❖ DIGITAL IMAGE PROCESSING

Ω UNIT – IV

Color Image Processing: Color Models: RGB Color Model – HSL Color Model CMYK Color Model -CIE Color Model. Color Transformation. Image Compression: Lossy Image Compression – Lossless Image Compression. Morphological Image Processing: Introduction Operations: Erosion and Dilation – Opening and Closing – Thinning – Skeletonization.

Color Image Processing

- In automated image analysis, color is a powerful descriptor, which simplifies object identification and extraction.
- The human eye can distinguish between thousands of color shades and intensities but only about 20-30 shades of gray. Hence, use of color in human image processing would be very effective.
- Color image processing consists of two parts: Pseudo-color processing and Full color processing.
- In pseudo-color processing, (false) colors are assigned to a monochrome image. For example, objects with different intensity values maybe assigned different colors, which would enable easy identification/recognition by humans.
- In full-color processing, images are acquired with full color sensors/cameras. This has become common in the last decade or so, due to the easy and cheap availability of color sensors and hardware.

Color Fundamentals

- When a beam of sunlight is passed through a glass prism, the emerging beam of light is not white but consists of a continuous spectrum of colors (Sir Isaac Newton, 1666).

- The color spectrum can be divided into six broad regions: violet, blue, green, yellow, orange, and red.

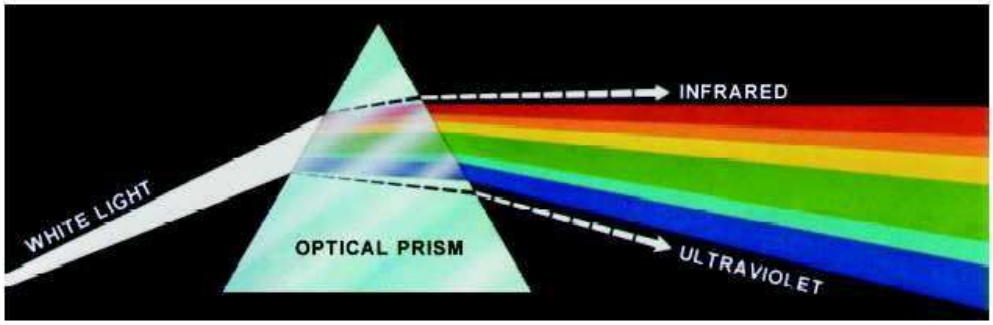


FIGURE 6.1 Color spectrum seen by passing white light through a prism. (Courtesy of the General Electric Co., Lamp Business Division.)

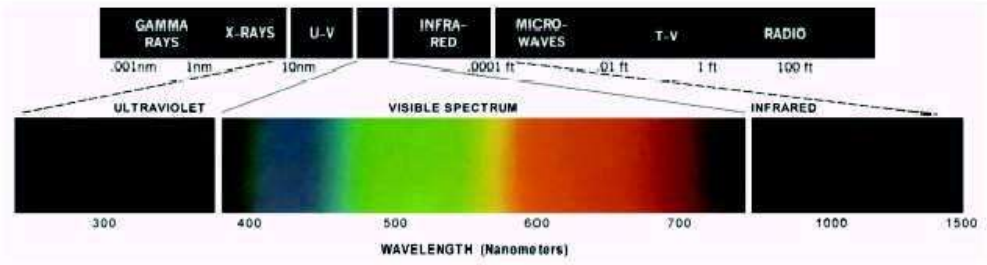


FIGURE 6.2 Wavelengths comprising the visible range of the electromagnetic spectrum. (Courtesy of the General Electric Co., Lamp Business Division.)

- The different colors in the spectrum do not end abruptly but each color blends smoothly into the next.

- Color perceived by the human eye depends on the nature of light reflected by an object. Light that is relatively balanced in all visible wavelengths is perceived as white. Objects that appear green reflect

more light in the 500-570 nm range (absorbing other wavelengths of light).

- Characterization of light is important for the understanding of color.

- If the light is **achromatic** (devoid of color), its only attribute is its **intensity** (amount of light). This is what we have been dealing with so far. The term graylevel refers to the scalar measure of the intensity of light --- black to grays to white.

- **Chromatic** light spans the electromagnetic (EM) spectrum from approximately 400 nm to 700 nm.

- Three basic quantities are used to describe the quality of a chromatic source of light:

- **Radiance** is the total amount of light that flows from a light source (measured in Watts).
- **Luminance** gives a measure of the amount of energy an observer perceives from a light source (measured in lumens).
- **Brightness** is a subjective descriptor that is impossible to measure.

- Cones in the retina are responsible for color perception in the human eye.

- Six to seven million cones in the human eye can be divided into three categories: red light sensitive cones (65%), green light sensitive cones (33%) and blue light sensitive cones (2%). The latter cones are the most sensitive ones.

- Absorption of light by the three types of cones is illustrated in the figure below:

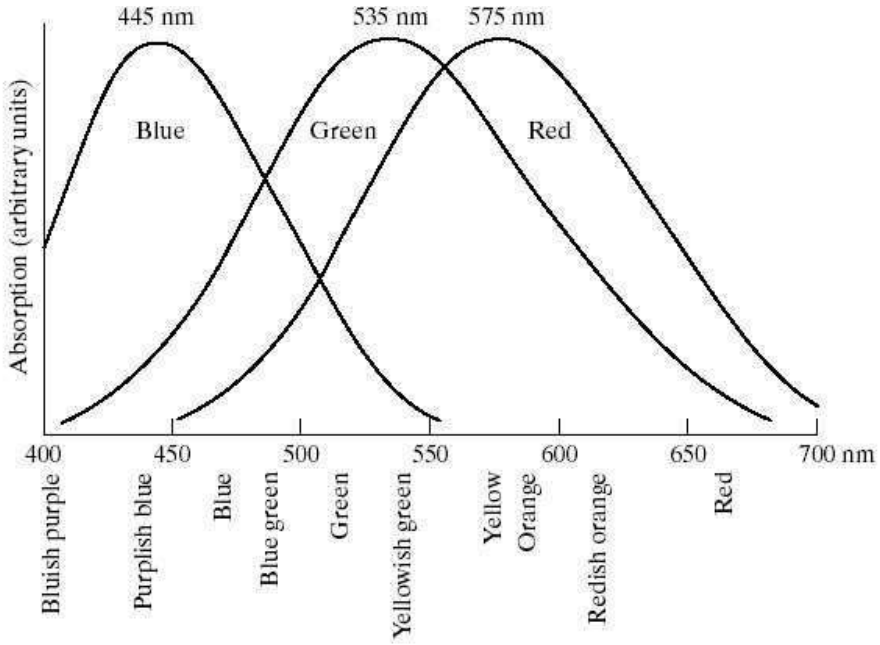
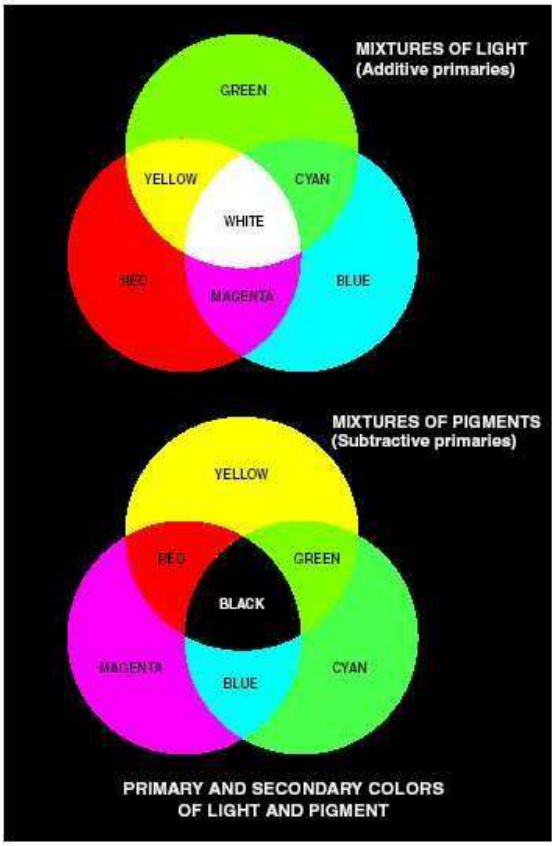


FIGURE 6.3 Absorption of light by the red, green, and blue cones in the human eye as a function of wavelength.

- Due to the absorption characteristics of the human eye, all colors perceived by the human can be considered as a variable combination of the so called three **primary colors**:
  - Red (R) (700 nm)
  - Green (G) (546.1 nm)
  - Blue (B) (435.8 nm)

- The wavelengths for the three primary colors are established by standardization by the CIE (International Commission on Illumination). They correspond to the experimental curve only approximately.

- Note that the specific color wavelengths are used mainly for standardization. It is not possible to produce all colors purely by combining these specific wavelengths.



a
b

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

- Primary colors when added produce **secondary colors**:
  - Magenta (red + blue)
  - Cyan (green + blue)
  - Yellow (red + green)

- Mixing the three primaries, or a secondary with its opposite primary, in the right intensities produces white light.

