AI TRAINING SERIES

Day 09

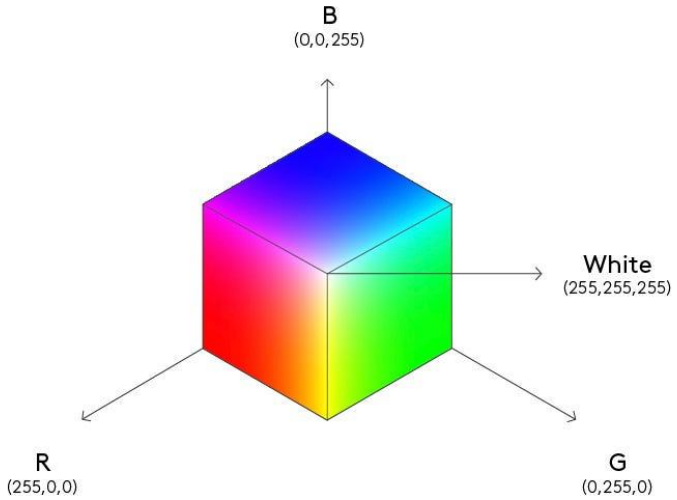
ROS & Image Processing

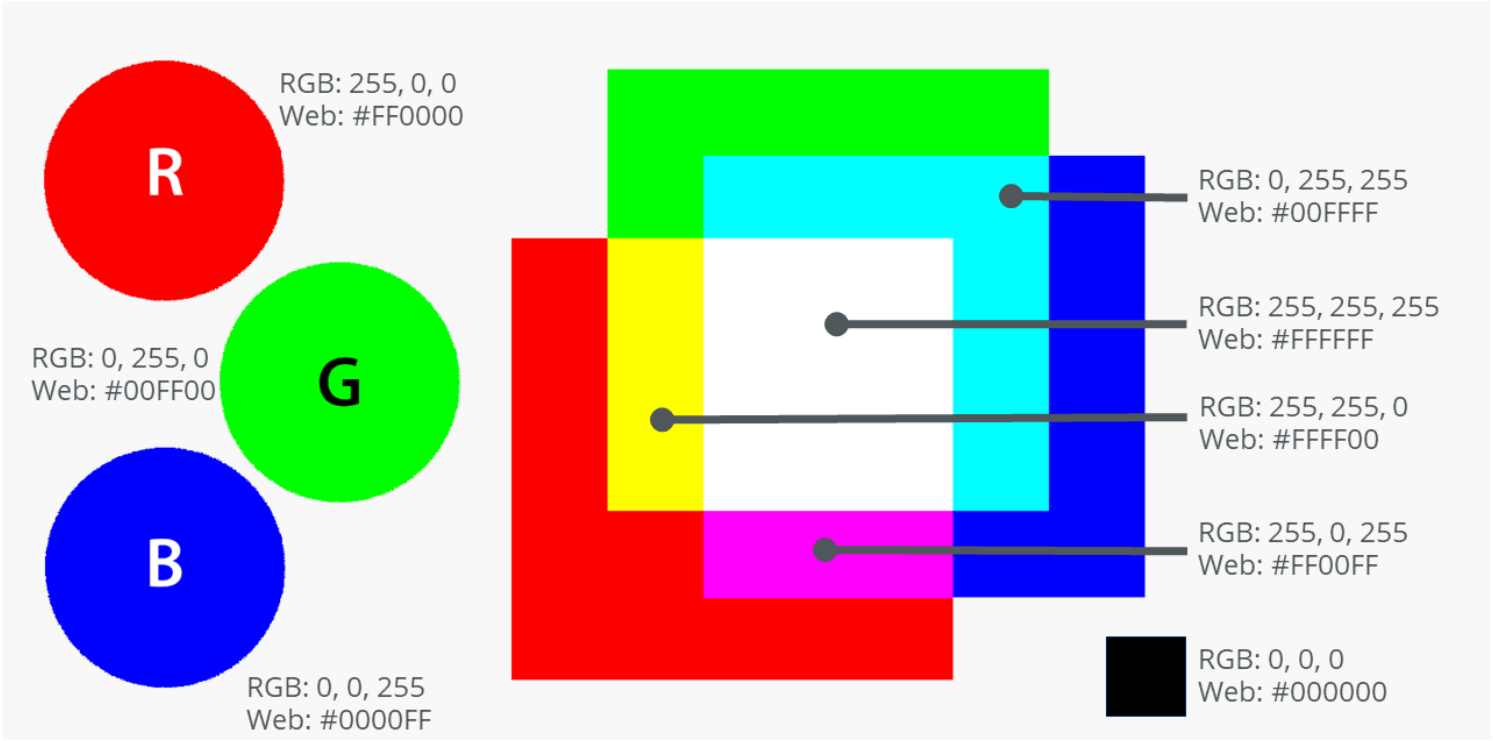
Contents

Ⅰ. Color Space

RGB Color Space

- 빨간색(Red), 초록색(Green), 파란색(Blue) 세 가지 색의 빛을 섞어서 다양한 색을 만드는 방식
- 각 색(R, G, B)은 0~255까지 밝기를 조절

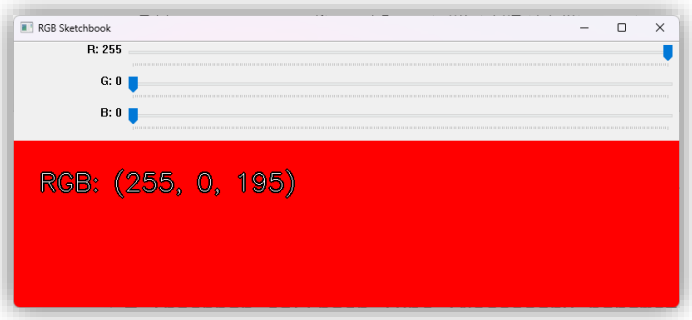




Ⅰ. Color Space

RGB Color Space (cont.)

