|  |  |
| --- | --- |
| 교육제목 | 데이터 기반 인공지능 시스템 엔지니어 양성 과정 |
| 교육일시 | 211110 |
| 교육장소 | YGL 학과장 및 자택(디스코드 사용 온라인 학습) |
| **교육내용** | |
| 1. Cv2 활용하여 웹캠 영상 저장(복습)   import numpy as np  import sys  import cv2  cap = cv2.VideoCapture(0)  if not cap.isOpened():  print('video open failed')  cap.release()  sys.exit()    cv2.namedWindow('webcam')  width = int(cap.get(cv2.CAP\_PROP\_FRAME\_WIDTH))  height = int(cap.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))  fps = int(cap.get(cv2.CAP\_PROP\_FPS))  fourcc = cv2.VideoWriter\_fourcc(\*'DIVX')  out = cv2.VideoWriter('파일이름', fourcc, fps, (width, height))  while True:    ret, frame = cap.read()    if not ret:  print('video read failed')  break  ########################  edge = cv2.Canny(frame, 50, 100)  edge = cv2.cvtColor(edge, cv2.COLOR\_GRAY2BGR)      ########################  out.write(edge)  cv2.imshow('webcam', frame)    if cv2.waitKey(10) == 27:  break    cap.release()  out.release()  cv2.destroyAllWindows()   1. 마우스 모션 입력받아서 영상에 라인 그리기   oldx = -1  oldy = -1  def call\_mouse(event, x, y,flags, param):  global oldx, oldy  if event == cv2.EVENT\_LBUTTONDOWN:  print('EVENT\_LBUTTONDOWN :', x,y)  oldx, oldy = x, y    elif event == cv2.EVENT\_MOUSEMOVE: # 움직일때  if flags == cv2.EVENT\_FLAG\_LBUTTON: # 하지만 왼클릭하고 움직이게 설정  cv2.line(img, (oldx, oldy), (x,y),(255, 0, 0), 6,  cv2.LINE\_AA)  cv2.imshow('img',img)  oldx, oldy = x, y  img = np.ones((400, 600, 3), dtype = np.uint8)\*255  cv2.namedWindow('img')  cv2.setMouseCallback('img', call\_mouse, img)  cv2.imshow('img', img)  cv2.waitKey()  cv2.destroyAllWindows()   1. 영상에 트랙바 삽입   def call\_track(pos):  print(pos)  img[:] = pos  cv2.imshow('img', img)    img = np.ones((400, 600, 3), dtype = np.uint8)  cv2.namedWindow('img')  cv2.createTrackbar('level', 'img', 5, 255, call\_track)  cv2.imshow('img', img)  cv2.waitKey()  cv2.destroyAllWindows()   1. 영상 밝게 만들기   src = cv2.imread('fig/lenna.bmp', cv2.IMREAD\_COLOR)  # dst = src + 100 이건 이상하게 나옴  dst = cv2.add(src, (100, 100, 100, 0)) #마지막 0은 알파채널  # dst = np.clip(src + 100. , 0, 255).astype(np.uint8)  dst = cv2.cvtColor(src, cv2.COLOR\_BGR2HSV) # HSV채널로 바꾸기  cv2.imshow('src',src)  cv2.imshow('dst',dst)  cv2.waitKey()  cv2.destroyAllWindows()   1. 산술 연산   img1 = cv2.imread('fig/lenna256.bmp', cv2.IMREAD\_GRAYSCALE)  img2 = np.zeros((256,256), dtype=np.uint8) # 256\*256의 검은색 영상  print(img1.dtype)  print(img2.dtype)  cv2.circle(img2, (128,128), 100, 100, -1)  cv2.circle(img2, (128,128), 50, 50, -1)  dst1 = cv2.add(img1, img2)  dst2 = cv2.addWeighted(img1, 0.5, img2, 0.5, 0.0) # img1 를 0.5배, img2도 0.5배  # 추가로 더하는건 없다 0.0, 합해서 1이 되게  dst3 = cv2.subtract(img2, img1) # 빼기  dst4 = cv2.absdiff(img1, img2) # 뺴고나서 절대값만  cv2.imshow('img1', img1)  cv2.imshow('img2', img2)  cv2.imshow('dst1', dst1)  cv2.imshow('dst2', dst2)  cv2.imshow('dst3', dst3)  cv2.imshow('dst4', dst4)  cv2.waitKey()  cv2.destroyAllWindows()  src = cv2.imread('fig/flowers\_rgb.jpg', cv2.IMREAD\_COLOR)  print(src.shape)  print(type(src)) # 넘파이네  # b, g, r = cv2.split(src) # 값이 높은건 해당 매트릭스....