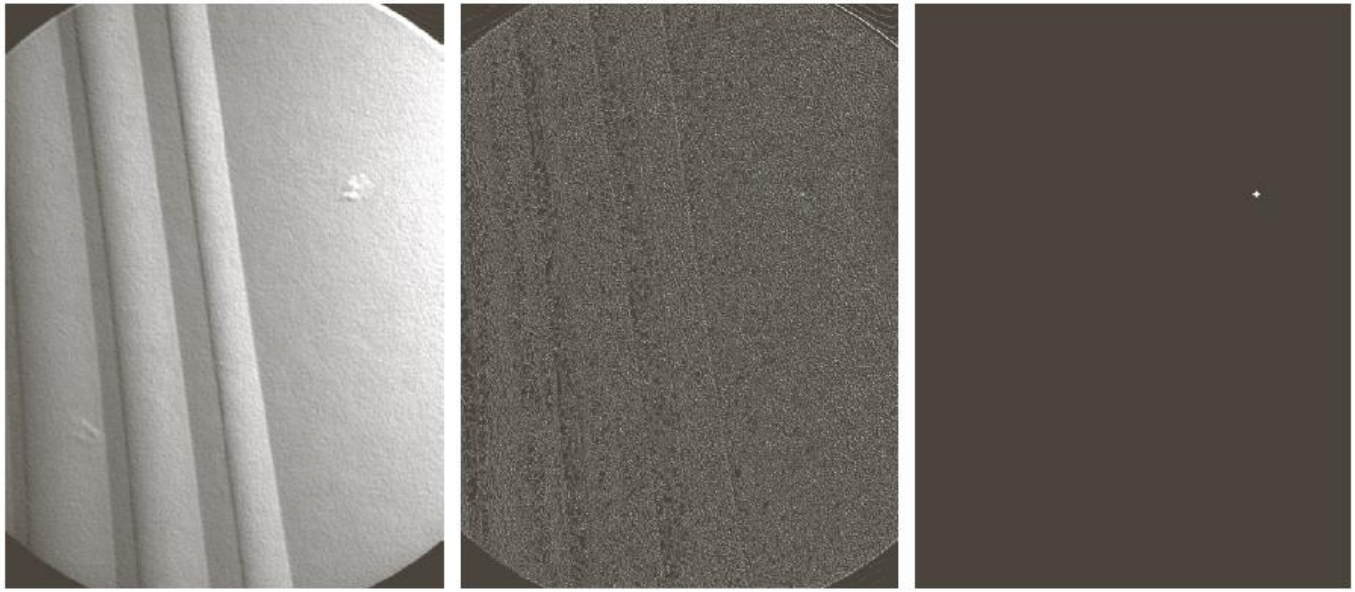
$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$ the Laplacian

$$\frac{\partial^2 f}{\partial x^2} \approx f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f}{\partial y^2} \approx f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f(x,y) \approx f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

1	1	1
1	−8	1
1	1	1



a
b c d

FIGURE 10.4
(a) Point detection (Laplacian) mask. (b) X-ray image of turbine blade with a porosity. The porosity contains a single black pixel. (c) Result of convolving the mask with the image. (d) Result of using Eq. (10.2-8) showing a single point (the point was enlarged to make it easier to see). (Original image courtesy of X-TEK Systems, Ltd.)

Using the Laplacian mask in Figure 10.4(a) we say that a point has been detected at the location ***(x, y)*** on which the mask is centred if the absolute value of the response of the mask at that point exceeds a specified threshold. Such points are labelled ***1*** in the output image and all others are labelled ***0***, thus producing a binary image. The output is obtained using the following expression:

$$g(x,y)=\begin{cases}1 & \text{if} \quad |R(x,y)| \geq T \\ 0 & \text{otherwise}\end{cases}$$

where g is the output image, $T > 0$ is the threshold, and R is given by (1). This formulation measures the weighted difference between a pixel and its 8-neighbors. The idea is that the intensity of an isolated point will be quite different from its surroundings and thus will be easily detectable by this type of mask. The only differences in intensity that are considered of interest are those large enough (as determined

