Identify and explain five color models used in digital image processing.

**RGB (Red, Green, Blue)**

**Description:**

The RGB model is an **additive color model** where red, green, and blue light are combined to create other colors. Each color is represented as a **tuple** of three values ranging from **0 to 255.**

**Use Cases and Applications:**

Digital displays (monitors, TVs, smartphones).

Web design and photography.

Gaming setups and lighting systems.

**Code Example:**

import cv2

img = cv2.imread("/content/Lenna.png") # Load an image in BGR format

print("Original Color Image")

cv2\_imshow(img)

B, G, R = cv2.split(img) # Splitting into Red, Green, Blue channels

zeros = np.zeros(img.shape[:2], dtype="uint8")# Creating black mask for visualization

print("Red Channel")

cv2\_imshow(cv2.merge([zeros, zeros, R])) # Red in BGR

print("Green Channel")

cv2\_imshow(cv2.merge([zeros, G, zeros])) # Green in BGR

print("Blue Channel")

cv2\_imshow(cv2.merge([B, zeros, zeros])) # Blue in BGR

**HSV (Hue, Saturation, Value)**

**Description:**

HSV represents colors in terms of **hue** (color type), **saturation** (intensity), and **value** (brightness). It is particularly intuitive for color manipulation.

**Use Cases and Applications:**

Color-based object detection.

Image segmentation and filtering.

Artistic applications like painting.

**Code Example:**

import cv2

import numpy as np

img = cv2.imread('image.jpg') # Load an image

hsv\_img = cv2.cvtColor(img, cv2.COLOR\_BGR2HSV) # Convert RGB to HSV

cv2.imshow('HSV Image', hsv\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

**CMY (Cyan, Magenta, Yellow)**

**Description:**

CMY is a **subtractive color model** used in **printing**. It works by subtracting varying amounts of cyan, magenta, and yellow pigments from white.

**Use Cases and Applications:**

Commercial printing (magazines, books).

Color photocopiers and fabric printing.

**Code Example:**

def rgb\_to\_cmy(r, g, b):

c = 1 - r / 255

m = 1 - g / 255

y = 1 - b / 255

return c, m, y

r, g, b = 255, 0, 0 # Red color in RGB

c, m, y = rgb\_to\_cmy(r, g, b)

print(f"Cyan: {c}, Magenta: {m}, Yellow: {y}")

**CMYK (Cyan, Magenta, Yellow, Black)**

**Description:**

CMYK enhances CMY by adding black (K) for better depth and color accuracy in printing.

**Use Cases and Applications:**

Offset printing and graphic design.

Product packaging and marketing materials

**Code Example:**

def rgb\_to\_cmyk(r, g, b):

r\_prime = r / 255

g\_prime = g / 255

b\_prime = b / 255

k = 1 - max(r\_prime, g\_prime, b\_prime)

if k == 1:

c, m, y = 0, 0, 0

else:

c = (1 - r\_prime - k) / (1 - k)

m = (1 - g\_prime - k) / (1 - k)

y = (1 - b\_prime - k) / (1 - k)

return c, m, y, k

r, g, b = 255, 0, 0 # Red color in RGB

c, m, y, k = rgb\_to\_cmyk(r, g, b)

print(f"Cyan: {c}, Magenta: {m}, Yellow: {y}, Black: {k}")



**HSL (Hue, Saturation, Lightness)**

**Description:**

HSL separates hue (color), saturation (intensity), and lightness (brightness). It is similar to HSV but focuses on lightness instead of value.

**Use Cases and Applications:**

Web development for harmonious color combinations.

Color grading in video editing.

**Code Example:**

# Convert BGR to HLS

hls\_img = cv2.cvtColor(img, cv2.COLOR\_BGR2HLS)

H, L, S = cv2.split(hls\_img)

print("Hue Channel")

cv2\_imshow(H)

print("Lightness Channel")

cv2\_imshow(L)

print("Saturation Channel")

cv2\_imshow(S) 

**YCbCr (Luminance-Chrominance) Model**

**Description:**

YCbCr separates luminance (Y) from chroma components (Cb for blue difference and Cr for red difference). It is widely used in video compression.

**Use Cases and Applications:**

JPEG image compression.

Digital video formats like MPEG.

**Code Example:**

ycrcb\_img = cv2.cvtColor(img, cv2.COLOR\_BGR2YCrCb) # Convert BGR to YCrCb

Y, Cr, Cb = cv2.split(ycrcb\_img)

print("Luminance (Y) Channel")

cv2\_imshow(Y)

print("Chrominance (Cr) Channel")

cv2\_imshow(Cr)

print("Chrominance (Cb) Channel")

cv2\_imshow(Cb) 

**LAB (Lightness, A, B)\*\***

**Description:**

LAB represents colors using lightness (L\*), green-red component (A\*), and blue-yellow component (B\*). It is device-independent.

**Use Cases and Applications:**

Color correction in photography.

Image enhancement tasks.

**Code Example:**

lab = cv2.cvtColor(img, cv2.COLOR\_BGR2LAB)

cv2.imshow("LAB Image", lab)

cv2.waitKey(0)

cv2.destroyAllWindows()



**Grayscale**

**Description:**

Grayscale discards color information by representing each pixel with a single intensity value ranging from 0 to 255.

**Use Cases and Applications:**

Feature extraction in computer vision9.

Edge detection algorithms9.

**Code Example:**

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

cv2.imshow("Grayscale Image", gray)

cv2.waitKey(0)

cv2.destroyAllWindows()

| **Color Model** | **Description** | **Use Cases/Applications** | **Requirements** |
| --- | --- | --- | --- |
| RGB | Additive model for digital displays. | Digital displays, web design, photography. | Brightness and color accuracy. |
| HSV | Hue, Saturation, Value for intuitive color manipulation. | Color-based object detection, image segmentation. | Color intensity and brightness control. |
| CMY | Subtractive model for printing without black. | Basic printing processes. | Pigment subtraction for color creation. |
| CMYK | Enhanced CMY with black for better depth. | Offset printing, graphic design. | High-quality color reproduction with black ink. |
| HSL | Hue, Saturation, Lightness for harmonious color combinations. | Web development, color grading. | Lightness and saturation balance. |
| LAB | Device-independent model for color correction. | Color correction in photography, image enhancement. | Accurate color representation across devices. |
| Grayscale | Single-channel intensity representation. | Feature extraction, edge detection. | Simple intensity-based image processing. |
| YCbCr | Separates luminance from chroma for video and image compression. | JPEG compression, digital video formats. | Efficient luminance and chroma separation. |