그거임 그레이스케일로나옴  # b = src[:, :, 0]  # g = src[:, :, 1]  # r = src[:, :, 2] # 이렇게 하기보단HSV채널을 훨씬 많이 씀  src\_hsv = cv2.cvtColor(src, cv2.COLOR\_BGR2HSV)  h,s,v = cv2.split(src\_hsv)  # h = np.clip(h\*2. , 0, 255).astype(np.uint8)  cv2.imshow('src', src)  # cv2.imshow('b\_channel', b)  # cv2.imshow('g\_channel', g)  # cv2.imshow('r\_channel', r)  cv2.imshow('h\_channel', h) # hue 색상  cv2.imshow('s\_channel', s) # 채도  cv2.imshow('v\_channel', v) # 명도  cv2.waitKey()  cv2.destroyAllWindows()   1. 히스토그램   import matplotlib.pyplot as plt  src = cv2.imread('fig/lenna.bmp', cv2.IMREAD\_GRAYSCALE)  # calcHist(images, channels, mask, histSize, ranges[, hist[, accumulate]])  # imgages : 입력영상 리스트(리스트로 입력)  # channels : 채널리스트, 3채널의 경우 [0,1,2]  # mask : 마스크 영상 입력, 영상전체는 None으로 지정  # histSize : 히스토그램 빈의 크기  # range : 히스토그램의 최솟값과 최댓값  # hist : 계산된 히스토그램, numpy.ndarray 타입  # accumulate : 기존의 히스토그램을 누적할 경우 True  hist = cv2.calcHist([src], [0], None, [256], [0, 256])  cv2.imshow('src', src)  cv2.waitKey()  cv2.destroyAllWindows()  plt.plot(hist)  plt.show()  src = cv2.imread('fig/lenna.bmp', cv2.IMREAD\_COLOR)  # calcHist(images, channels, mask, histSize, ranges[, hist[, accumulate]])  # imgages : 입력영상 리스트(리스트로 입력)  # channels : 채널리스트, 3채널의 경우 [0,1,2]  # mask : 마스크 영상 입력, 영상전체는 None으로 지정  # histSize : 히스토그램 빈의 크기  # range : 히스토그램의 최솟값과 최댓값  # hist : 계산된 히스토그램, numpy.ndarray 타입  # accumulate : 기존의 히스토그램을 누적할 경우 True  b, g, r = cv2.split(src)  hist\_b = cv2.calcHist([b], [0], None, [256], [0, 256])  hist\_g = cv2.calcHist([g], [0], None, [256], [0, 256])  hist\_r = cv2.calcHist([r], [0], None, [256], [0, 256])  cv2.imshow('src', src)  cv2.waitKey()  cv2.destroyAllWindows()  plt.plot(hist\_b, color = "b", label = "b")  plt.plot(hist\_g, color = "g", label = "g")  plt.plot(hist\_r, color = "r", label = "r")  plt.legend()  plt.show()   1. Normalize   import numpy as np  import sys  import cv2  src = cv2.imread('fig/Hawkes.jpg', cv2.IMREAD\_GRAYSCALE)  cv2.imshow('src', src)  # minVal, maxVal, minLoc, maxLoc : \_ 는 안쓴다는 뜻 고로 minVal,maxVal만  # smin, smax, \_, \_ = cv2.minMaxLoc(src)  # dst = np.clip(255\*(src - smin)/(smax - smin), 0 , 255).astype(np.uint8)  # print('min', smin)  # print('max', smax)  # cv2.normalize(src, dst=None, alpha=None, beta=None, norm\_type=None, dtype=None, mask=None) -> dst  # src: 입력영상  # dst: 결과영상  # alpha: 정규화 최소값 (예, 0)  # beta: 정규화 최댓값 (예, 155)  # norm\_type: cv2.NORM\_MINMAX  # dtype = -1, src와 같은 type  dst = cv2.normalize(src, None, 0, 255, cv2.NORM\_MINMAX, -1)  cv2.imshow('src', src)  cv2.imshow('dst', dst)  cv2.waitKey()  cv2.destroyAllWindows()   1. Equalize   src = cv2.imread('fig/Hawkes.jpg', cv2.IMREAD\_GRAYSCALE)  # equalizeHist(src, dst=None) -> dst  # src: 입력영상,gray scale 영상만 가능  dst\_equal = cv2.equalizeHist(src)  dst\_norm = cv2.normalize(src, None, 0, 255, cv2.NORM\_MINMAX, -1)  cv2.imshow('equ', dst\_equal)  cv2.imshow('norm', dst\_norm)  cv2.waitKey()  cv2.destroyAllWindows()   1. YcrCb   src = cv2.imread('fig/field.bmp')  src\_YCrCb = cv2.cvtColor(src, cv2.COLOR\_BGR2YCrCb)  src\_split = cv2.split(src\_YCrCb)  src\_split[0] = cv2.equalizeHist(src\_split[0])  dst\_ycrcb = cv2.merge(src\_split)  dst = cv2.cvtColor(dst\_ycrcb, cv2.COLOR\_YCrCb2BGR)  cv2.imshow('src', src)  cv2.imshow('dst', dst)  cv2.waitKey()  cv2.destroyAllWindows() | |