- A primary color of pigment is defined as one that subtracts or absorbs 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 pigment colors are red, green, and blue.

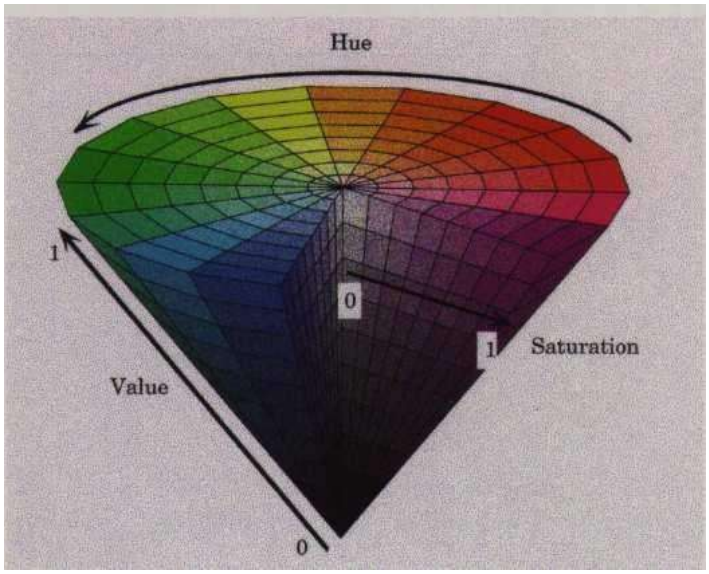
- Mixing the three pigment primaries, or a secondary with its opposite primary, in the right intensities produces black.

- Color television or a computer monitor is an example of additive nature of the color of light. The inside of the screen is coated with dots of phosphor, each being capable of producing one of the three primary colors. A combination of light of the three primary colors produces all the different colors we see.

- Printing is an example of the subtractive nature of color pigments. For example, a pigment of red color actually absorbs light of all wavelengths, except that corresponding to red color.

- The characteristics used to distinguish one color from another are:
  - **Brightness** (or value) embodies the chromatic notion of intensity.
  - **Hue** is an attribute associated with the dominant wavelength in a mixture of light waves. It represents the dominant color as perceived by an observer (ex. orange, red, violet).
  - **Saturation** refers to the relative purity or the amount of white light mixed with a hue. Pure colors are fully saturated. Colors such as pink (red + white) and lavendar (violet + white) are less saturated, with the saturation being inversely proportional to the amount of white light added.

- Hue and saturation together are called **chromaticity**. A color can be described in terms of its brightness and chromaticity.



Tristimulus values

- The amounts of red, green, and blue needed to form any particular color are called the **tristimulus** values and are denoted by *X*, *Y*, and *Z*, respectively.

- In general, color is specified by its three **trichromatic coefficients**:

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

  - Naturally, $x + y + z = 1$.

Chromaticity diagram

- Another approach to specifying colors is via the CIE **chromaticity diagram**, which represents color composition by means of x (red) and y (green) values.

- For any value of x (red) and y (green), the corresponding value of z (blue) is given by $z = 1 - (x + y)$.

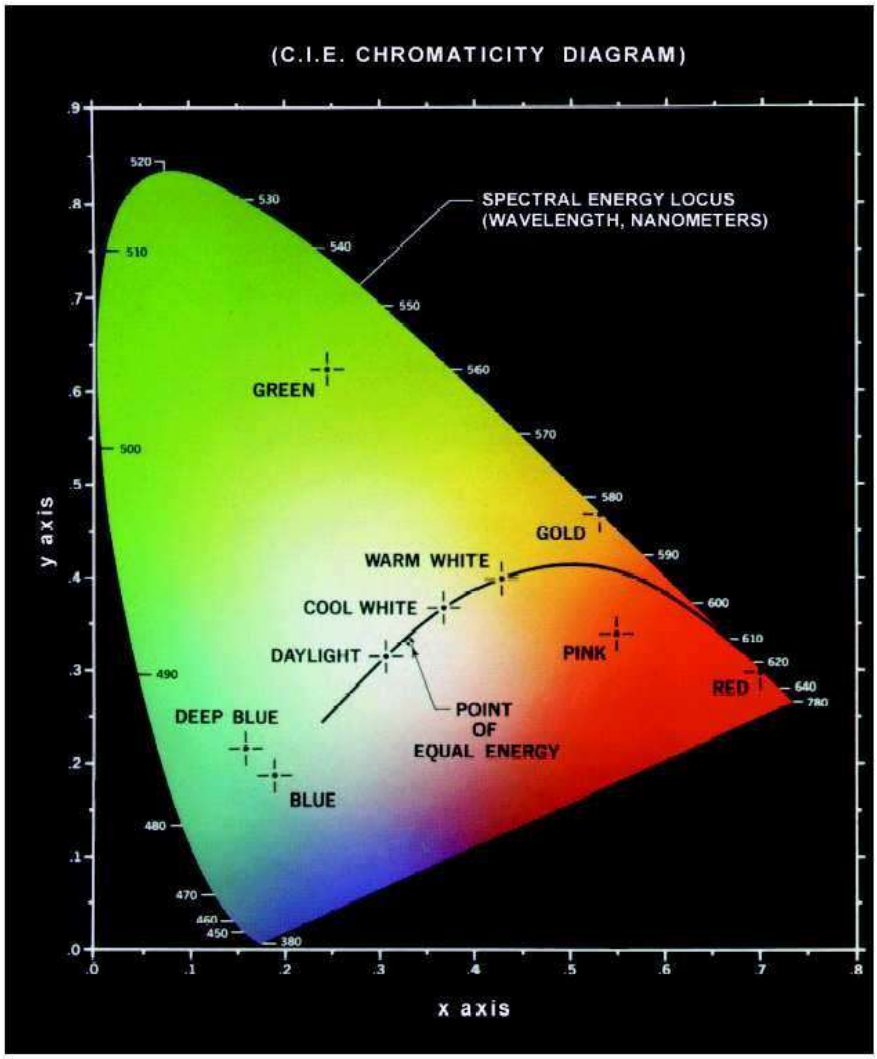


FIGURE 6.5
Chromaticity diagram. (Courtesy of the General Electric Co., Lamp Business Division.)

- The positions of various spectrum colors (completely saturated or "pure" colors) are indicated along the boundary of the tongue-shaped chromaticity diagram.

- Points inside this region represent some mixture of the pure colors.

- Point of equal energy corresponds to equal fractions of the three primary colors. It represents the *Commission Internationale de l'Eclairge* --- *The International Commission on Illumination* (CIE) standard for white light.

- As a point leaves the boundary and moves towards the center, more white light is added to the color and it becomes less saturated.

- The point of equal energy corresponds to zero saturation.

- The chromaticity diagram can be used for color mixing, since a line joining two points in the diagram represents all the colors that can be obtained by mixing the two colors additively.

- A line joining the point of equal energy to any point on the boundary represents different shades of that color.

- Similarly, the triangular region enclosed by the line segments joining three points in the chromaticity diagram represents all the colors that can be obtained by combining the three colors.

- This is consistent with the remark made earlier that the three pure primary colors by themselves cannot produce all the colors (unless we change the wavelengths as well).

- The triangular region shown in the figure below represents the typical range of colors (gamut of colors) produced by RGB monitors.

- The irregular region inside the triangular region represents the color gamut of modern high-quality color printers.

- Color printing is a complicated process and it is more difficult to control the color of printed object than it is to control the color displayed on a monitor.