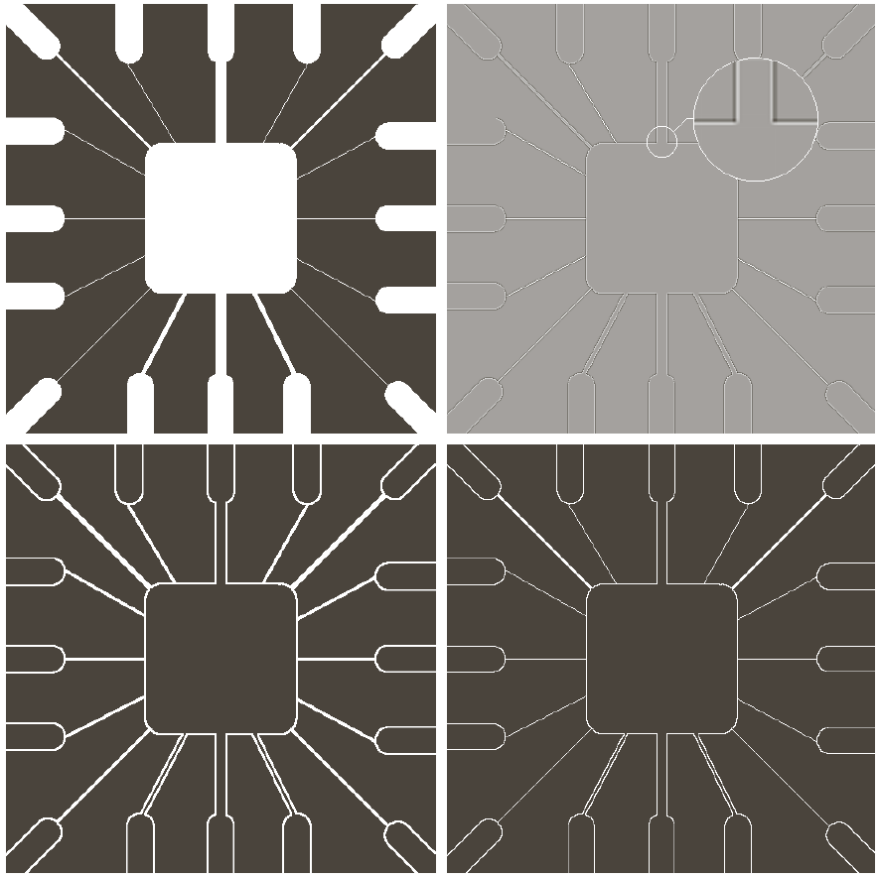
by ***T)*** to be considered isolated points. The sum of the coefficients of the mask is zero, indicating that the mask response will be zero in areas of constant intensity.

Line Detection

For line detection, we can expect second derivatives to result in a stronger response and to produce thinner lines than first derivatives. We can use the Laplacian mask in Figure 10.4(a) for line detection also, taking care of the double-line effect of the second order derivative.

Figure 10.5(a) shows a ***486 × 486*** (binary) portion of a wire-bond mask for an electronic circuit and Figure 10.5(b) shows its Laplacian. Scaling is necessary in this case (the Laplacian image contains negative values). Mid grey represents ***0***, darker shades of grey represent negative values, and lighter shades are positive. It might appear that negative values can be handled simply by taking the absolute value of the Laplacian image. Figure 10.5(c) shows that this approach doubles the thickness of the lines. A more suitable approach

is to use only the positive values of the Laplacian (Figure 10.5(d)).



a b
c d

FIGURE 10.5
(a) Original image. (b) Laplacian image; the magnified section shows the positive/negative double-line effect characteristic of the Laplacian. (c) Absolute value of the Laplacian. (d) Positive values of the Laplacian.

The Laplacian detector in Figure 10.4(a) is isotropic, so its response is independent of the direction (with respect to the four directions of the ***3 × 3*** Laplacian mask: vertical, horizontal, and two diagonals). Often, interest lies in detecting lines in *specified* directions.

Consider the masks in Figure 10.6. Suppose that an image with a constant background and containing various lines (oriented at ***0º, ±45º*** and ***90º***) is filtered with the first mask. The maximum responses would occur at image locations in

which horizontal lines passed through the middle row of the mask.

−1	−1	−1	2	−1	−1	−1	2	−1	−1	−1	2
2	2	2	−1	2	−1	−1	2	−1	−1	2	−1
−1	−1	−1	−1	−1	2	−1	2	−1	2	−1	−1
Horizontal			+45°			Vertical			−45°		

FIGURE 10.6 Line detection masks. Angles are with respect to the axis system in Fig. 2.18(b).