- Trackbar 를 사용해 R,G,B 값을 control 하여 변하는 RGB color 값을 확인하는 예제 코드



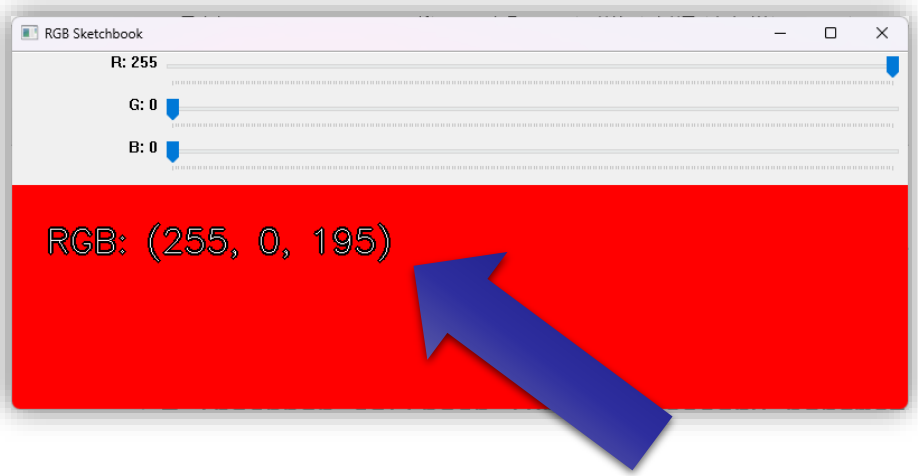
`img[:] = [b, g, r]`
- `img[:]` 이미지 전체 픽셀 (모든 행, 모든 열)
- `[b, g, r]` 각 픽셀의 색상 값을 BGR 순서로 지정

```python
import cv2
import numpy as np

# Trackbar callback func (Necessary parameter - do nothing here)
def nothing(x):
    pass

# 800x200 빈 스케치북 생성
img = np.zeros((200, 800, 3), np.uint8)

cv2.namedWindow('RGB Sketchbook')
cv2.createTrackbar('R', 'RGB Sketchbook', 0, 255, nothing)
cv2.createTrackbar('G', 'RGB Sketchbook', 0, 255, nothing)
cv2.createTrackbar('B', 'RGB Sketchbook', 0, 255, nothing)

while True:
    r = cv2.getTrackbarPos('R', 'RGB Sketchbook')
    g = cv2.getTrackbarPos('G', 'RGB Sketchbook')
    b = cv2.getTrackbarPos('B', 'RGB Sketchbook')
    img[:] = [b, g, r]  # OpenCV는 BGR 순서

    cv2.imshow('RGB Sketchbook', img)

    if cv2.waitKey(1) & 0xFF == 27:  # ESC 키로 종료
        break

cv2.destroyAllWindows()
```

I . Color Space

RGB Color Space (cont.)

- cv2.puttext 함수를 2번 사용하여 RGB 값을 아래처럼 text 로 출력해 주도록 구현하기

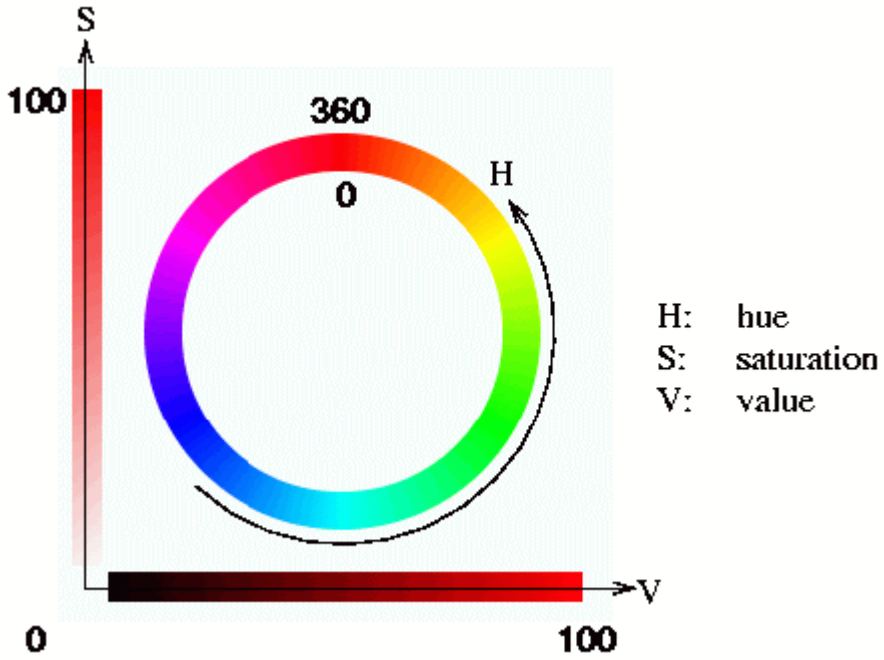


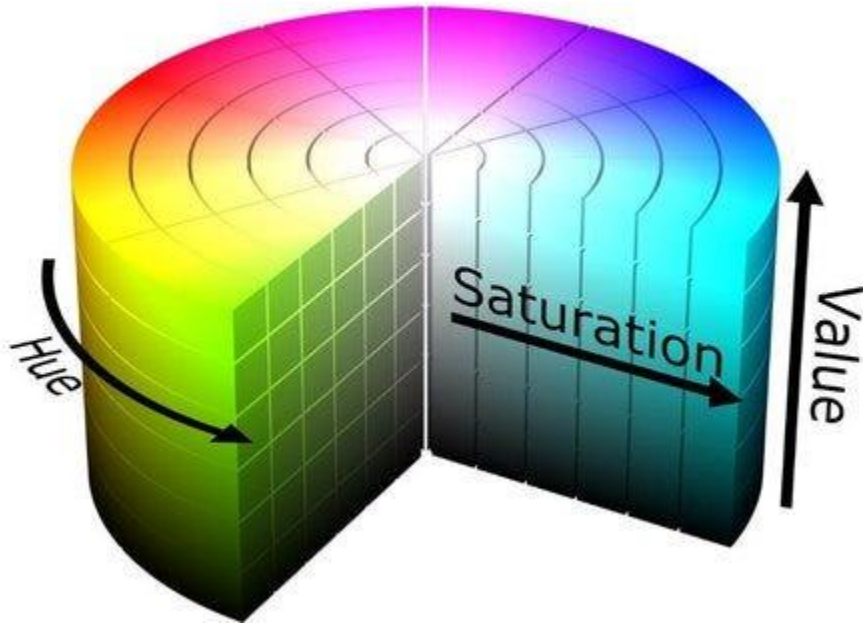

```python
import cv2
import numpy as np

# Trackbar callback func (Necessary parameter - do nothing here)
def nothing(x):
    pass

# 800x200 빈 스케치북 생성
img = np.zeros((200, 800, 3), np.uint8)

cv2.namedWindow('RGB Sketchbook')
cv2.createTrackbar('R', 'RGB Sketchbook', 0, 255, nothing)
cv2.createTrackbar('G', 'RGB Sketchbook', 0, 255, nothing)
cv2.createTrackbar('B', 'RGB Sketchbook', 0, 255, nothing)

while True:
    r = cv2.getTrackbarPos('R', 'RGB Sketchbook')
    g = cv2.getTrackbarPos('G', 'RGB Sketchbook')
    b = cv2.getTrackbarPos('B', 'RGB Sketchbook')
    img[:] = [b, g, r]  # OpenCV는 BGR 순서

    # 텍스트 표시
    text_rgb = f"RGB: ({r}, {g}, {b})"

    # 텍스트 배경 처리용 (검정 테두리)
    cv2.putText(img, text_rgb, (30, 60), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,0), 3)
    cv2.putText(img, text_rgb, (30, 60), cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 1)

    cv2.imshow('RGB Sketchbook', img)

    if cv2.waitKey(1) & 0xFF == 27:  # ESC 키로 종료
        break

cv2.destroyAllWindows()
```

