|  |  |
| --- | --- |
| 교육제목 | 데이터 기반 인공지능 시스템 엔지니어 양성 과정 |
| 교육일시 | 211116 |
| 교육장소 | YGL 학과장 및 자택(디스코드 사용 온라인 학습) |
| **교육내용** | |
| 1. 리사이즈로 확대하기   import numpy as np  import sys  import cv2  src = cv2.imread('fig/rose.jpg')  print(src.shape)  dst\_1 = cv2.resize(src, (2\*src.shape[0], 2\*src.shape[1]), interpolation = cv2.INTER\_NEAREST)  dst\_2 = cv2.resize(src, (2\*src.shape[0], 2\*src.shape[1]), interpolation = cv2.INTER\_LINEAR)  dst\_3 = cv2.resize(src, (2\*src.shape[0], 2\*src.shape[1]), interpolation = cv2.INTER\_CUBIC)  cv2.imshow('src', src)  cv2.imshow('dst\_1', dst\_1)  cv2.imshow('dst\_2', dst\_2)  cv2.imshow('dst\_3', dst\_3)    cv2.waitKey()  cv2.destroyAllWindows()   1. 이미지 회전하기   src = cv2.imread('fig/puppy.bmp')  rad = 20\*np.pi/180  rad\_30 = 30\*np.pi/180  M = np.array([[np.cos(rad), -np.sin(rad), 0],  [np.sin(rad), np.cos(rad), 0]], np.float32)  M1 = np.array([[np.cos(rad\_30), -np.sin(rad\_30), 0],  [np.sin(rad\_30), np.cos(rad\_30), 0]], np.float32)  dst = cv2.warpAffine(src, M, (0, 0))  dst2 = cv2.warpAffine(src, M1, (0, 0))  cv2.imshow('src', src)  cv2.imshow('dst', dst)  cv2.imshow('dst2', dst2)  cv2.waitKey()  cv2.destroyAllWindows()   * 이 경우 이미지가 0,0을 중심으로 회전하기 때문에 영상이 옆으로 밀려나가버린다  1. 영상 센터를 기준으로 회전하기   src = cv2.imread('fig/rose.jpg')  center = (src.shape[1]/2, src.shape[0]/2) # 이미지도메인에서 해서 xy순서바꿔줌  print(center)  affine = cv2.getRotationMatrix2D(center, -20, 1)  dst\_rot = cv2.warpAffine(src, affine, (0, 0), borderMode = cv2.BORDER\_REPLICATE ) # 0,0 출력영상의 크기. 똑같이 나옴.  cv2.imshow('src', src)  cv2.imshow('dst\_rot', dst\_rot)  cv2.waitKey()  cv2.destroyAllWindows()   1. 이건 뭐였지?   src = cv2.imread('fig/puppy.bmp')  rad = 20\*np.pi/180  rad\_30 = 30\*np.pi/180  M = np.array([[1, 0.2, 0],  [0, 1, 0]], np.float32)  h, w = src.shape[:2]  print(h, w)  dst = cv2.warpAffine(src, M, (w+100, h))  cv2.imshow('src', src)  cv2.imshow('dst', dst)  cv2.waitKey()  cv2.destroyAllWindows()   1. 필터로 엣지 강조   src = cv2.imread('fig/lenna.bmp', cv2.IMREAD\_GRAYSCALE)  kernel\_dx = np.array([[-1, 0, 1],  [-2, 0, 2],  [-1, 0 , 1]], dtype= np.float32)  # print(kernel\_dx)  # dst\_dx = cv2.fillter2D(src, kernel\_dx)  # cv2.Sobel(src, ddepth, dx, dy, dst, ksize, scale, delt, borderType) -> dst  # src : 입력영상  # ddepth : 출력영상의 데이터 타입 (-1)  # dx : x 방향 미분차수  # dy : x 방향 미분차수  # dst : 출력영상  # ksize : 커널의 크기  # scale : 연산결과에 추가적으로 곱할 값  # delta : 연산결과에 추가적으로 더할 값  # borderType : 가장자리 픽셀확장 방식  dx = cv2.Sobel(src, -1, 1, 0, delta=128)  dy = cv2.Sobel(src, -1, 0, 1, delta=128)  mag = cv2.magnitude(dx, dy)  cv2.imshow('src', src)  cv2.imshow('dx', dx)  cv2.imshow('dy', dy)  cv2.waitKey()  cv2.destroyAllWindows()   1. Sobel edge : 조금 더 edge를 구분하기 쉬워짐   src = cv2.imread('fig/lenna.bmp', cv2.IMREAD\_GRAYSCALE)  if src is None:  print('Image load failed!')  