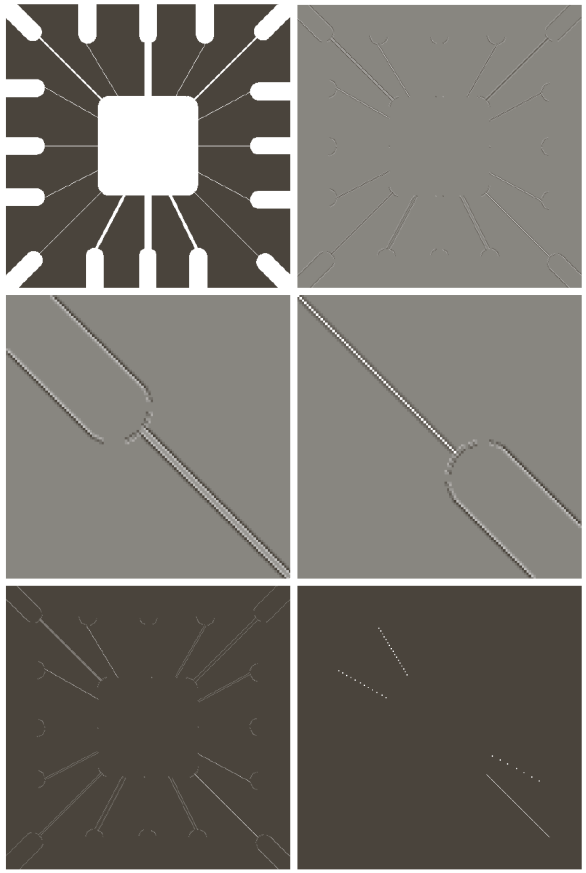
A similar experiment would reveal that the second mask in Figure 10.6 responds best to lines oriented $+45^o$; the third mask to vertical lines; and the fourth mask to lines in the -45^o direction.

Let R_1, R_2, R_3 and R_4 denote the response of the masks in Figure 10.6 from left to right, where the Rs are given by (1). Suppose that an image is filtered (individually) with the four masks. If at a given point in the image $|R_k| > |R_j|$, for all $j \neq k$,

that point is said to be more likely associated with a line in the direction of mask ***k***.

If we are interested in detecting all the lines in an image in the direction defined by a given mask, we simply run the mask through the image and threshold the absolute value of the result. The points that are left are the strongest responses which, for line ***1*** pixels thick, correspond closest to the direction defined by the mask.

In Figure 10.7(a) image, we are interested in lines oriented at ***+45º***. We use the second mask, the result is in Figure 10.7(b).



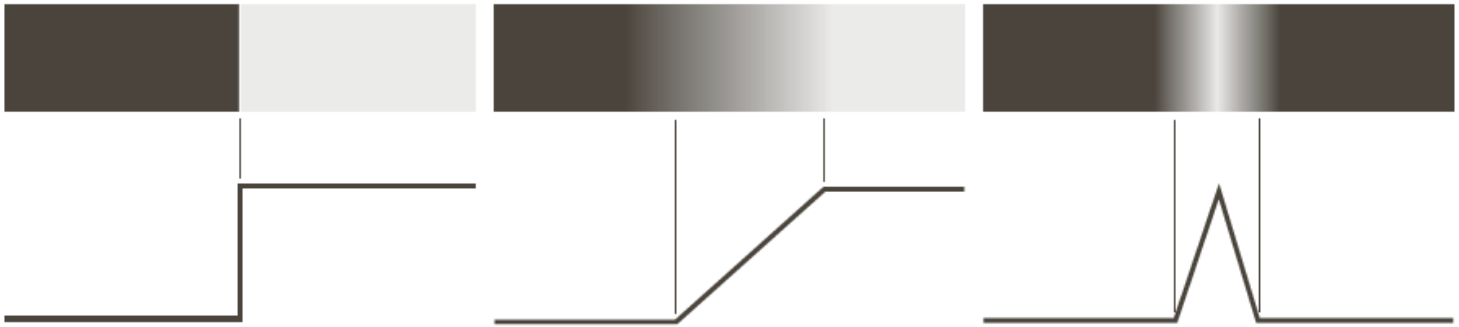
a b
c d
e f

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 Models

Edge detection is the approach used most frequently for segmenting images based on abrupt (local) changes in intensity.