Ⅰ. Color Space

HSV Color Space

- HSV는 색깔을 사람의 감각처럼 표현한 색 모델
- H (Hue, 색조), (Saturation, 채도), V (Value, 명도)

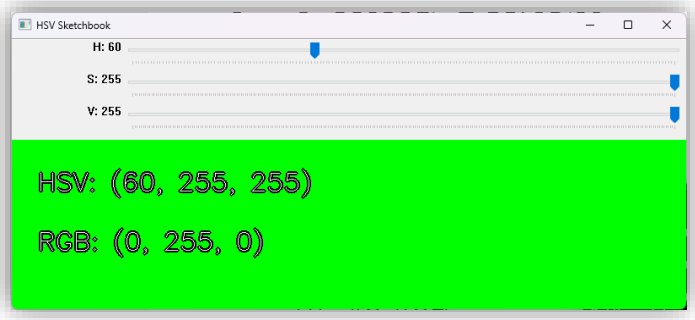




Ⅰ. Color Space

HSV Color Space (cont.)

- Trackbar 를 사용해 H,S,V 값을 control 하여 변하는 HSV color 값을 확인하는 예제 코드
- H 값은 0 ~ 359 까지로 표현되나 OpenCV 에서는 표현 범위의 한계(8bit: 0 - 255) 로 인해 H 값만 2로 나눈 값을 사용함 (0~359 / 2 = 0~179)



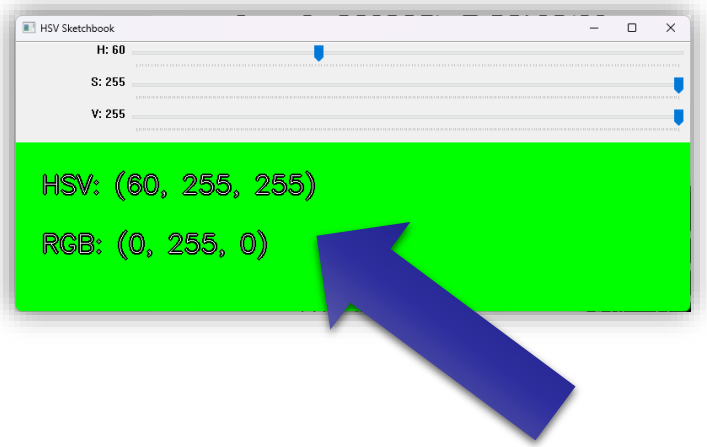
OpenCV 에서 기본적으로 사용하는 색공간은 BGR 이기 때문에 Display 출력을 위해 HSV 를 BGR 로 변경이 필요함

```python
import cv2
import numpy as np

def nothing(x): # Trackbar callback func (Necessary parameter - do nothing here)
    pass

# Create a blank image (sketchbook)
img = np.zeros((200, 800, 3), np.uint8)

# Create HSV Sketchbook window and add trackbars
cv2.namedWindow('HSV Sketchbook')
cv2.createTrackbar('H', 'HSV Sketchbook', 0, 179, nothing)  # Hue: 0~179 (OpenCV range)
cv2.createTrackbar('S', 'HSV Sketchbook', 0, 255, nothing)  # Saturation: 0~255
cv2.createTrackbar('V', 'HSV Sketchbook', 0, 255, nothing)  # Value: 0~255

while True:
    # Get current trackbar values
    h = cv2.getTrackbarPos('H', 'HSV Sketchbook')
    s = cv2.getTrackbarPos('S', 'HSV Sketchbook')
    v = cv2.getTrackbarPos('V', 'HSV Sketchbook')

    # Convert HSV to BGR (OpenCV uses BGR)
    hsv_color = np.uint8([[[h, s, v]]])
    b, g, r = cv2.cvtColor(hsv_color, cv2.COLOR_HSV2BGR)[0][0]

    # Apply background color
    img[:] = [b, g, r]

     # Show image
    cv2.imshow('HSV Sketchbook', img)

    if cv2.waitKey(1) & 0xFF == 27:
        break

cv2.destroyAllWindows()
```

Ⅰ. Color Space

HSV Color Space (cont.)