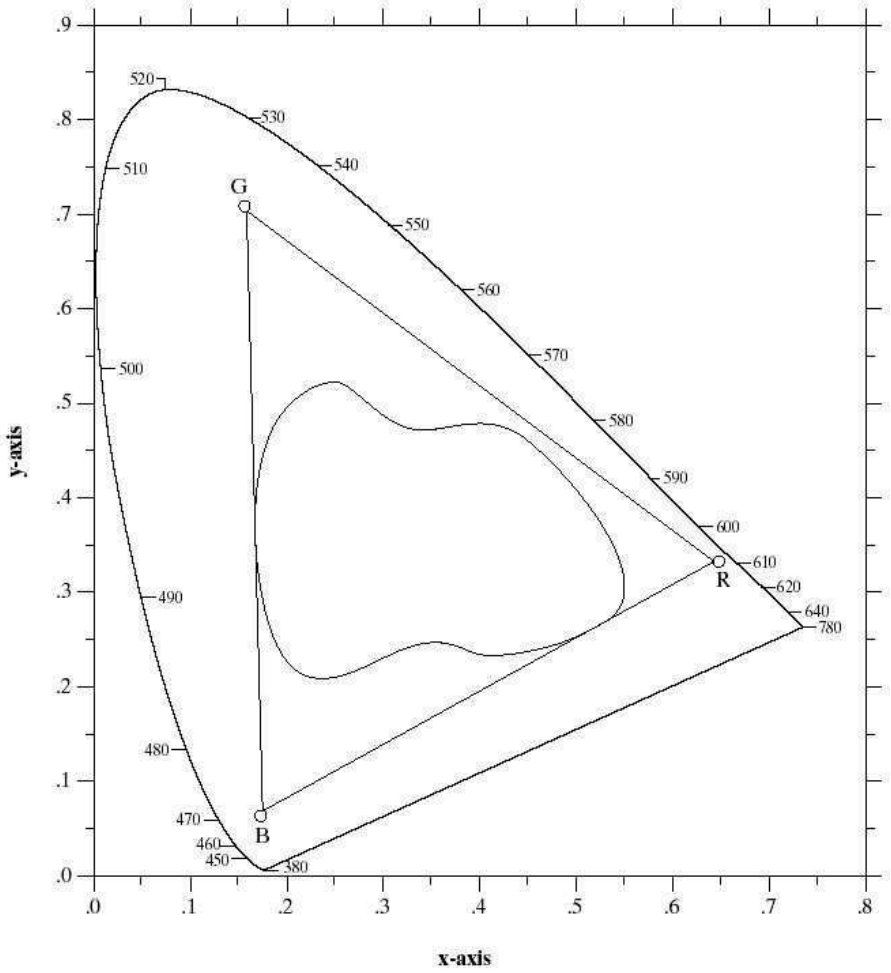


FIGURE 6.6 Typical color gamut of color monitors (triangle) and color printing devices (irregular region).

Color Models

- The purpose of a color model (or color space or color system) is to facilitate the specification of color in some standard fashion.

- A color model is a specification of a 3-D coordinate system and a subspace within that system where each color is represented by a single point.

- Most color models in use today are either based on hardware (color camera, printer) or on applications involving color manipulation (computer graphics, animation).

- In image processing, the hardware based color models mainly used are: RGB, CMYK, and HSI.

- The RGB (red, green, blue) color system is used mainly in color monitors and video cameras.

- The CMYK (cyan, magenta, yellow, black) color system is used in printing devices.

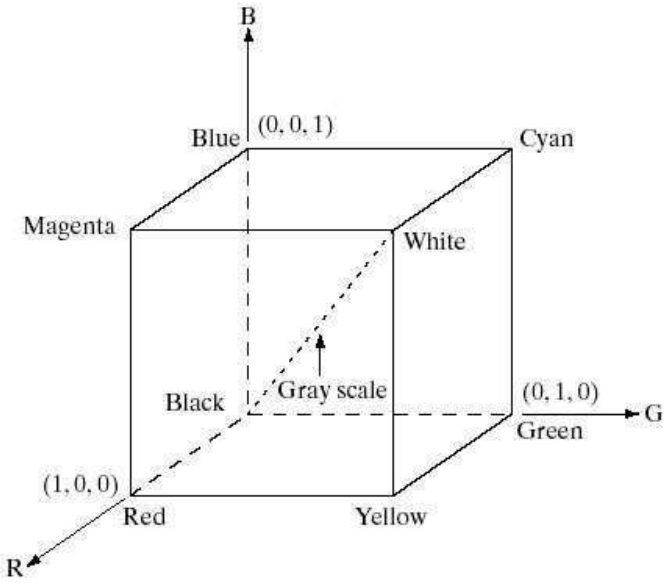
- The HSI (hue, saturation, intensity) is based on the way humans describe and interpret color. It also helps in separating the color and grayscale information in an image.

RGB Color model

- Each color appears in its primary spectral components of **red** (R), **green** (G), and **blue** (B).

- Mainly used for hardware such as color monitors and color video camera.

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).



- It is based on a Cartesian coordinate system. All color values are normalized so that the values of *R*, *G*, and *B* are in the range [0,1]. Thus, the color subspace of interest is the unit cube.

- The primary colors red, green, and blue correspond to three corners of the cube, whereas the secondary colors cyan, magenta, and yellow correspond to three other corners. Origin (0,0,0) represents black and (1,1,1) represents white.

- Grayscale (monochrome) is represented by the diagonal joining blak to white.

- Different points on or inside the cube correspond to different colors and can be represents as a vector or three values or coordinates. Each coordinate represents the amount of that primary color present in the given color.

- Images in the RGB model consist of three independent component images, one for each primary color.

- When fed to into an RGB monitor, these three images combine on the phosphor screen to produce a composite color image.

- The number of bits used to represent each pixel in RGB space is called **pixel depth**.

- For example, if eight bits are used to represent each of the primary components, each RGB color pixel would have a depth of 24 bits. This is usually referred to as a **full color** image.

- There are $2^{24} = 16{,}777{,}216$ unique colors possible in this system.

FIGURE 6.8 RGB 24-bit color cube.

+

=

+

- Although high-end monitors can display true 24-bit colors, more modest display devices are limited to smaller (typically 256) set of colors.

- Moreover, in many applications, it not useful to use more than a few (say 10-20) colors.

- Given the variety of display devices, it is useful to have a small subset of colors that are reproduced reliably and faithfully, independently of the display hardware specifics. This subset of colors is referred to as **safe RGB colors** or the set of **all-systems-safe colors**. They are also referred to as **safe web colors** or **safe browser colors** in internet applications.

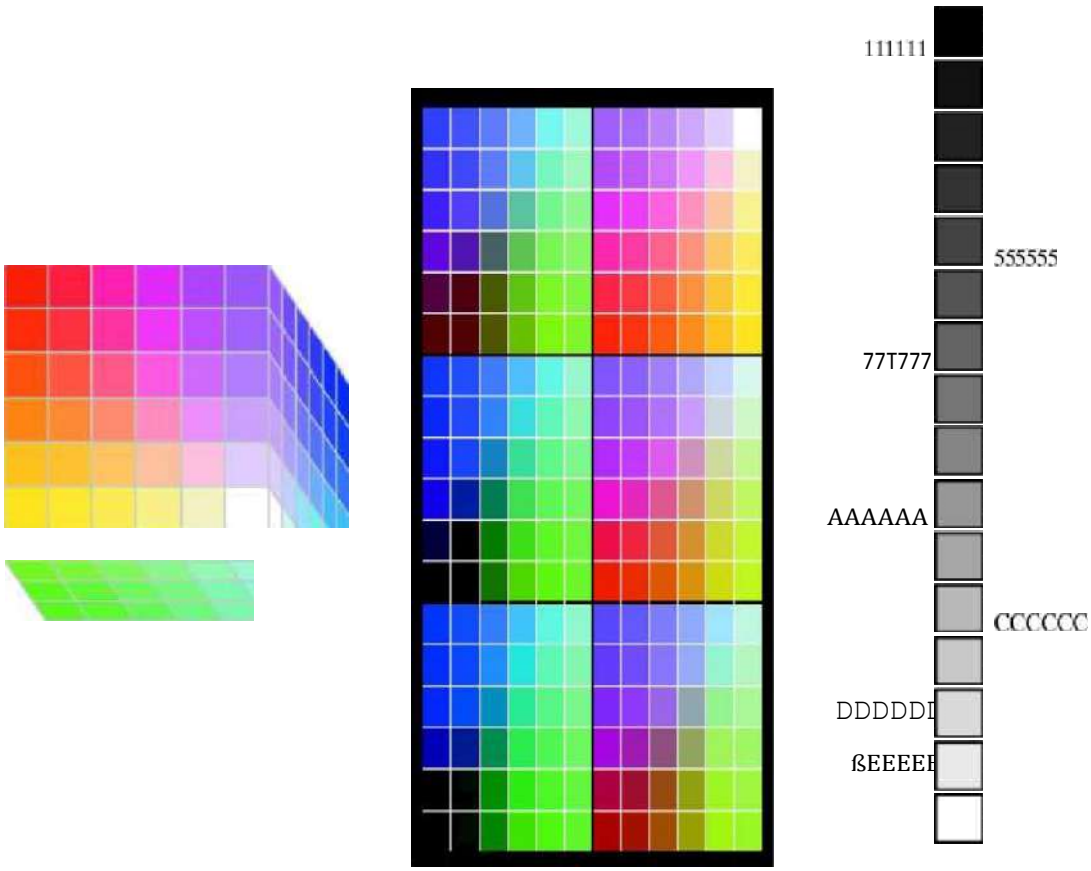
- Assuming 256 distinct colors as the minimum capability of any color display device, a standard notation to refer to these "safe" colors is necessary.

- Forty of these 256 colors are known to be processed differently by various operating systems, leaving 216 colors that are common to most systems.

- These 216 colors are formed by a combination of RGB values, where each component is restricted to be one of possible six values in the set {0, 51, 102, 153, 204, 255} or using hexadecimal notation {00, 33, 66, 99, CC, FF}. Note that all the values are divisible by 3.

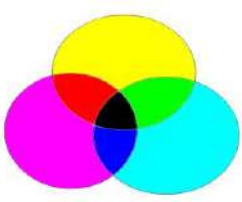
- These $2^6 = 216$ colors have become de facto standard for safe colors,

  especially in internet applications. They are commonly used, whenever it is desired that the colors viewed by most people appear the same.