Edge models are classified according to their intensity profiles. A *step edge* involves a transition between two intensity levels occurring ideally over the distance of ***1*** pixel. Figure 10.8(a) shows a section of a vertical step edge and a horizontal intensity profile through the edge.



a b c

FIGURE 10.8
From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

In practice, digital images have edges that are blurred and noisy, with the degree of blurring determined principally by limitations in the focusing mechanism, and the noise level determined principally by the electronic components of the imaging system. In such situations, edges are more closely

modelled as having an intensity *ramp* profile, such as the edge in Figure 10.8(b). The slope of the ramp is inversely proportional to the degree of blurring in the edge. In this model, we no longer have a thin (***1*** pixel thick) path. An edge point now is any point contained in the ramp and an edge segment would then be a set of such points that are connected.

A third model of an edge is the so-called *roof edge*, having the characteristics illustrated in Figure 10.8(c). Roof edges

are models of lines through a region, with the base (width) of a roof edge being determined by the thickness and sharpness of the line.

It is not unusual to find images that contain all three types of edges.

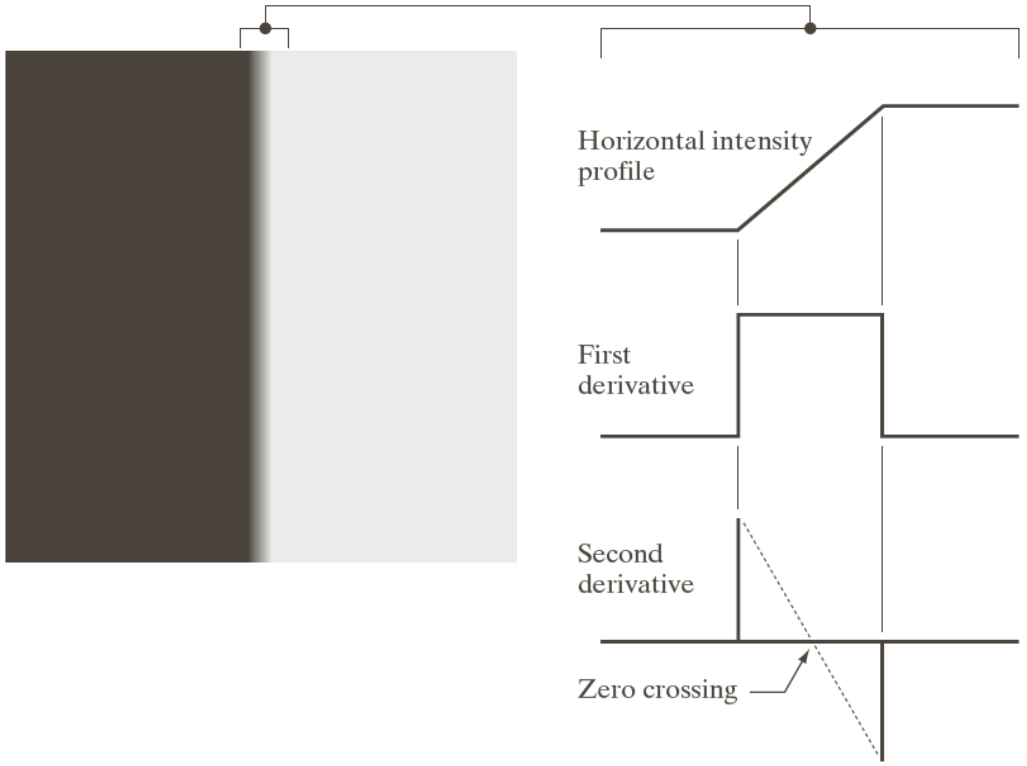
The *magnitude* of the first derivative can be used to detect the presence of an edge at a point in an image. Similarly, the *sign* of the second derivative can be used to determine whether an

edge pixel lies on the dark or light side of an edge. The second derivative has the following properties:

(1) it produces two values for every edge in an image (an undesirable feature)

(2) its zero crossing can be used for locating the centres of thick edges

The *zero crossing* of the second derivative is the intersection between the zero-intensity axis and a line extending between the extrema of the second derivative.



a b

FIGURE 10.10
(a) Two regions of constant intensity separated by an ideal vertical ramp edge. (b) Detail near the edge, showing a horizontal intensity profile, together with its first and second derivatives.

There are three fundamental steps performed in edge detection:

1. *Image smoothing for noise reduction*
2. *Detection of edge points* – this is a local operation that extracts from an image all points that are potential candidates to become edge points
3. *Edge localization* – the objective of this step is to select from the candidate points only the points that are true members of the set of points comprising an edge.