- cv2.puttext 함수를 사용하여 RGB 값과 HSV 값을 아래처럼 text 로 출력해 주도록 구현하기

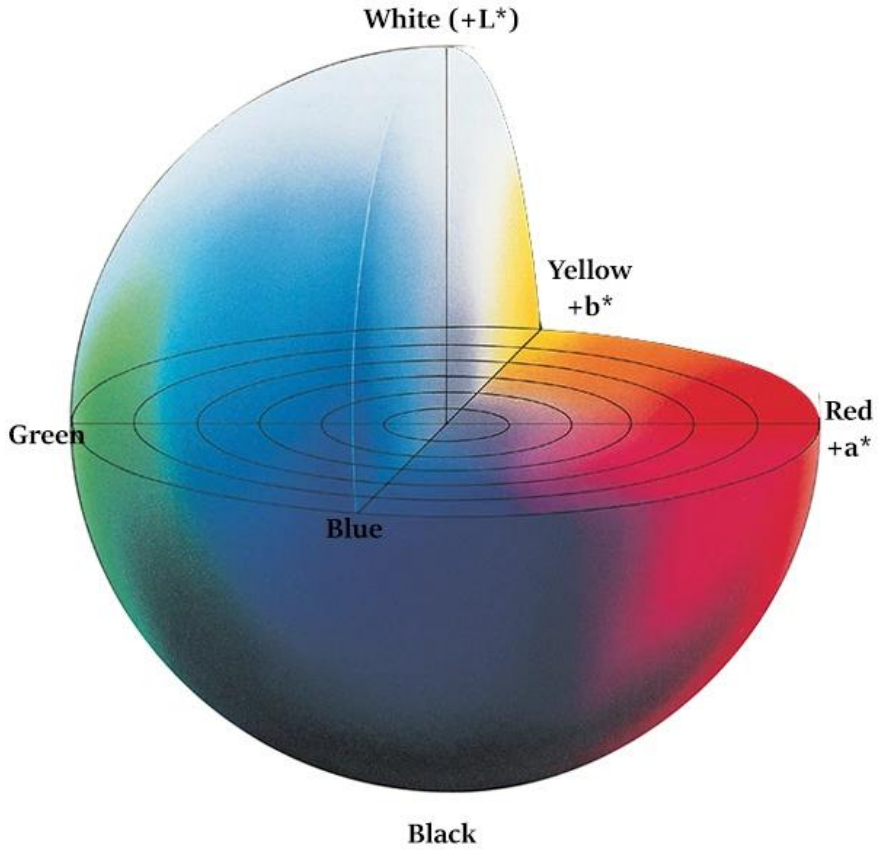



```python
import cv2
import numpy as np
def nothing(x):
    pass
img = np.zeros((200, 800, 3), np.uint8)
cv2.namedWindow('HSV Sketchbook')
cv2.createTrackbar('H', 'HSV Sketchbook', 0, 179, nothing)  # Hue: 0~179 (OpenCV range)
cv2.createTrackbar('S', 'HSV Sketchbook', 0, 255, nothing)  # Saturation: 0~255
cv2.createTrackbar('V', 'HSV Sketchbook', 0, 255, nothing)  # Value: 0~255

while True:
    # Get trackbar values
    h = cv2.getTrackbarPos('H', 'HSV Sketchbook')
    s = cv2.getTrackbarPos('S', 'HSV Sketchbook')
    v = cv2.getTrackbarPos('V', 'HSV Sketchbook')

    # Convert HSV to BGR (OpenCV uses BGR)
    hsv_color = np.uint8([[[h, s, v]]])
    b, g, r = cv2.cvtColor(hsv_color, cv2.COLOR_HSV2BGR)[0][0]

    # Apply background color
    img[:] = [b, g, r]

    # Prepare text
    text_hsv =
    text_rgb =

    # Draw text (black border + white text)
    cv2.putText(img, text_hsv,
    cv2.putText(img, text_hsv,

    cv2.putText(img, text_rgb,
    cv2.putText(img, text_rgb,

    cv2.imshow('HSV Sketchbook', img)
    if cv2.waitKey(1) & 0xFF == 27:
        break

cv2.destroyAllWindows()
```

Ⅰ. Color Space

LAB Color Space

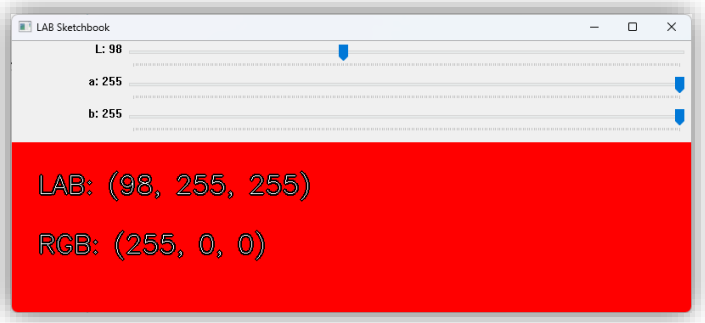
- 사람이 실제로 느끼는 색감에 더 가깝게 만든 색상 모델로 조명 변화에 강해서, 색 필터링, 색 추출, 색 비교 시 강점
- L (Lightness): 밝기 (0 = 어두움, 255 = 밝음)
- a: 초록색 ↔ 빨간색 (값이 작을수록 초록, 클수록 빨강)
- b: 파란색 ↔ 노란색 (값이 작을수록 파랑, 클수록 노랑)



Ⅰ. Color Space

LAB Color Space (cont.)

- Trackbar 를 사용해 L,A,B 값을 control 하여 변하는 LAB color 값을 확인하는 예제 코드



OpenCV 에서 기본적으로 사용하는 색공간은 BGR 이기 때문에 Display 출력을 위해 LAB 을 BGR 로 변경이 필요함

```python
import cv2
import numpy as np
def nothing(x):
    pass
img = np.zeros((200, 800, 3), np.uint8) # Create a blank image (sketchbook)
# Create LAB Sketchbook window and add trackbars
cv2.namedWindow('LAB Sketchbook')
cv2.createTrackbar('L', 'LAB Sketchbook', 0, 255, nothing) # Lightness: 0~255
cv2.createTrackbar('a', 'LAB Sketchbook', 0, 255, nothing) # a: green-red, center 128
cv2.createTrackbar('b', 'LAB Sketchbook', 0, 255, nothing) # b: blue-yellow, center 128

while True:
    l = cv2.getTrackbarPos('L', 'LAB Sketchbook') # Get current trackbar position for L
    a = cv2.getTrackbarPos('a', 'LAB Sketchbook') # Get current trackbar position for a
    b = cv2.getTrackbarPos('b', 'LAB Sketchbook') # Get current trackbar position for b

    lab_color = np.uint8([[[l, a, b]]]) # Convert LAB to BGR (OpenCV uses BGR)
    rgb_b, rgb_g, rgb_r = cv2.cvtColor(lab_color, cv2.COLOR_LAB2BGR)[0][0]

    text_lab = f"LAB: ({l}, {a}, {b})" # Prepare text for LAB
    text_rgb = f"RGB: ({rgb_r}, {rgb_g}, {rgb_b})" # Prepare text for RGB

    # Draw text (black border + white text)
    cv2.putText(img, text_lab, (30, 60), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 0), 3)
    cv2.putText(img, text_lab, (30, 60), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 1)
    cv2.putText(img, text_rgb, (30, 130), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 0), 3)
    cv2.putText(img, text_rgb, (30, 130), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 1)

    img[:] = [rgb_b, rgb_g, rgb_r] # Apply background color

    cv2.imshow('LAB Sketchbook', img) # Show image

    # Exit on ESC key
    if cv2.waitKey(1) & 0xFF == 27:
        break

cv2.destroyAllWindows()
```