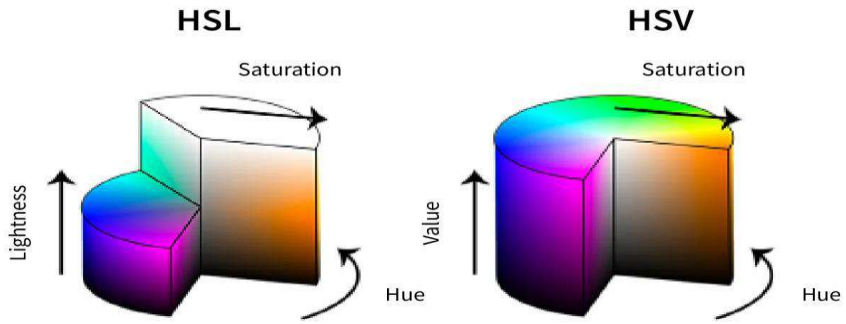
CMYK color model

CMYK stands for Cyan, Magenta, Yellow, and Key (Black), and it is a subtractive color model used in printing. In this model, colors are created by subtracting different amounts of cyan, magenta, yellow, and black ink from a white background. Unlike the RGB model, which is an additive model, the model is a subtractive model, which means that it starts with a white background and subtracts colors from it to create the desired hue. The CMYK color model is used in printing technologies such as offset printing, flexography, and digital printing.



HSL and HSV color models

HSL stands for Hue, Saturation, and Lightness, while HSV stands for Hue, Saturation, and Value. These color models are often used for color manipulation and adjustment in image processing. The HSL model represents colors in terms of their hue (the actual color), saturation (the intensity of the color), and lightness (the brightness of the color). The HSV model represents colors in terms of their hue, saturation, and value (a measure of the brightness of the color). The HSL and HSV models are particularly useful for adjusting color balance and enhancing color contrast in an image.



❖ Color transforms

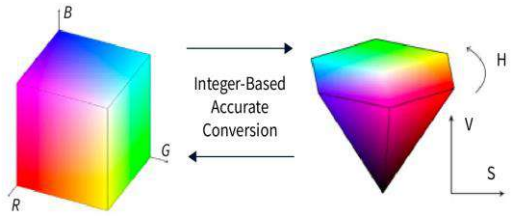
Color transforms are mathematical operations used to manipulate the color information in digital images. Here are some common types of color transforms in digital image processing.

Color correction transforms (e.g., white balance correction)

Color correction transforms are used to adjust the colors in an image to match a particular standard or desired appearance. One common example of a color correction transform is white balance correction, which is used to correct the color cast caused by the color temperature of the light source used to capture an image. White balance correction adjusts the color balance of an image to ensure that whites appear white, and colors appear as they should.

❖ Color space conversion transforms (e.g., RGB to HSL conversion)

Color space conversion transforms are used to convert an image from one color model to another. For example, an RGB image can be converted to the HSL or HSV color models for color manipulation and adjustment. Color space conversion transforms are often used in image compression, where converting an image to a different color space can reduce its file size without significant loss of image quality.



❖ **Color enhancement transforms (e.g., contrast enhancement)**

Color enhancement transforms are used to improve the visual appearance of an image by adjusting color contrast, saturation, and other color-related properties. For example, contrast enhancement can be used to increase the visual separation between colors in an image, while saturation enhancement can be used to make colors appear more vivid and Intense.

- Color quantization transforms (e.g., reducing the number of colors in an image)

Color quantization transforms are used to reduce the number of colors in an image by mapping the original color values to a smaller set of color values. This can be useful for reducing the file size of an image or for simplifying its visual appearance. One common example of a color quantization transform is dithering, which is used to simulate the appearance of additional colors by mixing existing colors in a pattern.

Lossless and Lossy data compression

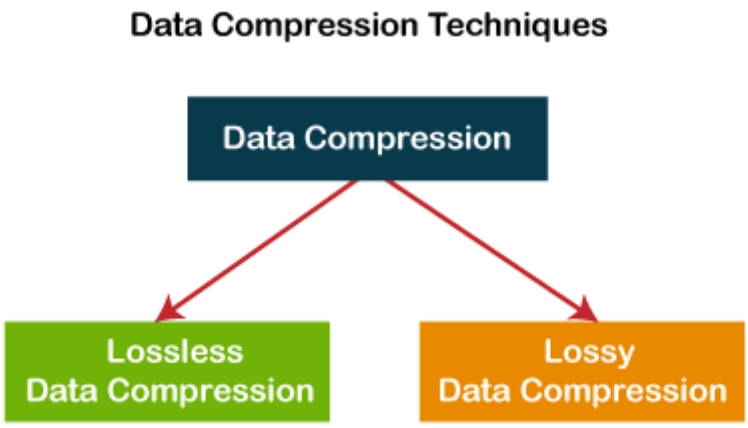
What is Data Compression

Data Compression is also referred to as **bit-rate reduction** or **source coding**. This technique is used to reduce the size of large files.

The advantage of data compression is that it helps us save our disk space and time in the data transmission.

There are mainly two types of data compression techniques

1. Lossless Data Compression
2. Lossy Data Compression

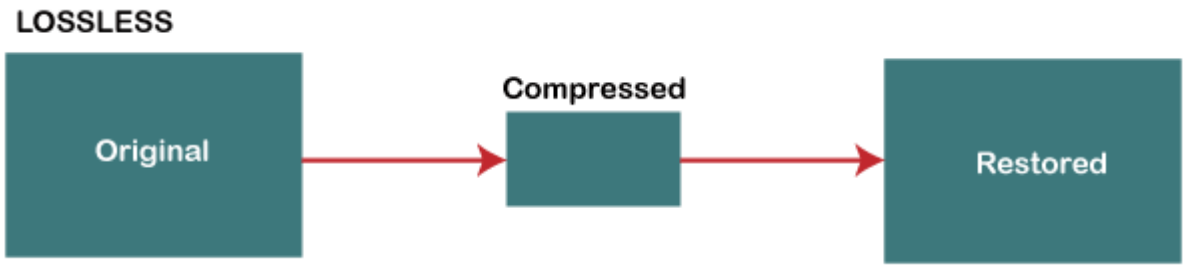


What is Lossless data compression

Lossless data compression is used to compress the files **without losing an original file's quality and data**. Simply, we can say that in lossless data compression, file size is reduced, but the quality of data remains the same.

The main advantage of lossless data compression is that we can restore the original data in its original form after the decompression.

Lossless data compression mainly used in the sensitive documents, confidential information, and PNG, RAW, GIF, BMP file formats.



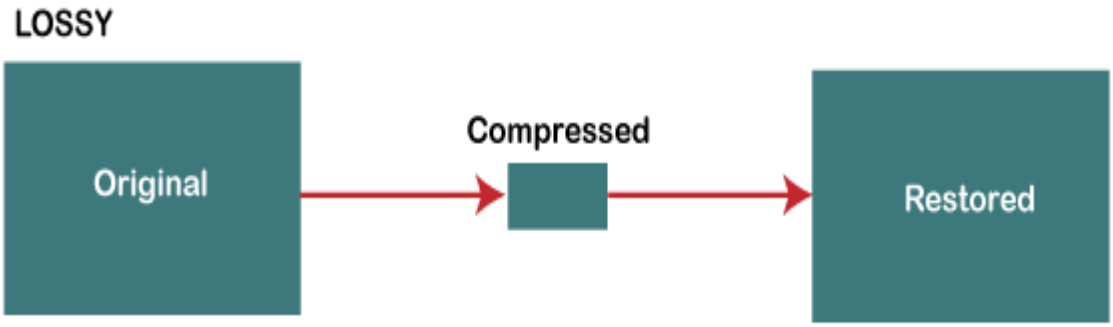
Some most important Lossless data compression techniques are -

1. Run Length Encoding (RLE)
2. Lempel Ziv - Welch (LZW)
3. Huffman Coding
4. Arithmetic Coding

What is Lossy data compression

Lossy data compression is used to compress larger files into smaller files. In this compression technique, some specific amount of **data and quality are removed (loss) from the original file**. It takes less memory space from the original file due to the loss of original data and quality. This technique is generally useful for us when the quality of data is not our first priority.

Lossy data compression is most widely used in JPEG images, MPEG video, and MP3 audio formats.



Some important Lossy data compression techniques are -

1. Transform coding
2. Discrete Cosine Transform (DCT)
3. Discrete Wavelet Transform (DWT)

Difference between lossless and lossy data compression

As we know, both lossless and lossy data compression techniques are used to compress data form its original size. The main difference between lossless and lossy data compression is that we can restore the lossless data in its original form after the decompression, but lossy data can't be restored to its original form after the decompression.

