|  |  |
| --- | --- |
| 교육 제목 | AI 프로젝트 기반 S/W 전문가 양성과정 |
| 교육 일시 | 21.12.27 |
| 교육 장소 | 집 |
| **교육 내용** | |
| 오전 | 1. 히스토그램 역행 cv2.calcBackProject([src\_ycrcb], hist, [1, 2], [256, 256], [0, 256], 1) calcBackProject(images, channels, hist, ranges, scale[, dst]) -> dst 2. blur – mean filter kernel = np.ones((3, 3), np.float32) / 9 cv2.filter2D(src, -1, kernel) # 컨볼루션 cv2.blur(src, (3, 3)) 3. blur – gaussian cv2.GaussianBlur(src, (0, 0), 1) 마지막 1은 sigma 4. sharpening cv2.addWeighted(src, 2, dst\_g, -1, 0) # 2는 alpha, -1은 beta, 0은 감마 5. 양방향 필터 – bilateral filter cv2.bilateralFilter(src, -1, 5, 1) # 5는 sigmacolor, 1은 sigmaspace 6. 기하학적 변환 - 늘리기 affine = np.array([[1, 0, 20], [0, 1, 100]], np.float32) cv2.warpAffine(src, affine, (0, 0)) 7. 기하학적 변환 – 회전 affine = cv2.getRotationMatrix2D(cp, 20, 1) dst = cv2.warpAffine(src, affine, (0, 0)) 8. caffe models http://dl.caffe.berkeleyvision.org/ 9. 모델을 갖다 쓰려면 model, config, classified파일 필요 10. 사용할 네트워크(모델) 불러들이기 model = "./DNN/googlenet/bvlc\_googlenet.caffemodel" config = "./DNN/googlenet/deploy.prototxt" net = cv2.dnn.readNet(model, config) cv2.dnn.readNet(model[, config[, framework]]) -> retval 11. blob image 만들기 – 이미지 전처리 blob = cv2.dnn.blobFromImage(img, 1, (224, 224), (104, 117, 123), swapRB = False) cv2.dnn.blobFromImage(image[, scalefactor[, size[, mean[, swapRB[, crop[, ddepth]]]]]]) -> retval 12. for문 돌려보기 실습 |
| 오후 | 1. \*.prototxt = caffemodel config \*.pbtxt = tensorflow model config 2. 안면인식 caffemodel, deploy.prototxt 준비해서 모델 돌리기 pb, pbtxt 준비해서 모델 돌리기 out = facenet.forward()으로 모델을 돌리는데 out return은 (1, 1, 200, 7)으로 나옴 idx 0, 1 dim은 필요 없어서 버리고 200, 7만 갖고 씀  c는 confidence x1, y1, x2, y2는 각 꼭지점의 좌표임 좌표를 갖다가 이미지에 rectangle 만들고 그보다 조금 위 픽셀에 confidence 값을 print해 줌 3. 웹캠 동영상으로 안면인식 실습 4. YOLO3  model = "./DNN/yolo\_v3/yolov3.weights" 🡪 model config = "./DNN/yolo\_v3/yolov3.cfg" 🡪 config class\_labels = "./DNN/yolo\_v3/coco.names" 🡪 class net = cv2.dnn.readNet(model, config) net.getLayerNames() 🡪 레이어의 이름 뽑아내기 output\_layers = [layer\_names[i-1] for i in net.getUnconnectedOutLayers()] net.forward(output\_layers) 🡪 layer 이름 뽑은거 넣어주기 h, w = img.shape[:2] 가져오기 class, confidence, box 각각의 비어있는 리스트 할당(여러장의 사진) 인식하는 개체가 여러 개니까 out이 여러 개임 아래의 구조로 return 받음  각각의 value들을 뽑아서 쓰기 |