Ⅰ. Color Space

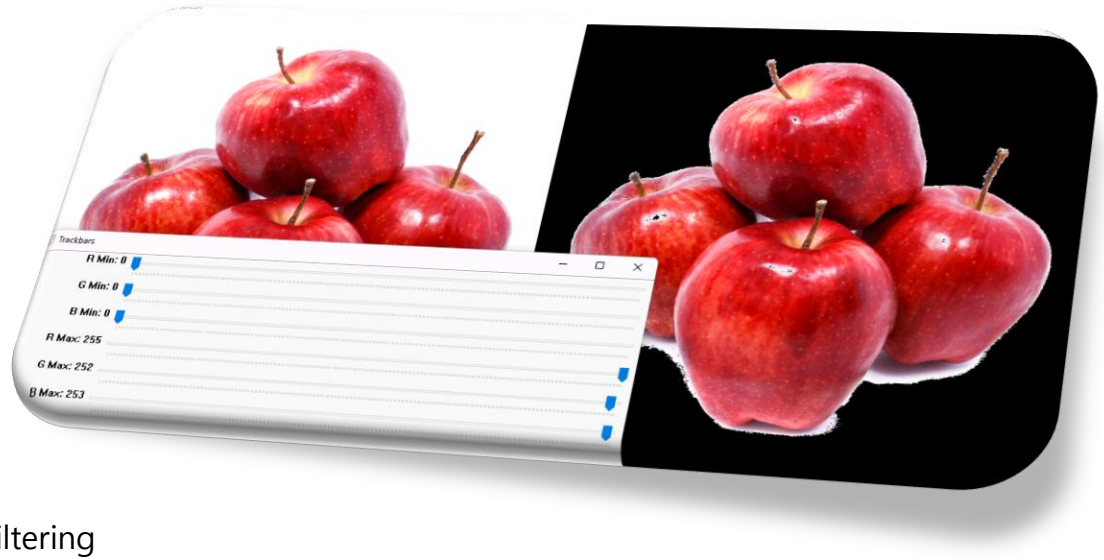
https://colorizer.org/

RGB vs HSV vs LAB

항목	RGB	HSV	LAB
구성	Red, Green, Blue	Hue, Saturation, Value	Lightness, a (green-red), b (blue-yellow)
범위 (OpenCV)	0~255 (각 채널)	H: 0179, S/V: 0255	L: 0255, a/b: 0255 (중심 128)
시각적 직관성	낮음	높음 (색 분리 쉬움)	높음 (사람 눈 기준)
조명 영향	높음	중간	낮음 (조명 보정에 강함)
용도	이미지 저장/표현	색 추적, 객체 인식	색 보정, 색 차이 분석 (ΔE), 밝기 분리 작업

Contents

Ⅱ. Color Filter 구현하기



RGB Color Filter 구현하기

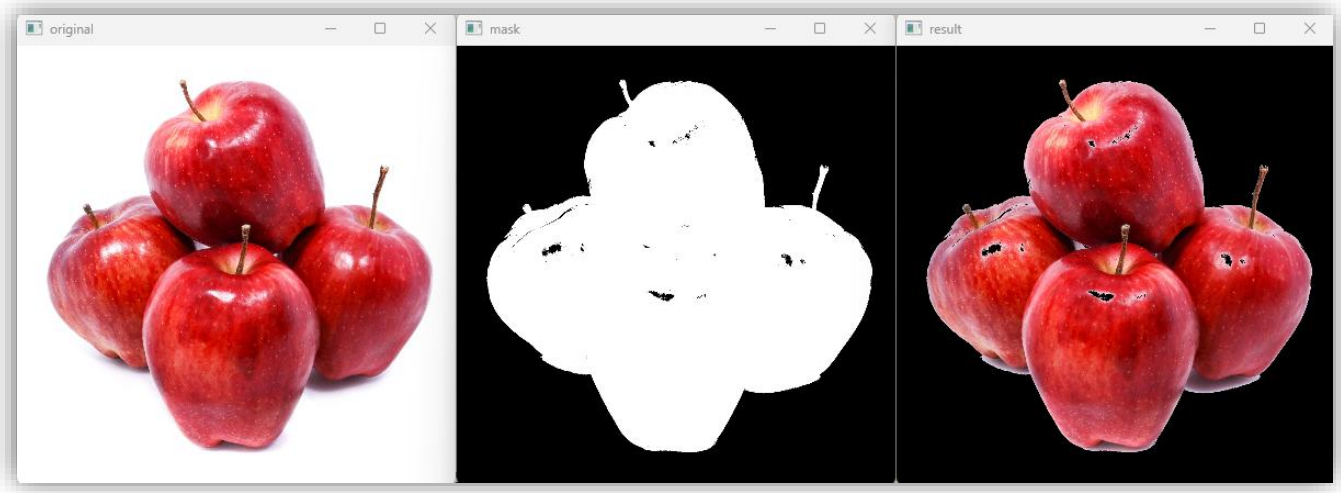
1. 각 color 별 min / max 값을 tuning 해서 filtering 할 값의 범위를 정하기

```
r_min, r_max, g_min, g_max, b_min, b_max
```