The below table shows the difference between lossless and lossy data compression -

S.No	Lossless data compression	Lossy data compression
1.	In Lossless data compression, there is no loss of any data and quality.	In Lossy data compression, there is a loss of quality and data, which is not measurable.
2.	In lossless, the file is restored in its original form.	In Lossy, the file does not restore in its original form.
3.	Lossless data compression algorithms are Run Length Encoding, Huffman encoding, Shannon fano encoding, Arithmetic encoding, Lempel Ziv Welch encoding, etc.	Lossy data compression algorithms are: Transform coding, Discrete Cosine Transform, Discrete Wavelet Transform, fractal compression, etc.
4.	Lossless compression is mainly used to compress text-sound and images.	Lossy compression is mainly used to compress audio, video, and images.
5.	As compare to lossy data compression, lossless data compression holds more data.	As compare to lossless data compression, lossy data compression holds less data.
6.	File quality is high in the lossless data compression.	File quality is low in the lossy data compression.
7.	Lossless data compression mainly supports RAW, BMP, PNG, WAV, FLAC, and ALAC file types.	Lossy data compression mainly supports JPEG, GIF, MP3, MP4, MKV, and OGG file types.

❖ ***Morphological operations***

Introduction

Image filters and thresholds enable us to detect structures of various shapes and sizes for different applications. Nevertheless, despite our best efforts, the binary images produced by our thresholds often still contain inaccurate or undesirable detected regions. They could benefit from some extra cleaning up.

At this stage, we are primarily working with shapes – morphology – so most of the techniques we describe here are often called morphological operations.

Ω Morphological operations using rank filters

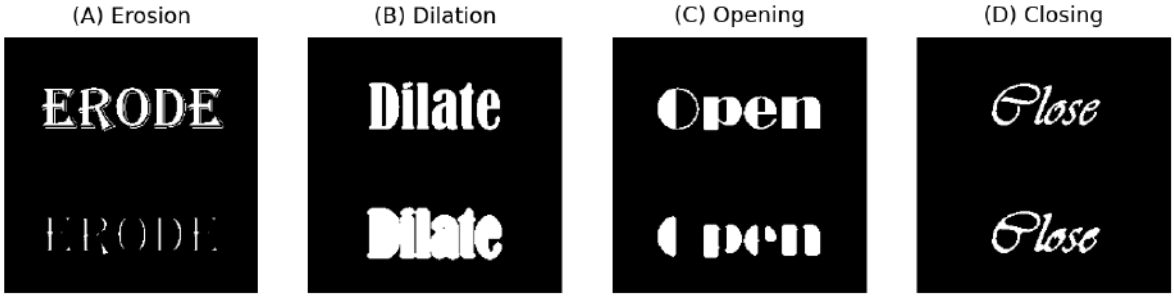


Fig. 102 Overview of erosion, dilation, opening and closing. The original image is shown at the top, while the processed part is at the bottom in each case.

❖ ***Erosion & dilation***

Our first two morphological operations, **erosion** and **dilation**, are actually identical to minimum and maximum filtering respectively, described in the previous chapter. The names erosion and dilation are used more often when speaking of binary images, but the operations are the same irrespective of the kind of image.

Structuring elements

The neighborhood used to calculate the result for each pixel is defined by a **structuring element**. This is similar to a filter kernel, except that it only

has values 0 and 1 (for ignoring or including the neighbourhood pixel, respectively).

Here, we assume the background value in our binary image is 0 (black) and foreground is 1 (white).
Erosion will make objects in the binary image smaller, because a pixel will be set to the background value if *any* other pixels in the neighborhood are background. This can split single objects into multiple pieces.

Conversely, **dilation** makes objects bigger, since the presence of a single foreground pixel anywhere in the neighborhood will result in a foreground output. This can also cause objects to merge.

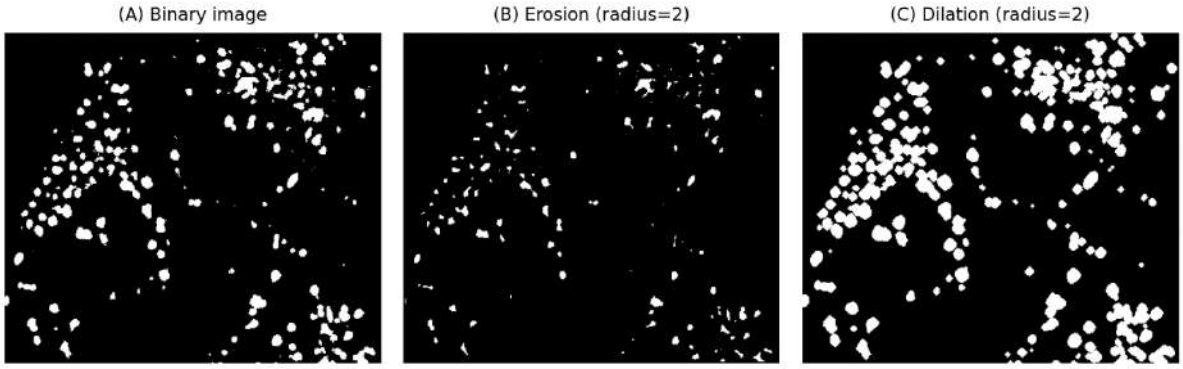


Fig. 103 The effects of erosion and dilation on a binary image of small structures.

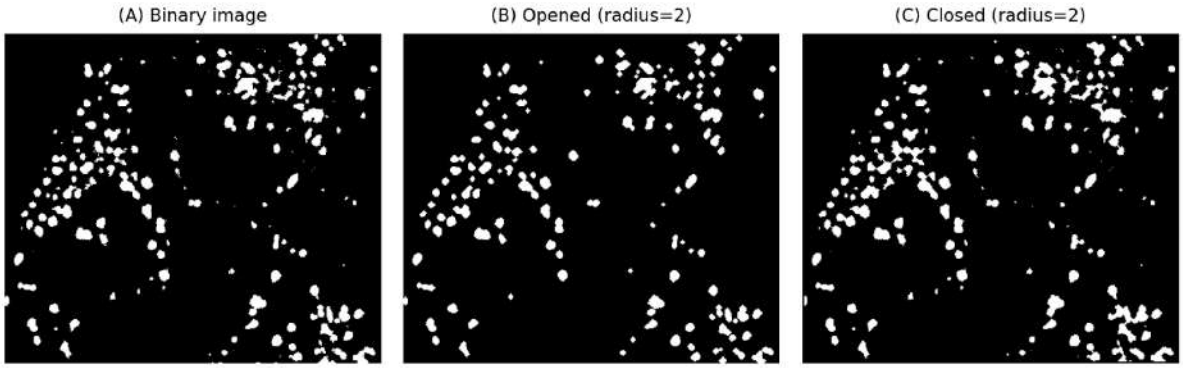
Opening & closing

The fact that erosion and dilation alone affect sizes can be a problem: we may like their abilities to merge, separate or remove objects, but prefer that they had less impact upon areas and volumes. Combining both operations helps achieve this.

Opening consists of an erosion followed by a dilation. It therefore first shrinks objects, and then expands whatever remains to *approximately* its original size.

Such a process is not as pointless as it may first sound. If erosion causes very small objects to completely disappear, clearly the dilation cannot make them reappear: they are gone for good. Barely-connected objects separated by erosion are also not reconnected by the dilation step.

Closing is the opposite of opening, i.e. a dilation followed by an erosion, and similarly changes the shapes of objects. The dilation can cause almost-connected objects to merge, and these often then remain merged after the erosion step. If you wish to count objects, but they are wrongly subdivided in the segmentation, closing may help make the counts more accurate.



The effects of opening and closing on a binary image of small structures.

Unlike when using erosion or dilation alone, the sizes of objects are largely preserved although the contours are modified. Opening has the effect of completely removing the smallest or thinnest objects.

Boundaries & outlines

We can make use of the operations above to identify outlines in a binary image. To do this, we first need a clear definition of what we mean by 'outline'.

The **inner boundary** may be defined as *the foreground pixels that are adjacent to background pixels*. We can determine the inner boundary by

- Duplicating the binary image
- Eroding with a 3×3 structuring element
- Subtracting the eroded image from the original

The **outer boundary** may be defined as *the background pixels that are adjacent to foreground pixels*. We can determine the outer boundary by

- Duplicating the binary image
- Dilating with a 3×3 structuring element
- Subtracting the original image from the dilated image

Ω ***Thicker boundaries***

There's no reason to limit outlines to being 1 pixel thick. Choosing a larger structuring element makes it possible create thicker outlines. We might also subtract an eroded image from a dilated image to identify a thicker boundary that contains both inner and outer pixels.

One application of creating thick boundaries in microscopy images of cells is to generate a binary image of the nuclei, and then a second binary image representing a ring around the nucleus. This makes it possible to make measurements that are likely to be within the cytoplasm, just outside the nucleus, without the task of identifying the full area of the cell – which is often difficult if the cell or membrane are not clearly visible.

