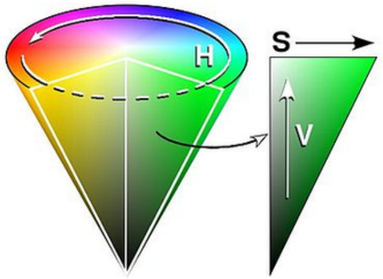
1차 업무

1) 빨간색 영역을 감지하고 필터링하는 예제 코드

RGB → HSV 변환

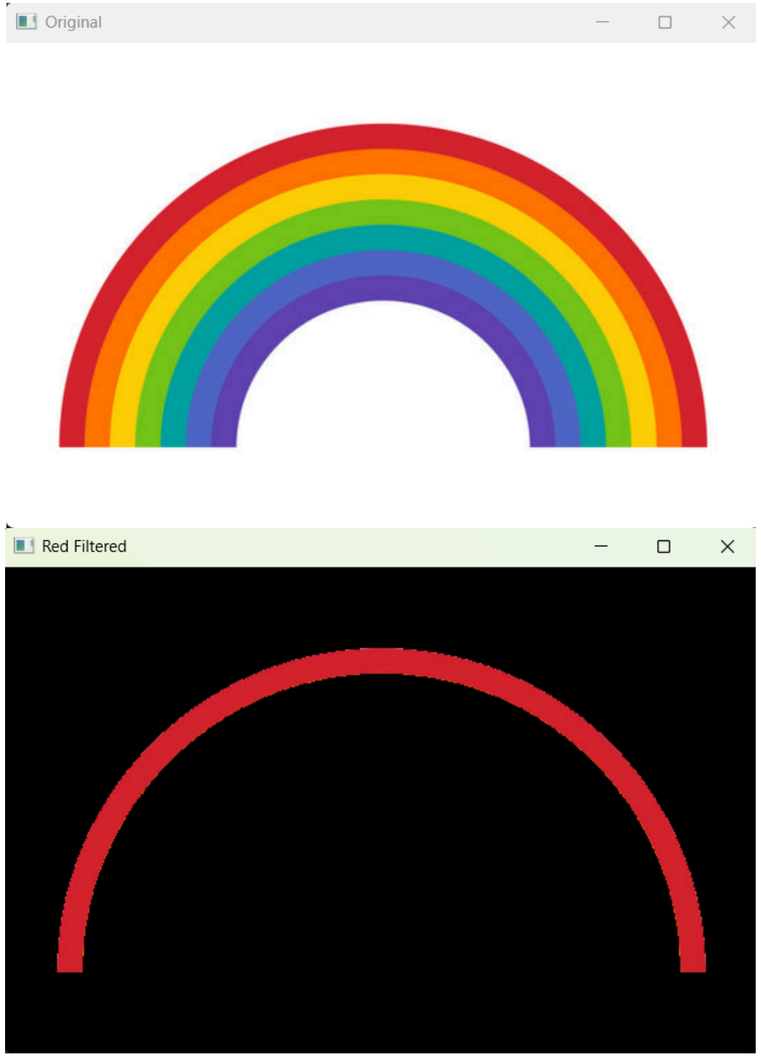


H : 색상, 색의 종류

S : 채도, 색의 탁하고 선명한 정도

V : 명도, 빛의 밝기.