2. 예를 들면 r_min = 200, r_max = 220 이면 R 색상 값이 200 ~ 220 사이인 값만 filtering

3. OpenCV 의 TrackBar 를 사용해 실시간 threshold 값을 tuning 하도록 만들기

4. Filtering 하는 mask 를 생성해 불필요한 영역 제거



Ⅱ. Color Filter 구현하기

RGB Color Filter 구현하기

A	B	output
0	0	0
0	1	0
1	0	0
1	1	1

Mask
- 검은색: 0
- 흰색: 1

cv2.inRange 함수를 사용해 lower 와 upper 사이에 있는 image 값들의 mask 를 생성

```python
import cv2
import numpy as np

def nothing(x): pass
# Trackbar UI
cv2.namedWindow("Trackbars")
cv2.resizeWindow("Trackbars", 800, 260)
cv2.createTrackbar("R Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("R Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("G Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("G Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("B Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("B Max", "Trackbars", 255, 255, nothing)

img = cv2.imread("apple.webp") # image file load
while True:
    # Trackbar values
    r_min = cv2.getTrackbarPos("R Min", "Trackbars")
    r_max = cv2.getTrackbarPos("R Max", "Trackbars")
    g_min = cv2.getTrackbarPos("G Min", "Trackbars")
    g_max = cv2.getTrackbarPos("G Max", "Trackbars")
    b_min = cv2.getTrackbarPos("B Min", "Trackbars")
    b_max = cv2.getTrackbarPos("B Max", "Trackbars")
    # Filter in BGR space (OpenCV uses BGR)
    lower = np.array([b_min, g_min, r_min])
    upper = np.array([b_max, g_max, r_max])
    mask = cv2.inRange(img, lower, upper)
    result = cv2.bitwise_and(img, img, mask=mask)
    # Show filtered result
    combined = np.hstack((img, result))
    cv2.imshow("RGB Filter Result", cv2.resize(combined, (1280, 600)))
    key = cv2.waitKey(33)
    if key == 27: # 'ESC' to exit
        break

cv2.destroyAllWindows()
```

Ⅱ. Color Filter 구현하기

HSV Color Filter 구현하기

```python
import cv2
import numpy as np

def nothing(x): pass

cv2.namedWindow("Trackbars")
cv2.resizeWindow("Trackbars", 800, 260)
cv2.createTrackbar("H Min", "Trackbars", 0, 179, nothing)
cv2.createTrackbar("H Max", "Trackbars", 179, 179, nothing)
cv2.createTrackbar("S Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("S Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("V Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("V Max", "Trackbars", 255, 255, nothing)

img = cv2.imread("apple.png")

while True:
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

    h_min = cv2.getTrackbarPos("H Min", "Trackbars")
    h_max = cv2.getTrackbarPos("H Max", "Trackbars")
    s_min = cv2.getTrackbarPos("S Min", "Trackbars")
    s_max = cv2.getTrackbarPos("S Max", "Trackbars")
    v_min = cv2.getTrackbarPos("V Min", "Trackbars")
    v_max = cv2.getTrackbarPos("V Max", "Trackbars")

    lower = np.array([h_min, s_min, v_min])
    upper = np.array([h_max, s_max, v_max])
    mask = cv2.inRange(hsv, lower, upper)
    result = cv2.bitwise_and(img, img, mask=mask)

    combined = np.hstack((img, result))
    cv2.imshow("HSV Filter & Hue Visualization", cv2.resize(combined, (1280, 600)))

    key = cv2.waitKey(33)
    if key == 27: # ESE to exit
        break

cv2.destroyAllWindows()
```

Ⅱ. Color Filter 구현하기

HSV Color Filter 구현하기 (cont.)

- Input 을 이미지 파일 말고 Camera Input 으로 수정하기
- 사람의 피부만 filtering 되도록 파라미터 값 변경해 보기

```
import cv2
import numpy as np

def nothing(x): pass

cv2.namedWindow("Trackbars")
cv2.resizeWindow("Trackbars", 800, 260)
cv2.createTrackbar("H Min", "Trackbars", 0, 179, nothing)
cv2.createTrackbar("H Max", "Trackbars", 179, 179, nothing)
cv2.createTrackbar("S Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("S Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("V Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("V Max", "Trackbars", 255, 255, nothing)

cap =

while True:
    ret, frame =
    if not ret:
        break

    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    h_min = cv2.getTrackbarPos("H Min", "Trackbars")
    h_max = cv2.getTrackbarPos("H Max", "Trackbars")
    s_min = cv2.getTrackbarPos("S Min", "Trackbars")
    s_max = cv2.getTrackbarPos("S Max", "Trackbars")
    v_min = cv2.getTrackbarPos("V Min", "Trackbars")
    v_max = cv2.getTrackbarPos("V Max", "Trackbars")
    lower = np.array([h_min, s_min, v_min])
    upper = np.array([h_max, s_max, v_max])
    mask = cv2.inRange(hsv, lower, upper)
    result = cv2.bitwise_and(frame, frame, mask=mask)
    combined = np.hstack((frame, result))
    cv2.imshow("HSV Filter & Hue Visualization", cv2.resize(combined, (1280, 600)))

    key = cv2.waitKey(1)
    if key == 27: # ESE to exit
        break
cap.release()
cv2.destroyAllWindows()
```

Ⅱ. Color Filter 구현하기

LAB Color Filter 구현하기

```python
import cv2
import numpy as np

def nothing(x): pass

cv2.namedWindow("Trackbars")
cv2.resizeWindow("Trackbars", 800, 260)
cv2.createTrackbar("L Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("L Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("A Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("A Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("B Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("B Max", "Trackbars", 255, 255, nothing)

img = cv2.imread("apple.png")

while True:
    lab = cv2.cvtColor(img, cv2.COLOR_BGR2LAB)

    l_min = cv2.getTrackbarPos("L Min", "Trackbars")
    l_max = cv2.getTrackbarPos("L Max", "Trackbars")
    a_min = cv2.getTrackbarPos("A Min", "Trackbars")
    a_max = cv2.getTrackbarPos("A Max", "Trackbars")
    b_min = cv2.getTrackbarPos("B Min", "Trackbars")
    b_max = cv2.getTrackbarPos("B Max", "Trackbars")

    lower = np.array([l_min, a_min, b_min])
    upper = np.array([l_max, a_max, b_max])
    mask = cv2.inRange(lab, lower, upper)
    result = cv2.bitwise_and(img, img, mask=mask)

    combined = np.hstack((img, result))
    cv2.imshow("LAB Filter Result", cv2.resize(combined, (1280, 600)))

    if cv2.waitKey(33) & 0xFF == 27:  # ESC key
        break

cv2.destroyAllWindows()
```