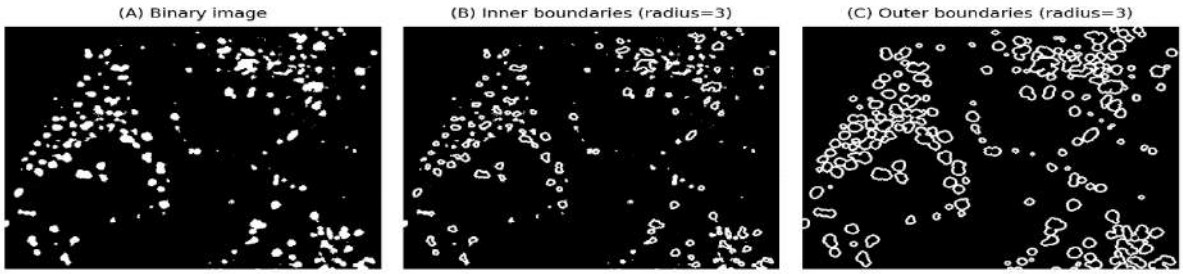


Fig. 105 Calculating inner and outer boundaries, using erosion or dilation. The radius of the structuring element can be used to tune the boundary thickness.

❖ ***Finding local minima & maxima***

Erosion and dilation can be used to find pixels that are **local maxima** or **local minima** very easily, with the caveat that the results are inexact and often unusable. Nevertheless, the trick works 'well enough' sufficiently often to be worth knowing.

Here, we focus on maxima; the process for detecting local minima is identical, except that either the image should be inverted or erosion used instead of dilation.

A local maximum can be defined as a pixel with a value greater than all its neighbors, or a connected group of pixels with the same higher value than the surrounding pixels. An easy way to detect these pixels is to dilate the image with 3×3 maximum filter, and check for pixel values that are

unchanged (i.e. where the pixel was already a maximum within its neighborhood).

This is inexact because it does not *only* identify maxima; it also detections some ‘plateaus’ where pixels have identical values to their neighbors. In practice, this is not always a problem because noise can make plateaus virtually non-existent for many real-world images (at least ones that haven’t been clipped).

A bigger problem is that the approach often identifies far too many maxima to be useful

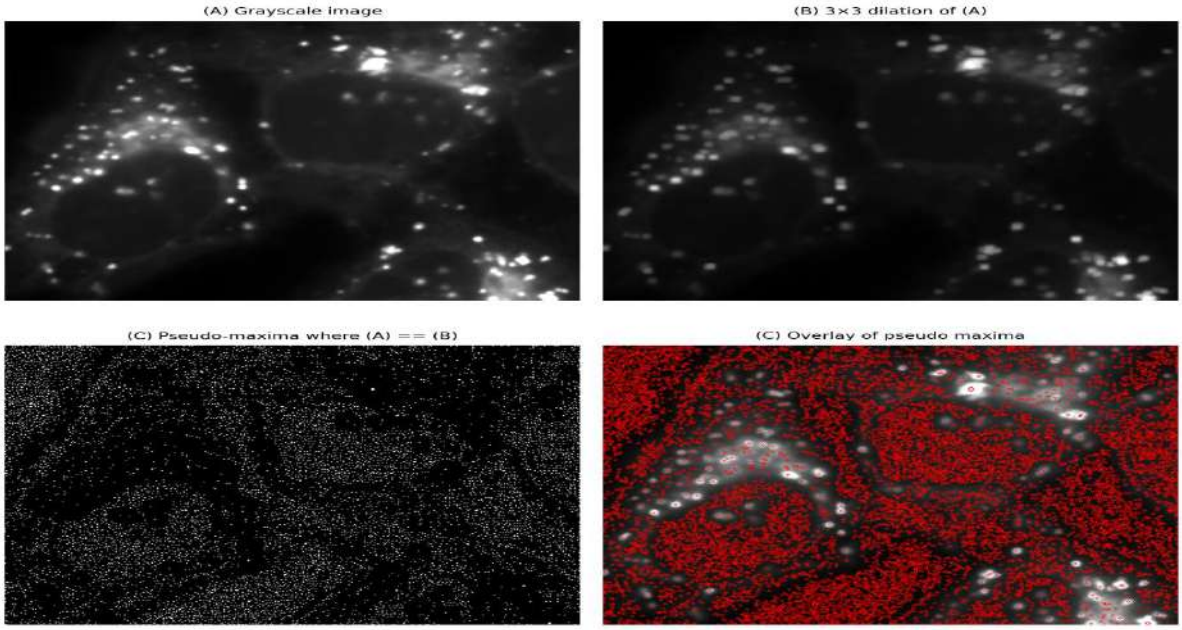


Fig. 106 Identifying local maxima with the help of a 3×3 dilation tends to find too many maxima to be useful.

We can reduce these by either increasing the size of the maximum filter (therefore requiring pixels to be maximal across a larger region), or by pre-smoothing the image (usually with a Gaussian filter). However, tuning the parameters becomes difficult.

We will see an alternative approach that is often more intuitive in H-Maxima & H-Minima.

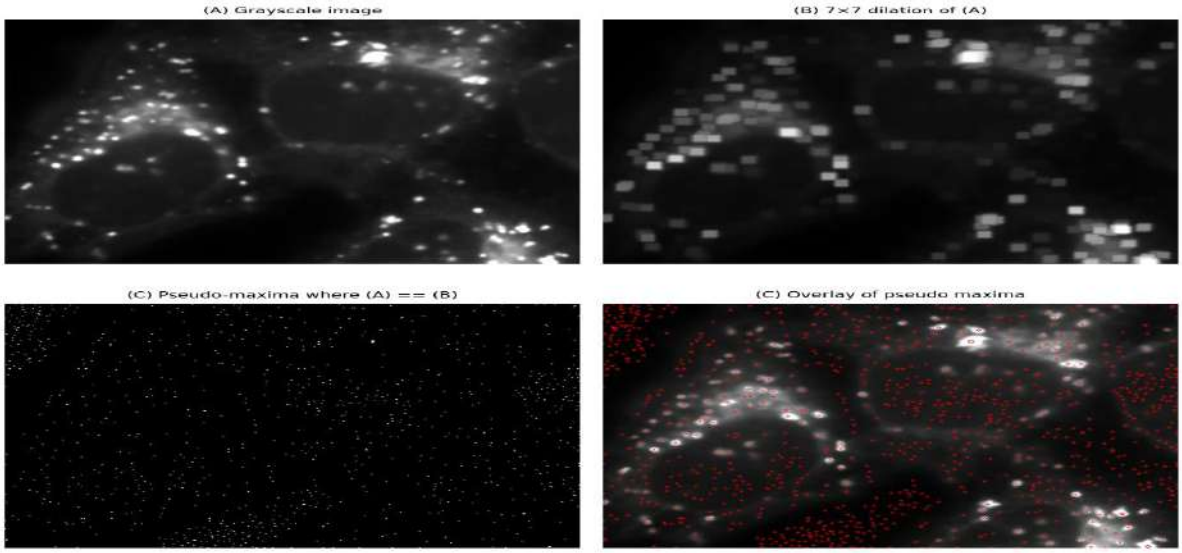


Fig. 107 Identifying local maxima with the help of a larger dilation (here, 7×7 pixels) can sometimes give better results than using a smaller dilation

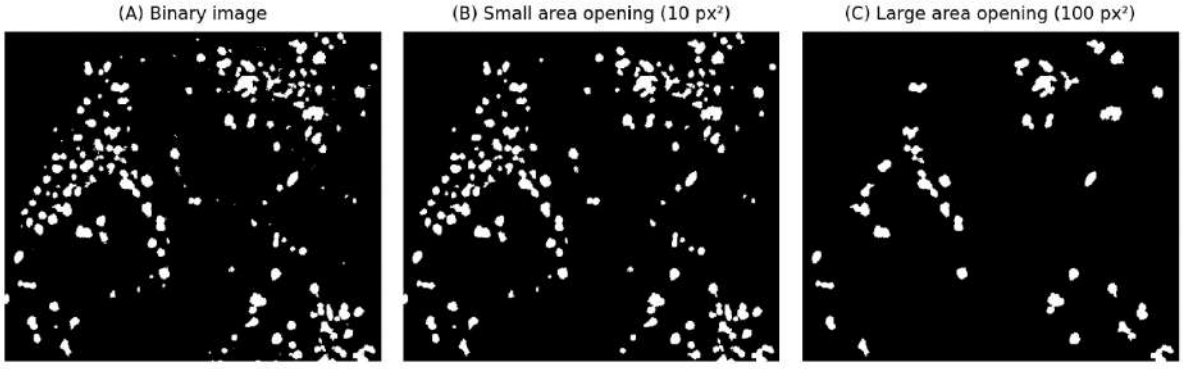
❖ ***More morphological operations***

Area opening

Area opening is similar to *opening*, except it avoids the need for any kind of maximum or minimum filtering.

It works by identifying **connected components** in the binary image, which are contiguous regions of foreground pixels. For each connected component, the number of pixels is counted to give an area in px^2. If the area of a component falls below a specified area threshold, the pixels for that component are set to the background, i.e. the component is removed.

