RGB to HSV 코드

| import numpy as np  import cv2  R = 0; G = 255; B = 0  my\_color = np.uint8([[[B,G,R]]]) # BGR  my\_hsv = cv2.cvtColor(my\_color,cv2.COLOR\_BGR2HSV)  print(f"H:{my\_hsv[0][0][0]} S:{my\_hsv[0][0][1]} V:{my\_hsv[0][0][2]}") |
| --- |

토마토 영상을 HSV로 변환하여 각 채널 출력

| import cv2  img\_src = cv2.imread('images/tomato.jpg', cv2.IMREAD\_COLOR)  # 이미지를 BGR에서 HSV로 색변환  img\_hsv = cv2.cvtColor(img\_src, cv2.COLOR\_BGR2HSV) # HSV  img\_h, img\_s, img\_v = cv2.split(img\_hsv) # H,S,V로 채널 분리  cv2.imshow('src',img\_src)  cv2.imshow('HSV-h',img\_h) # HSV에서 H채널 출력  cv2.imshow('HSV-s',img\_s) # HSV에서 S채널 출력  cv2.imshow('HSV-v',img\_v) # HSV에서 V채널 출력  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

토마토에서 초록색 추출 - Hue(H)만 사용

| import cv2  img\_src = cv2.imread('images/tomato.jpg', cv2.IMREAD\_COLOR)  # 이미지를 BGR에서 HSV로 색변환  img\_hsv = cv2.cvtColor(img\_src, cv2.COLOR\_BGR2HSV) # HSV  img\_h, img\_s, img\_v = cv2.split(img\_hsv) # H,S,V로 채널 분리  # 원하는 color의 h정보를 적용 : 초록색을 추출 ( 90도~150도(대략) )  # Opencv에서는 0~180까지만 사용 초록정보를 나누기 2함 (45~75 (대략))  mask\_h = cv2.inRange(img\_h, 46, 75) # 초록 정보 추출  img\_dst = cv2.bitwise\_and(img\_src, img\_src, mask=mask\_h)  cv2.imshow('green',img\_dst)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

토마토에서 초록색 추출 2 : HSV 전체를 마스크로 사용

| import cv2  img\_src = cv2.imread('images/tomato.jpg', cv2.IMREAD\_COLOR)  # 이미지를 BGR에서 HSV로 색변환  img\_hsv = cv2.cvtColor(img\_src, cv2.COLOR\_BGR2HSV) # HSV  # HSV로 초록색 정보를 좀 더 구체적으로 표시  lower\_green = (50,65,0) # 자료형은 튜플형태로(H,S,V)  upper\_green = (75,255,255) # 자료형은 튜플형태로(H,S,V)  img\_mask = cv2.inRange(img\_hsv, lower\_green, upper\_green) # 초록 정보 추출  img\_dst = cv2.bitwise\_and(img\_src, img\_src, mask=img\_mask)  cv2.imshow('green',img\_dst)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

RED의 경우 Hue 영역이 떨어져서 존재하므로 두영역을 찾고 합침

| import cv2  img\_src = cv2.imread('images/tomato.jpg', cv2.IMREAD\_COLOR)  # 이미지를 BGR에서 HSV로 색변환  img\_hsv = cv2.cvtColor(img\_src, cv2.COLOR\_BGR2HSV) # HSV  # HSV에서 RED는 2개의 영역이 존재  s\_min = 50; s\_max = 255  v\_min = 50; v\_max = 255  lower\_red1 = (0, s\_min, v\_min) # 자료형은 튜플형태로(H,S,V)  upper\_red1 = (7, s\_max, v\_max) # 자료형은 튜플형태로(H,S,V)  lower\_red2 = (165, s\_min, v\_min) # 자료형은 튜플형태로(H,S,V)  upper\_red2 = (180, s\_max, v\_max) # 자료형은 튜플형태로(H,S,V)  img\_mask1 = cv2.inRange(img\_hsv, lower\_red1, upper\_red1) # 빨강정보1(0~7) 추출  img\_mask2 = cv2.inRange(img\_hsv, lower\_red2, upper\_red2) # 빨강정보2(165~180) 추출  img\_mask = cv2.addWeighted(img\_mask1, 1.0, img\_mask2, 1.0, 0.0)  img\_dst = cv2.bitwise\_and(img\_src, img\_src, mask=img\_mask)  cv2.imshow('dst',img\_dst)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

## IMAGE FLIP

이미지 대칭 사용

| # 이미지 대칭  import cv2  img\_src = cv2.imread('images/animal-03.jpg',cv2.IMREAD\_COLOR)  img\_flip\_lr = cv2.flip(img\_src,1) # 1:좌우대칭 / 0:상하대칭 / -1:상하좌우대칭  img\_dst1 = cv2.hconcat([img\_src,img\_flip\_lr])  img\_flip\_ud = cv2.flip(img\_src,0) # 1:좌우대칭 / 0:상하대칭 / -1:상하좌우대칭  img\_flip\_lrud = cv2.flip(img\_src,-1) # 1:좌우대칭 / 0:상하대칭 / -1:상하좌우대칭  img\_dst2 = cv2.hconcat([img\_flip\_ud,img\_flip\_lrud])  img\_dst = cv2.vconcat([img\_dst1,img\_dst2])  cv2.imshow('dst',img\_dst)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