Ⅱ. Color Filter 구현하기

LAB Color Filter 구현하기

- Input 을 이미지 파일 말고 Camera Input 으로 수정하기
- 사람의 피부만 filtering 되도록 파라미터 값 변경해 보기

```python
import cv2
import numpy as np

def nothing(x): pass
cv2.namedWindow("Trackbars")
cv2.resizeWindow("Trackbars", 800, 260)
cv2.createTrackbar("L Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("L Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("A Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("A Max", "Trackbars", 255, 255, nothing)
cv2.createTrackbar("B Min", "Trackbars", 0, 255, nothing)
cv2.createTrackbar("B Max", "Trackbars", 255, 255, nothing)

cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    if not ret:
        break
    lab = cv2.cvtColor(img, cv2.COLOR_BGR2LAB)

    l_min = cv2.getTrackbarPos("L Min", "Trackbars")
    l_max = cv2.getTrackbarPos("L Max", "Trackbars")
    a_min = cv2.getTrackbarPos("A Min", "Trackbars")
    a_max = cv2.getTrackbarPos("A Max", "Trackbars")
    b_min = cv2.getTrackbarPos("B Min", "Trackbars")
    b_max = cv2.getTrackbarPos("B Max", "Trackbars")

    lower = np.array([l_min, a_min, b_min])
    upper = np.array([l_max, a_max, b_max])
    mask = cv2.inRange(lab, lower, upper)
    result = cv2.bitwise_and(img, img, mask=mask)

    combined = np.hstack((frame, result))
    cv2.imshow("LAB Filter Result ", cv2.resize(combined, (1280, 600)))

    key = cv2.waitKey(33)
    if key == 27: # ESE to exit
        break

cap.release()
cv2.destroyAllWindows()
```

Ⅱ. Color Filter 구현하기

https://www.youtube.com/watch?v=3IJ8RFtlDLY
https://www.youtube.com/watch?v=b5lgnNEzGeU

How to improve filtering quality?

- Morphological Operations: 이미지 속의 작은 오염(노이즈)을 제거하거나 객체를 더 명확하게 만드는 도구
- 침식 (Erosion): 흰색 영역이 조금씩 줄어드는 효과로 커널(작은 사각형) 내에 모두 흰색이어야 흰색 유지, 아니면 검정
- 팽창 (Dilation): 흰색 영역이 커지는 효과로 커널 내 하나라도 흰색이면 흰색 유지
- Open: 침식 후 팽창 (노이즈 제거)
- Close: 팽창 후 침식 (구멍 메움)

```
... 생략 ...

    l_min = cv2.getTrackbarPos("L Min", "Trackbars")
    l_max = cv2.getTrackbarPos("L Max", "Trackbars")
    a_min = cv2.getTrackbarPos("A Min", "Trackbars")
    a_max = cv2.getTrackbarPos("A Max", "Trackbars")
    b_min = cv2.getTrackbarPos("B Min", "Trackbars")
    b_max = cv2.getTrackbarPos("B Max", "Trackbars")

    lower = np.array([l_min, a_min, b_min])
    upper = np.array([l_max, a_max, b_max])
    mask = cv2.inRange(lab, lower, upper)
    # Without noise reduction
    result = cv2.bitwise_and(frame, frame, mask=mask)

    # Create 5x5 kernel for Morphological Operations
    kernel = np.ones((5,5), np.uint8)

    # Open Morphological Operations
    mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)
    result_open = cv2.bitwise_and(frame, frame, mask=mask)

    # Close Morphological Operations
    mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
    result_close = cv2.bitwise_and(frame, frame, mask=mask)

    cv2.imshow("LAB Filter Result", cv2.resize(result, (640, 480)))
    cv2.imshow("LAB Filter Result with open", cv2.resize(result_open, (640, 480)))
    cv2.imshow("LAB Filter Result with close", cv2.resize(result_close, (640, 480)))

... 생략 ...
```

Ⅱ. Color Filter 구현하기

How to improve filtering quality?

- Median Blur: 사진에서 'Noise' 를 없애주는 똑똑한 도구로 각 픽셀 주변의 커널 내 픽셀 값들의 중간 값으로 픽셀을 대체하는 비선형 필터
- 경계선을 어느 정도 보존하면서 노이즈만 줄여 주게 됨
- ksize 는 커널 크기로 주변 픽셀 중간값으로 교체하는 범위이며 반드시 홀수로 사용되어야 함 (3, 5, 7, 9 등)