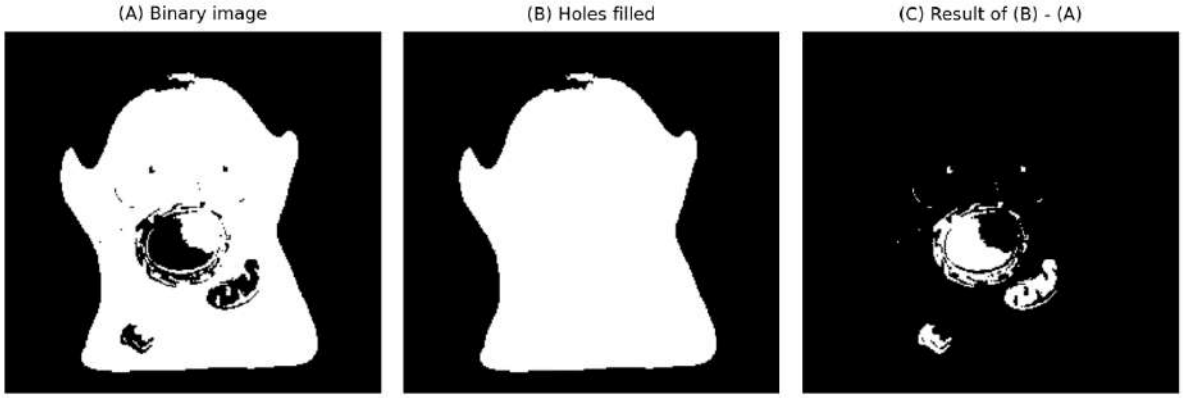
Area opening is often preferable to *opening*, because it has *no impact* on the shape of any structures larger than the area threshold. It simply applies a minimum area threshold, removing everything smaller.



Filling holes

Filling holes involves identifying connected components of *background pixels* that are entirely surrounded by foreground pixels. These components are then 'flipped' to become foreground pixels instead.

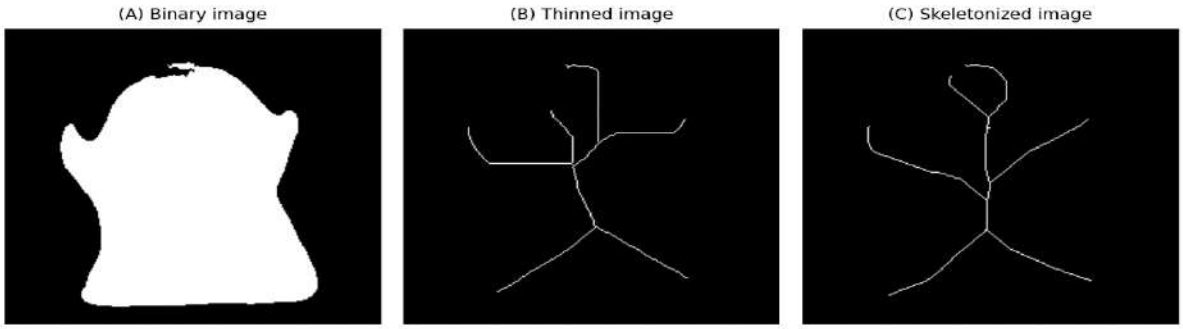
Should we then want to identify the holes themselves, we can subtract the original image from the filled image.



Filling holes in a binary image. Image subtraction makes it possible to extract the holes themselves.

Ω Thinning & skeletonization

Thinning and **skeletonization** are related operations that aim to 'thin down' objects in a binary image to just their centerlines. They are particularly useful with filamental or tube-like structures, such as axons or blood vessels.



The effects of thinning and skeletonization on a binary image.

What's the difference between thinning & skeletonization?

The truth is: I'm not entirely sure. There is quite a bit of overlap in the literature, and I've seen the same algorithm referred to by both names. Furthermore, there are different thinning algorithms that give different results; the situation is similar for skeletonization algorithms.

Software occasionally offers both thinning and skeletonization, but often just offers one or the other. It's worth trying any thinning/skeletonization methods available to see which performs best for any particular application.

❖ Morphological reconstruction

Morphological reconstruction is a somewhat advanced technique that underpins several powerful image processing operations. It's useful with both grayscale and binary images.

Morphological reconstruction requires two images of the same size: a **marker** image and a **mask** image. The pixel in the *mask* image should all have values greater than or equal to the corresponding pixels in the *marker* image.

The reconstruction algorithm progressively *dilates* the marker image (e.g. applies a 3×3 maximum filter), while constraining the marker to remain 'within' the mask; that is, the pixel values in the marker are never allowed to exceed the values in the mask. This dilation is repeated iteratively until the marker cannot change any further without exceeding the mask. The output is the new marker image, after all the dilations have been performed.

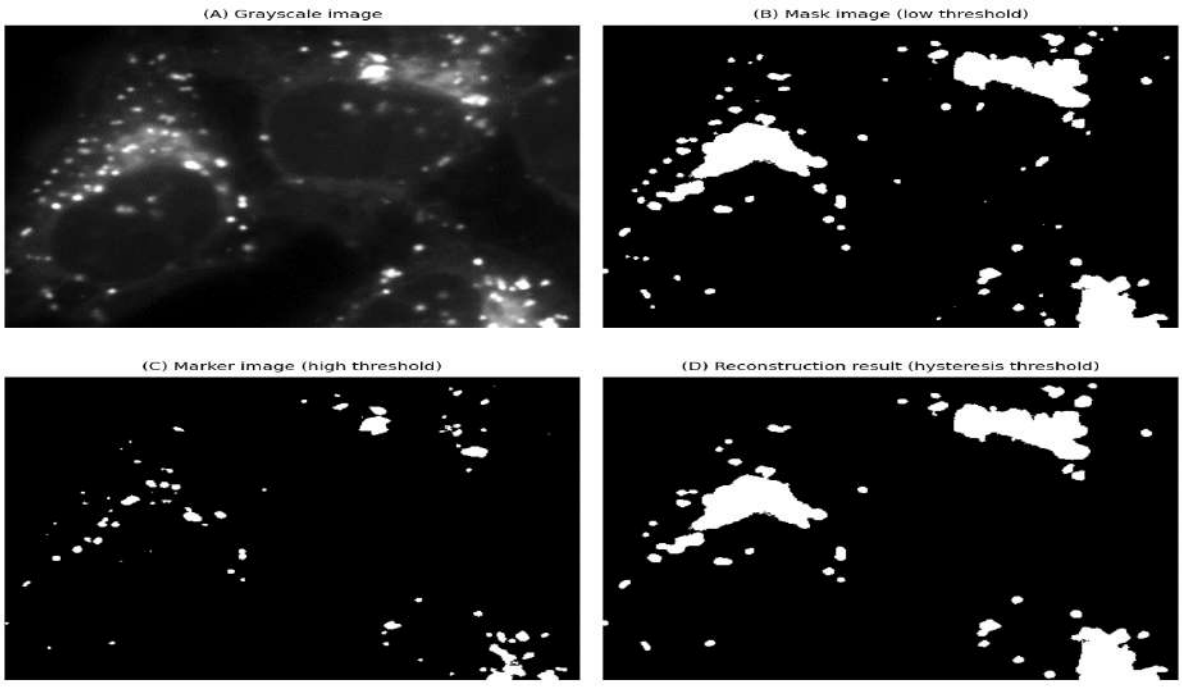
Some examples will help demonstrate how this works and why it's useful. The crucial difference in the methods below is how the marker and mask images are created.

❖ ***Hysteresis thresholding***

One use of morphological reconstruction is to implement a **double threshold**, also known as **hysteresis thresholding**.

For *low threshold* and *high threshold*, I assume we're detecting light structures on a dark background.
This involves defining both a **low threshold** and a **high threshold.** The low threshold operates like any global threshold to identify regions. However, a region is discarded from the binary image if it does not also contain at least one pixel that exceeds the high threshold.

This is achieved using morphological reconstruction by defining the *marker* as all pixels exceeding the high threshold, and the *mask* as all pixels exceeding the low threshold. The markers will expand to fill the mask regions that contain them. But any mask regions that don't contain marker pixels are simply ignored.



The size and area of the objects detected by this method are determined by the low threshold, but at least one of the pixel values within the object must exceed the high threshold. This slightly mitigates the problem of a

single global threshold having a huge impact on analysis results, by the same threshold simultaneously influencing both what is detected and its size.