이미지 회전

| import cv2  img\_src = cv2.imread('images/animal-05.jpg', cv2.IMREAD\_COLOR)  height, width = img\_src.shape[:2]  center = (width/2, height/2) # 회전할 이미지의 중심점  angle = 45 # 회전할 각도  scale = 1 # scale  # 회전  matrix = cv2.getRotationMatrix2D(center, angle, scale)  img\_dst = cv2.warpAffine(img\_src, matrix, (width,height))  cv2.imshow('src',img\_dst)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

이미지 확대 축소 : pyrDown(), pyrUp(), resize()함수 사용

| import cv2  img\_src = cv2.imread('images/animal-01.jpg',cv2.IMREAD\_COLOR)  height, width = img\_src.shape[:2]  #이미지 축소  img\_pyrdown = cv2.pyrDown(img\_src) #이미지를 가로세로 절반으로 줄임  #이미지 확대 : 2배만 사용가능  img\_pyrup = cv2.pyrUp(img\_src,dstsize=(width\*2, height\*2),  borderType=cv2.BORDER\_DEFAULT)  #이미지 크기조절 (resize)  # dsize=(원하는 가로(w)크기, 원하는 세로(h) 크기)  img\_dst1 = cv2.resize(img\_src,dsize=(int(width\*0.7),int(height\*0.7)),  interpolation=cv2.INTER\_LINEAR)  # dsize부분은 None, fx=0.5, fy=0.5형식의 배율로 지정  img\_dst2 = cv2.resize(img\_src, None, fx=0.7, fy=0.7,  interpolation=cv2.INTER\_LINEAR)  cv2.imshow('resize1',img\_dst2)  cv2.imshow('resize2',img\_dst2)  cv2.imshow('down',img\_pyrdown)  cv2.imshow('up',img\_pyrup)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

이미지를 잘라서 수정한 후 원영상에 붙이기

| #이미지 자르기  import cv2  img\_src = cv2.imread('images/pawns.jpg', cv2.IMREAD\_COLOR)  # img\_src[h\_low:h\_high, w\_low:w\_high, :]  # img\_src[y\_low:y\_high, x\_low:x\_high, :] # y축을 먼저 적는다  img\_dst = img\_src[245:830, 45:310, :].copy() # 사본만들기  img\_dst = cv2.cvtColor(img\_dst, cv2.COLOR\_BGR2GRAY) # 그레이로변환  img\_dst = cv2.cvtColor(img\_dst, cv2.COLOR\_GRAY2BGR) # 3CH 그레이로변환  img\_src[245:830, 45:310, :] = img\_dst # 원본에 붙이기  cv2.imshow('dst',img\_src)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

이미지 3개 잘라서 수정하기

| import cv2  img\_src = cv2.imread('images/pawns.jpg', cv2.IMREAD\_COLOR)  # img\_src[h\_low:h\_high, w\_low:w\_high, :]  img\_dst = img\_src.copy()  img\_dst = cv2.cvtColor(img\_dst, cv2.COLOR\_BGR2GRAY)  img\_dst = cv2.cvtColor(img\_dst, cv2.COLOR\_GRAY2BGR) # 3채널 gray로 먼저 변환  # x,y좌표를 사용하여 이미지 자르기  pt1\_lower\_left = (50,240) ; pt1\_upper\_right = (310,825)  pt2\_lower\_left = (405,350) ; pt2\_upper\_right = (660,790)  pt3\_lower\_left = (770,440) ; pt3\_upper\_right = (970,790)  # 원본의 좌표에 3CH gray의 데이터를 넣기 img\_src와 img\_gray의 채널은 각각 3채널임’  img\_src[pt1\_lower\_left[1]:pt1\_upper\_right[1], pt1\_lower\_left[0]:pt1\_upper\_right[0], :] = \  img\_dst[pt1\_lower\_left[1]:pt1\_upper\_right[1], pt1\_lower\_left[0]:pt1\_upper\_right[0], :]  img\_src[pt2\_lower\_left[1]:pt2\_upper\_right[1], pt2\_lower\_left[0]:pt2\_upper\_right[0], :] = \  img\_dst[pt2\_lower\_left[1]:pt2\_upper\_right[1], pt2\_lower\_left[0]:pt2\_upper\_right[0], :]  img\_src[pt3\_lower\_left[1]:pt3\_upper\_right[1], pt3\_lower\_left[0]:pt3\_upper\_right[0], :] = \  img\_dst[pt3\_lower\_left[1]:pt3\_upper\_right[1], pt3\_lower\_left[0]:pt3\_upper\_right[0], :]  cv2.imshow('result', img\_src)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |

이미지 역상/ 반전 255->0 / 0->255로

| import cv2  img\_src = cv2.imread('images/animal-09.jpg',cv2.IMREAD\_COLOR)  img\_dst = cv2.bitwise\_not(img\_src)  img\_dst = cv2.hconcat([img\_src,img\_dst])  cv2.imshow('dst',img\_dst)  cv2.waitKey(0)  cv2.destroyAllWindows() |
| --- |