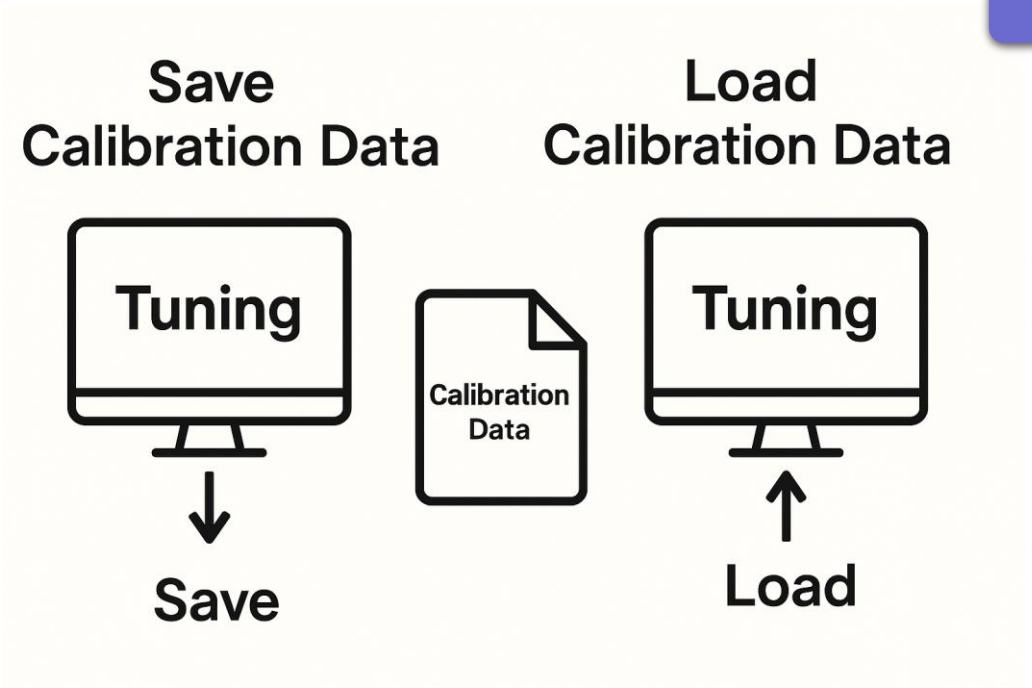

```
... 생략 ...

    l_min = cv2.getTrackbarPos("L Min", "Trackbars")
    l_max = cv2.getTrackbarPos("L Max", "Trackbars")
    a_min = cv2.getTrackbarPos("A Min", "Trackbars")
    a_max = cv2.getTrackbarPos("A Max", "Trackbars")
    b_min = cv2.getTrackbarPos("B Min", "Trackbars")
    b_max = cv2.getTrackbarPos("B Max", "Trackbars")

    lower = np.array([l_min, a_min, b_min])
    upper = np.array([l_max, a_max, b_max])
    mask = cv2.inRange(lab, lower, upper)
    # Without noise reduction
    result = cv2.bitwise_and(frame, frame, mask=mask)

    # Median Blur with 5x5 kernel size for noise reduction
    mask_blurred = cv2.medianBlur(mask, ksize=5)
    result_blur = cv2.bitwise_and(frame, frame, mask=mask_blurred)

    cv2.imshow("LAB Filter Result", cv2.resize(result, (640, 480)))
    cv2.imshow("LAB Filter Result with blur", cv2.resize(result_blur, (640, 480)))

... 생략 ...
```

Contents

Ⅲ . Tuning 한 Threshold 값 Save / Load

Tuning (or Calibration) 한 값들을 파일로 저장하고 사용할 때 이 파일을 로드한다면?

- 게임을 할 때도 플레이 한 내용을 Save 하고 다시 플레이 할 때 Save 한 내용을 Load 해서 사용
- Tuning Tool 에서는 반드시 tuning 한 내용을 file 로 저장 후 해당 내용을 load 해서 사용이 필요함
- 그렇다면 어떤 형식으로 tuning 한 내용을 file 로 저장할까?



JSON | YAML | INI | CSV | SQLITE

보안이슈? 암호화 후 저장

Ⅲ . Tuning 한 Threshold 값 Save / Load

JSON 파일로 Save / Load

- JSON(JavaScript Object Notation)은 데이터 교환을 위한 경량 텍스트 기반 포맷
- 웹 서버와 클라이언트 간 데이터 전송, 설정파일 등에서 빠르게 표준으로 자리잡음, 주석은 사용 불가
- 기본 Syntax
  - JSON 객체는 중괄호 `{ }` 둘러쌓아 표현
  - JSON 객체는 쉼표 `,` 를 사용하여 여러 프로퍼티를 포함
  - JSON 배열은 대괄호 `[ ]` 로 둘러쌓아 표현
  - 제공하는 type: 숫자, 문자열, Boolean, 객체, 배열, null

example.json

```json
{
  "name": "Alice",
  "age": 27,
  "is_student": false,
  "hobbies": ["reading", "music", "badminton"],
  "info": {
    "height": 170,
    "weight": 62
  },
  "friends": null
}
```

Python 에서 load

```python
import json

with open("example.json", "r") as f:
    data = json.load(f)

# Load “example.json” and use the results:
print(data["name"])           # 'Alice'
print(data["age"])            # 27
print(data["is_student"])     # False
print(data["hobbies"])        # ['reading', 'music', 'badminton']
print(data["info"]["height"]) # 170
print(data["info"]["weight"]) # 62
print(data["friends"])        # None (null)
```

Ⅲ . Tuning 한 Threshold 값 Save / Load

JSON 파일로 저장하기

```
json.dump(obj, fp, indent=None)
```

- Obj: 저장할 파이썬 객체 (dict, list 등)
- fp: 파일 객체 (write 모드로 open된 파일)
- Indent: 예쁘게 포맷(들여쓰기)하려면 정수값 (ex. 4)

JSON 파일 로드하기

```
json.load(fp)
```

- fp: 파일 객체 (write 모드로 open된 파일)

```python
import json

# 1. JSON 파일 읽기
with open("lab_config.json", "r") as f:
    config = json.load(f)

# 2. 각각의 값을 변수에 직접 할당
l_min = config["l_min"]
l_max = config["l_max"]
a_min = config["a_min"]
a_max = config["a_max"]
b_min = config["b_min"]
b_max = config["b_max"]

print("L 범위:", l_min, "~", l_max)
print("A 범위:", a_min, "~", a_max)
print("B 범위:", b_min, "~", b_max)

... Skip ...

# 3. 변수들을 다시 딕셔너리로 묶어서 저장
save_data = {
    "l_min": l_min,
    "l_max": l_max,
    "a_min": a_min,
    "a_max": a_max,
    "b_min": b_min,
    "b_max": b_max
}

with open("lab_config_saved.json", "w") as f:
    json.dump(save_data, f, indent=4)
```

Ⅲ . Tuning 한 Threshold 값 Save / Load

LAB Color Filter 를 구현하기

Requirements:

- 카메라 Input 사용
- 's' 키를 누르면 tuning 한 내용이 json 포멧으로 저장
- 2가지 이름으로 파일이 저장되며 하나는 LAB-cal.json 이 overwrite 되고, 다른 파일 이름은 LAB-cal-YYMMDD-HHMMSS.json 형태 (예: 250601, 오전 11시 10분 12초 → LAB-cal-250601-111012.json)
- LAB Color Filter 가 실행될 때 LAB-cal.json 파일을 load 하여 값 초기화

THANK YOU