Basic Edge Detection

The image gradient and its properties

The gradient of an image is the tool for finding edge strength and direction at location ***(x, y)***:

$$\nabla f \equiv \mathbf{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

This vector has the important geometrical property that it points in the direction of the greatest rate of change for f at location (x,y).

The *magnitude (length)* of vector ∇f

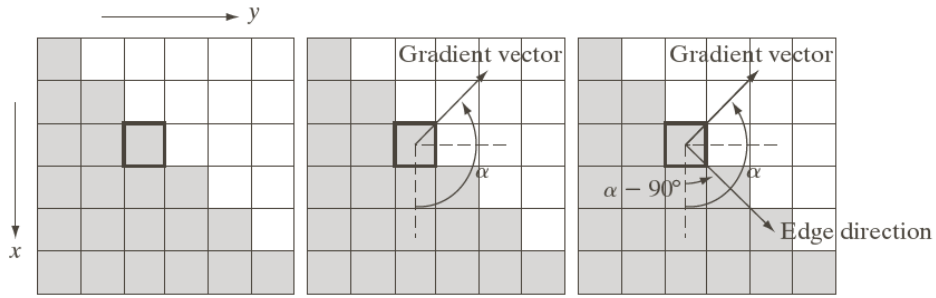
$$M(x,y) = \mathbf{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

is the value of the rate of change in the direction of the gradient vector.

The *direction* of the gradient vector is given by the angle:

$$\alpha(x, y) = \arctan\left[\frac{g_x}{g_y}\right]$$

measured with respect to the ***x***-axis. The direction of an edge at any arbitrary point ***(x, y)*** is *orthogonal* to the direction, $\alpha(x, y)$, of the gradient vector at the point.

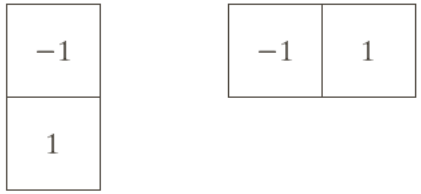


The gradient vector sometimes is called the *edge normal*. When the vector is normalized to unit length (by dividing it by its magnitude) the resulting vector is commonly referred to as the *edge unit normal*.

Gradient operators

$$g_x = \frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$

$$g_y = \frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y)$$



a b

FIGURE 10.13 One-dimensional masks used to implement Eqs. (10.2-12) and (10.2-13).

When diagonal edge direction is of interest, we need a 2-D mask. The *Roberts cross-gradient operators* are one of the earliest attempts to use 2-D masks with a diagonal preference. Consider the ***3×3*** region in Figure 10.14(a). The Roberts operators are based on implementing the diagonal differences.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0	0	−1
0	1	1	0

Roberts

−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

−1	−2	−1	−1	0	1
0	0	0	−2	0	2
1	2	1	−1	0	1

Sobel

a
b c
d e
f g

FIGURE 10.14
A 3 × 3 region of an image (the z's are intensity values) and various masks used to compute the gradient at the point labeled z_5.

$$g_x = \frac{\partial f}{\partial x} = z_9 - z_5 = f(x+1, y+1) - f(x, y)$$

$$g_y = \frac{\partial f}{\partial y} = z_8 - z_6 = f(x+1, y) - f(x, y+1)$$

Masks of size 2×2 are simple conceptually, but they are not as useful for computing edge direction as masks that are symmetric about the centre point, the smallest of which are of size 3×3.

Prewitt operators

$$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

Sobel operators

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

The Sobel masks have better noise-suppression (smoothing) effects than the Prewitt masks.

0	1	1	−1	−1	0
−1	0	1	−1	0	1
−1	−1	0	0	1	1

Prewitt

0	1	2	−2	−1	0
−1	0	1	−1	0	1
−2	−1	0	0	1	2

Sobel

a b
c d

FIGURE 10.15 Prewitt and Sobel masks for detecting diagonal edges.

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, $|g_x| + |g_y|$.

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.

a b

FIGURE 10.19 Diagonal edge detection. (a) Result of using the mask in Fig. 10.15(c). (b) Result of using the mask in Fig. 10.15(d). The input image in both cases was Fig. 10.18(a).

When interest lies both in highlighting the principal edges and on maintaining as much connectivity as possible, it is common practice to use both smoothing and thresholding.

a b

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.