sys.exit()  dx = cv2.Sobel(src, cv2.CV\_32F, 1, 0)  dy = cv2.Sobel(src, cv2.CV\_32F, 0, 1)  mag = np.clip(cv2.magnitude(dx, dy), 0, 255).astype(np.uint8)  dx = np.clip(dx, 0, 255).astype(np.uint8)  dy = np.clip(dx, 0, 255).astype(np.uint8)  dst = np.zeros(mag.shape[:2], np.uint8)  dst[mag > 100] = 255  # \_, dst = cv2.threshold(mag, 120, 255, cv2.THRESH\_BINARY)  cv2.imshow('src', src)  cv2.imshow('mag', mag)  cv2.imshow('dst', dst)  cv2.waitKey()  cv2.destroyAllWindows()   1. Canny edge : 엣지 따기 가장 좋은 방법.   src = cv2.imread('fig/9910BB335DB7C32F03.jpg', cv2.IMREAD\_GRAYSCALE)  cv2.imshow('src', src)  dx = cv2.Sobel(src, cv2.CV\_32F, 1, 0)  dy = cv2.Sobel(src, cv2.CV\_32F, 0, 1)  # 1주면 uint8로 들어가니까 mag 계산이 안됨. 32플롯으로 바꿔서 아웃풋 나오게 해야해  # 매그니튜드 계산  mag = np.clip(cv2.magnitude(dx, dy), 0, 255).astype(np.uint8)  dst\_sobel = np.zeros(mag.shape, np.uint8)  dst\_sobel[mag>100] = 255  dst\_canny = cv2.Canny(src, 100, 150) # lower/upper limit  cv2.imshow('src', src)  cv2.imshow('dst\_sobel', dst\_sobel)  cv2.imshow('dst\_canny', dst\_canny)  cv2.waitKey()  cv2.destroyAllWindows()   1. Binary thresholding   src = cv2.imread('fig/cells.png', cv2.IMREAD\_GRAYSCALE)  if src is None:  print('image read failed')  sys.exit()    # cv2.hreshold(src, thresh, maxval, type, dst) -> retval, dst  # src: 입력영상 (다채널 8bit 또는 32bit 실수형)  # thres: 사용자 지정 임계값  # maxval : 최댓값 (255)  # type: 임계값 함수 동작 지정 (cv2.THRESH\_BINARY, cv2.THRESH\_OTSU(자동))  # retval: 사용된 임계값  # dst: 출력영상  # reval = 임계값. 200이하 0으로 지정하면 임계값은 200이 됨.  # 여기선 그냥 지정하니까 문제 x otsu방법에서는 임계값을 구..하나? 아니다 자동으로 구함  reval1, dst1 = cv2.threshold(src, 200, 255, cv2.THRESH\_BINARY) #200이하 = 0  reval2, dst2 = cv2.threshold(src, 200, 255, cv2.THRESH\_BINARY\_INV)  # 인버전(inv)하면 뒤집혀서 마스크됨  cv2.imshow('src', src)  cv2.imshow('dst1', dst1)  cv2.imshow('dst2', dst2)  cv2.waitKey()  cv2.destroyAllWindows()   1. Thresh에 트랙바 넣기   def call\_trackbar(pos):  reval, dst = cv2.threshold(src, pos, 255,  cv2.THRESH\_BINARY\_INV) # pos 첫값 255 마지막값  cv2.imshow('dst',dst)              src = cv2.imread('fig/cells.png', cv2.IMREAD\_GRAYSCALE)  if src is None:  print('image read failed')  sys.exit()      cv2.imshow('src', src)  cv2.namedWindow('dst')  cv2.createTrackbar('level', 'dst', 128, 255, call\_trackbar)# 트랙바 생성, 트랙바이름,띄울창이름,이니셜,끝단  cv2.waitKey()  cv2.destroyAllWindows()   1. Otsu 사용.   src = cv2.imread('fig/rice.png', cv2.IMREAD\_GRAYSCALE)  thres , dst1 = cv2.threshold(src, 0, 255, cv2.THRESH\_OTSU) # 전체이진화 완벽하지않음  print("otsu's threshold:", thres)  dst2 = np.zeros(src.shape, np.uint8) #까만영상  bw = src.shape[1]//4  bh = src.shape[0]//4  for y in range(4): # 0 1 2 3  for x in range(4):  src\_ = src[y\*bh:(y+1)\*bh, x\*bw:(x+1)\*bw]  dst\_ = dst2[y\*bh:(y+1)\*bh, x\*bw:(x+1)\*bw]  cv2.threshold(src\_, 0, 255, cv2.THRESH\_OTSU, dst\_)    # print(src.shape)  # print(bw)  # print(bh)  cv2.imshow('src', src)  cv2.imshow('dst1', dst1)  cv2.imshow('dst2', dst2)  cv2.waitKey()  cv2.destroyAllWindows()   1. Thresh Gaussian   src = cv2.imread('fig/sudoku.jpg', cv2.IMREAD\_GRAYSCALE)  # cv2.adaptiveThreshold(src, maxValue, adaptiveMethod, thresholdType, blockSize, C, dst) -> dst  # src: 입력영상  # maxValue: 임계값 최대치, 255  # adaptiveMethod: 지역이진화 전 블러링함수, 예 cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C/cv2.ADAPTIVE\_THRESH\_mean\_C  # thresholdType:cv2.THRESH\_BINARY 또는 cv2.THRESH\_BINARY\_INV  # blocksize: 3이상의 홀수  # C: 블록 내 평균값  dst = cv2.adaptiveThreshold(src, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,  cv2.THRESH\_BINARY, 11, 5)  cv2.imshow('src',src)  cv2.imshow('dst',dst)  cv2.waitKey()  cv2.destroyAllWindows()   1. Erode, Dilate(침식, 팽창)   src = cv2.imread('fig/circuit.bmp', cv2.IMREAD\_GRAYSCALE)  # cv2.erode(src, kernel, dst = None, anchor=None, iterations=None, borderType=None, borderValue=None)  # src: 입력영상 (이진영상)  # kernel: 커널 사이즈, None = 3 x 3  # iterations: 여러번 수행  # anchor: 고정점. 기본값 (-1, -1) 중앙점사용  # cv2.getStructuringElemen(shape, ksize, anchor=None)-> retval  # shape: cv.MORPH\_RECT(사각형), cv.MORPH\_CROSS(십자가), cv.MORPH\_ELLIPSE(좌우로휨)  kernel = cv2.getStructuringElement(cv2.MORPH\_RECT, (3,5)) #커널을 위아래로 길게 해서 끊긴 회로 연결  dst1 = cv2.erode(src, kernel) # 침식  dst2 = cv2.dilate(src, kernel) # 팽창  dst3 = cv2.erode(dst2, kernel) # 팽창을 침식  cv2.imshow('src',src)  cv2.imshow('dst1',dst1)  cv2.imshow('dst2',dst2)  cv2.imshow('dst3',dst3)  cv2.waitKey()  cv2.destroyAllWindows()   1. 레이블링   src = cv2.imread('fig/keyboard.jpg', cv2.IMREAD\_GRAYSCALE)  # otsu 쓸거면 0 필요없긴하다  ret, dst = cv2.threshold(src, 0, 255, cv2.THRESH\_OTSU)  kernel = cv2.getStructuringElement(cv2.MORPH\_RECT, (3,3))  dst\_morph = cv2.morphologyEx(dst, cv2.MORPH\_OPEN, kernel)  dst = cv2.cvtColor(src, cv2.COLOR\_GRAY2BGR)  cnt, labels, stats, centroids = cv2.connectedComponentsWithStats(dst\_morph) # 여러개나옴  # connectedComponentsWithStats(image[, labels[, stats[, centroids[, connectivity[, ltype]]]]]) -> retval, labels, stats, centroids  print(cnt)  for i in range(1, cnt): # 0은 백그라운드니까 1부터 카운트까지  x, y, w, h, area = stats[i]    if area > 10000 or area < 200:  continue    cv2.rectangle(dst, (x,y,w,h), (0, 0, 255), 2)  cv2.imshow('src',src)  cv2.imshow('dst\_morph',dst)  cv2.waitKey()  cv2.destroyAllWindows() | |