빨간색 범위 지정 → 마스크 생성



week1.py > ...

```python
import cv2
import numpy as np
# 이미지 로드

# from datasets import load_dataset

# ds = load_dataset("ethz/food101")


image = cv2.imread('sample.jpg') # 분석할 이미지 파일

# BGR에서 HSV 색상 공간으로 변환
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# 빨간색 범위 지정 (두 개의 범위를 설정해야 함)
lower_red1 = np.array([0, 120, 70])
upper_red1 = np.array([10, 255, 255])
lower_red2 = np.array([170, 120, 70])
upper_red2 = np.array([180, 255, 255])

# 마스크 생성
mask1 = cv2.inRange(hsv, lower_red1, upper_red1)
mask2 = cv2.inRange(hsv, lower_red2, upper_red2)

mask = mask1 + mask2 # 두 개의 마스크를 합침

# 원본 이미지에서 빨간색 부분만 추출
result = cv2.bitwise_and(image, image, mask=mask)

# 결과 이미지 출력
cv2.imshow('Original', image)
cv2.imshow('Red Filtered', result)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

1차 업무

2) 트랙바에서 실시간 색상 뽑기

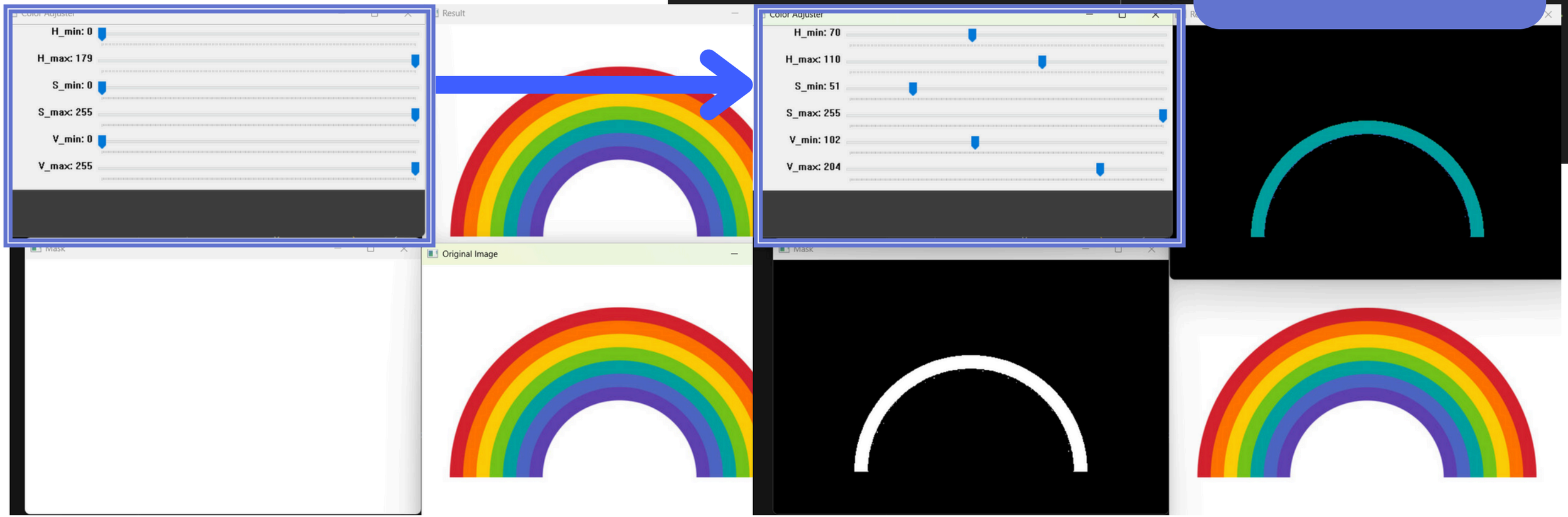
다른 색상을 뽑을 수는 없을까?

```python
### 2) 트랙바에서 실시간 색상 뽑기

import cv2
import numpy as np

def nothing(x):
    pass

# 이미지 로드 (본인의 이미지 경로로 변경하세요)
image = cv2.imread('sample.jpg')

# 이미지가 제대로 로드되었는지 확인
if image is None:
    print("이미지를 로드할 수 없습니다. 파일 경로를 확인해주세요.")
    exit()

# BGR에서 HSV 색상 공간으로 변환
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# 트랙바를 위한 윈도우 생성
cv2.namedWindow('Color Adjuster')
cv2.resizeWindow('Color Adjuster', 600, 300) # 윈도우 크기 조절
```