❖ ***H-Maxima & H-Minima***

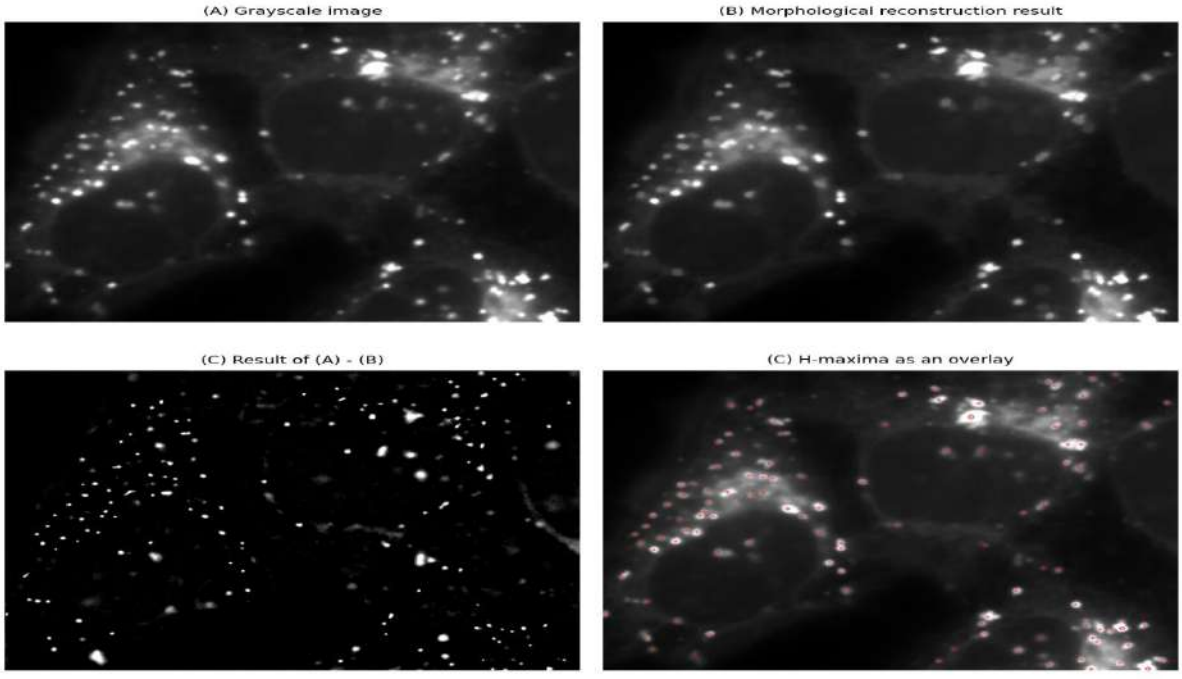
We saw previously that we could (kind of) identify local maxima in a very simple way using an image dilation, but the results are often too inaccurate to be useful.

H-Maxima and **H-Minima** can help us overcome this. These operations both require only one intuitive parameter: they enable us to identify maxima or minima using a local intensity threshold *H*.

This is achieved using morphological reconstruction. For H-maxima, the process is:

- Set the original grayscale image as the *mask*
- Subtract *H* from the mask to create the *markers*
- Apply morphological reconstruction using the markers and mask
- Subtract the reconstruction result from the *mask*
- Threshold the subtracted image with a global threshold of *H*

The main steps are illustrated in We can apply the same process to an inverted image to find *H-minima*.



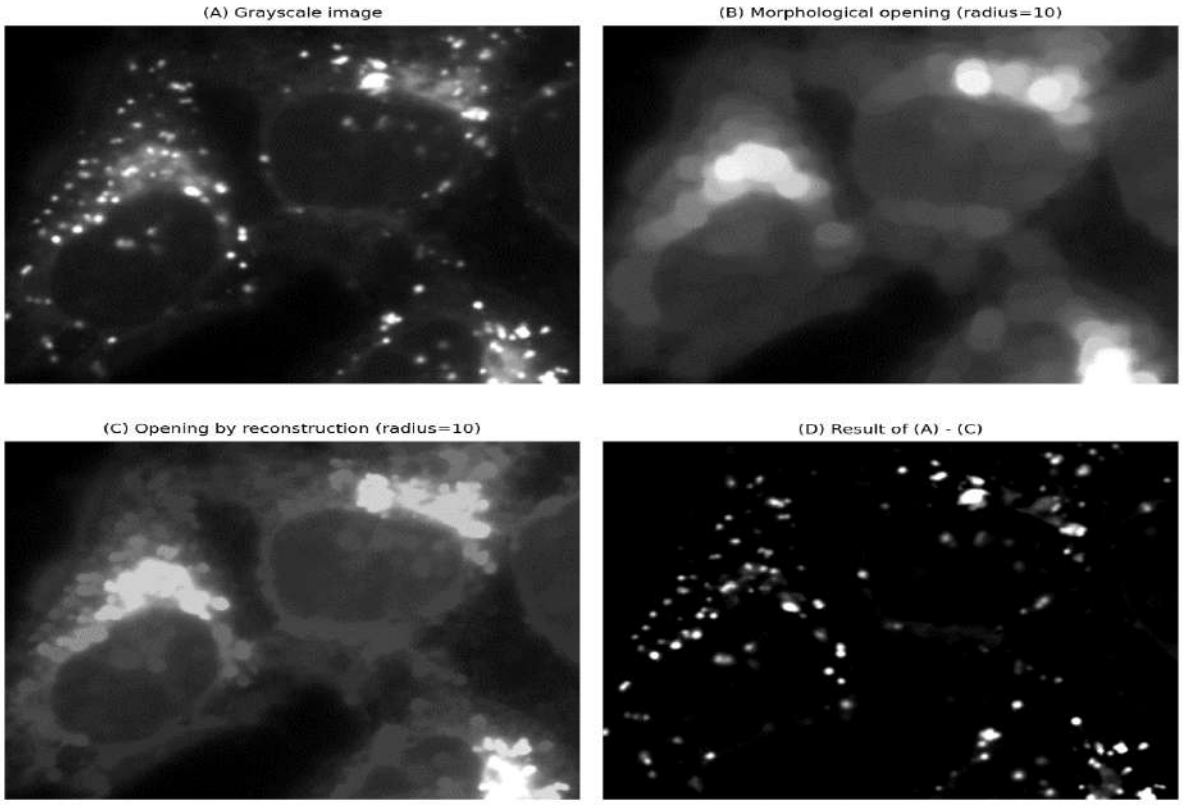
Calculating H-maxima using morphological reconstruction. Here, *H* is set (arbitrarily) to be the image standard deviation.

❖ ***Opening & closing by reconstruction***

H-maxima and H-minima use morphological reconstruction to effectively generate a background image that can be subtracted from the original. We do this by subtracting a constant *H*, which acts as a local intensity threshold.

We can also use morphological reconstruction to generate a background image based upon spatial information, rather than an intensity threshold *H*, by using **opening by reconstruction**. This effectively introduces a size component into our local threshold. **Closing by reconstruction** is an analogous operation that can be defined using morphological closing.

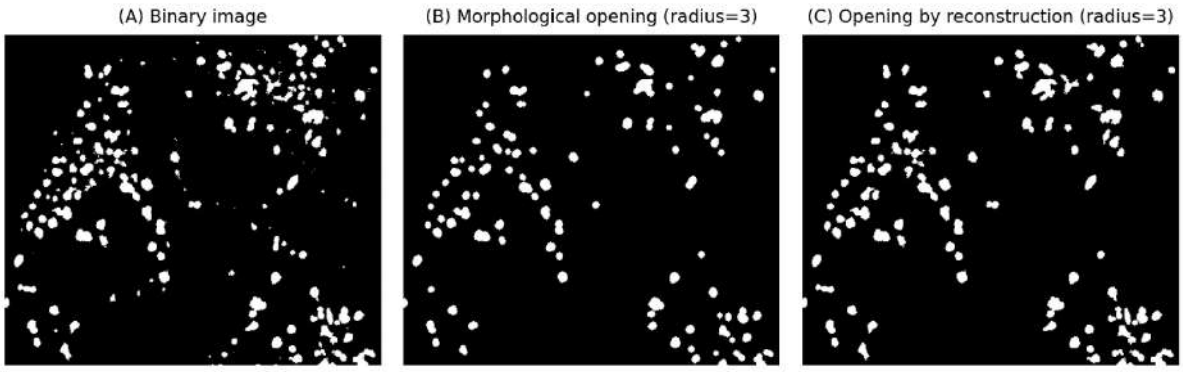
The starting point for opening by reconstruction is a *morphological opening* as defined above, i.e. an erosion followed by a dilation. This defines the marker image. The original image is used as the mask.



Using opening by reconstruction to obtain a background estimate. The estimate can be subtracted from an image before applying a global threshold.

As before, opening alone removes structures that are smaller than the structuring element, while slightly affecting the shapes of everything else. Opening by reconstruction essentially adds some further (constrained) dilations so that the structures that were *not* removed are more similar to how they were originally. This can make opening by reconstruction more attractive for generating background images that will be used for subtraction.

Opening by reconstruction can also be applied to binary images as an alternative to *opening* and *area opening*. Like area opening, opening by reconstruction is able to remove some objects while retaining the shapes of larger objects exactly.



Using opening by reconstruction to remove small (and thin) objects from a binary image, while retaining the original shape of everything that remains.

****