```python
while True:
    # 트랙바에서 현재 값 가져오기
    h_min = cv2.getTrackbarPos('H_min', 'Color Adjuster')
    h_max = cv2.getTrackbarPos('H_max', 'Color Adjuster')
    s_min = cv2.getTrackbarPos('S_min', 'Color Adjuster')
    s_max = cv2.getTrackbarPos('S_max', 'Color Adjuster')
    v_min = cv2.getTrackbarPos('V_min', 'Color Adjuster')
    v_max = cv2.getTrackbarPos('V_max', 'Color Adjuster')

    # HSV 하한 및 상한 배열 생성
    lower_bound = np.array([h_min, s_min, v_min])
    upper_bound = np.array([h_max, s_max, v_max])

    # 범위에 해당하는 마스크 생성
    mask = cv2.inRange(hsv, lower_bound, upper_bound)

    # 원본 이미지에 마스크 적용
    result

    # 결과
    cv2.ims
```

트랙바 조정

원하는 색상 조정 가능



1차 업무

3) 원본 이미지에서 실시간으로 색상 뽑기

더 쉽게 색을 뽑을 수는 없을까?

클릭하는 색상이 나오게 한다


```python
### 3) 원본 이미지에서 실시간으로 색상 뽑기

import cv2
import numpy as np

# 선택된 색상의 HSV 값을 저장할 변수
selected_hsv = [None, None, None]

# HSV 범위 설정을 위한 오프셋 정의 (원하는 만큼 자유롭게 변경)
# 좀 더 비슷한 주변 색상들까지 포함시킴!
HUE_OFFSET = 5
SAT_OFFSET = 30
VAL_OFFSET = 30

# 마우스 이벤트 콜백 함수
def mouse_callback(event, x, y, flags, param):
    global selected_hsv, image_hsv

    # 왼쪽 마우스 버튼을 클릭했을 때
    if event == cv2.EVENT_LBUTTONDOWN:
        # 클릭한 지점의 HSV 값 가져오기
        pixel_hsv = image_hsv[y, x]
        selected_hsv = pixel_hsv.tolist() # NumPy 배열을 리스트로 변환
        print(f"클릭한 픽셀의 HSV 값: {selected_hsv}")

# 이미지 로드
image = cv2.imread('sample.jpg')

# 이미지가 제대로 로드되었는지 확인
if image is None:
    print("이미지를 로드할 수 없습니다. 파일 경로를 확인해주세요.")
    exit()

# BGR -> HSV 색상 공간 변환
image_hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

# 이미지 창 생성 및 마우스 콜백 함수 연결
cv2.namedWindow('Image')
cv2.setMouseCallback('Image', mouse_callback)

print("이미지에서 완전 똑같은 색상을 추출하고 싶은 픽셀을 클릭하세요.")
print("'q' 키를 누르면 종료됩니다.")

while True:
    # 선택된 HSV 값이 있을 경우에만 마스크 생성
    if selected_hsv[0] is not None:
        # 선택된 HSV 값을 기준으로 범위 설정 (오프셋 적용)
        h_min = selected_hsv[0] - HUE_OFFSET
        h_max = selected_hsv[0] + HUE_OFFSET
        s_min = selected_hsv[1] - SAT_OFFSET
        s_max = selected_hsv[1] + SAT_OFFSET
        v_min = selected_hsv[2] - VAL_OFFSET
        v_max = selected_hsv[2] + VAL_OFFSET

        lower_bound = np.array([h_min, s_min, v_min])
        upper_bound = np.array([h_max, s_max, v_max])

        mask = cv2.inRange(image_hsv, lower_bound, upper_bound)

        result = cv2.bitwise_and(image, image, mask=mask)
        cv2.imshow('Result', result)
        cv2.imshow('Mask', mask)

    cv2.imshow('Image', image) # 원본 이미지 계속 표시

    # 'q' 키를 누르면 종료
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cv2.destroyAllWindows()
```