2021.10.15 양진현

|  |
| --- |
| 1번 코드 |
| import cv2 import numpy as np import glob import pickle # prepare object points, like (0,0,0), (1,0,0), (2,0,0) ....,(6,5,0) def undistort(img, cal\_dir='camera\_cal/wide\_dist\_pickle.p'):  with open(cal\_dir, mode='rb') as f:  file = pickle.load(f)  mtx = file['mtx']  dist = file['dist']  dst = cv2.undistort(img, mtx, dist, None, mtx)   return dst  objp = np.zeros((6\*9,3), np.float32) objp[:,:2] = np.mgrid[0:9, 0:6].T.reshape(-1,2)  # Arrays to store object points and image points from all the images. objpoints = [] # 3d points in real world space imgpoints = [] # 2d points in image plane. # Make a list of calibration images  objpoints = [] # 3d points in real world space imgpoints = [] # 2d points in image plane. # Make a list of calibration images  # 사용자가 제시한 조건에 맞는 파일명을 리스트 형식으로 반환 images = glob.glob('camera\_cal/\*.jpg') # Step through the list and search for chessboard corners  total\_images = len(images) for idx, fname in enumerate(images):  img = cv2.imread(fname)  gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  # Find the chessboard corners  ret, corners = cv2.findChessboardCorners(gray, (9,6), None)  # If found, add object points, image points  if ret == True:  objpoints.append(objp)  imgpoints.append(corners)  # Draw and display the corners  cv2.drawChessboardCorners(img, (9, 6), corners, ret)  write\_name = 'camera\_cal/result/corners\_found' + str(idx) + '.jpg'  cv2.imwrite(write\_name, img)  cv2.imshow('img', img)  cv2.waitKey(500)  idx+=1 cv2.destroyAllWindows()  img = cv2.imread('camera\_cal/test\_cal.jpg') height, width = img.shape[:2] img\_size = (width, height) # Do camera calibration given object points and image points ret, mtx, dist, rvecs, tvecs = cv2.calibrateCamera(objpoints, imgpoints,  img\_size, None,None) dst = cv2.undistort(img, mtx, dist, None, mtx) cv2.imwrite('camera\_cal/result/test\_undist.jpg',dst)  # Save the camera calibration result for later use (we won't worry about rvecs / tvecs) dist\_pickle = {} dist\_pickle["mtx"] = mtx dist\_pickle["dist"] = dist print("파라미터 값\n",mtx,dist) pickle.dump( dist\_pickle, open( "camera\_cal/wide\_dist\_pickle.p", "wb" ) )  cap=cv2.VideoCapture("video/drive.mp4") if cap.isOpened()==False:  print("카메라를 열 수 없습니다") while True:  rt, img = cap.read()   #동영상이 끝나면 재생되는 프레임의 위치를 0으로 다시 지정  if cap.get(cv2.CAP\_PROP\_POS\_FRAMES) == cap.get(cv2.CAP\_PROP\_FRAME\_COUNT):  cap.set(cv2.CAP\_PROP\_POS\_FRAMES, 0)  img\_dst = undistort(img)  videos=cv2.hconcat([img,img\_dst])  cv2.imshow("original video & calibrate video",videos)  key = cv2.waitKey(20)  if key == 27:  break  cv2.destroyAllWindows()  cv2.waitKey(0) cv2.destroyAllWindows() |

|  |
| --- |
| 2번 코드 |
| import cv2 import numpy as np import glob import pickle  obj\_pts = np.zeros((6\*9,3), np.float32) obj\_pts[:,:2] = np.mgrid[0:9, 0:6].T.reshape(-1,2) *# Stores all object points & img points from all images* objpoints = [] imgpoints = []  images = glob.glob('camera\_cal/\*.jpg') total\_images = len(images) for idx, fname in enumerate(images):  img = cv2.imread(fname)  gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  *# Find the chessboard corners* ret, corners = cv2.findChessboardCorners(gray, (9,6), None)  *# If found, add object points, image points* if ret == True:  objpoints.append(objp)  imgpoints.append(corners)  *# Draw and display the corners* cv2.drawChessboardCorners(img, (9, 6), corners, ret)  write\_name = 'camera\_cal/result/corners\_found' + str(idx) + '.jpg'  cv2.imwrite(write\_name, img)  cv2.imshow('img', img)  cv2.waitKey(500)  idx+=1 cv2.destroyAllWindows() |

|  |
| --- |
| 2번 결과 |
|  |

|  |
| --- |
| 3번 코드 |
| import cv2 import numpy as np import glob import pickle  img = cv2.imread('camera\_cal/test\_cal.jpg') height, width = img.shape[:2] img\_size = (width, height) *# Do camera calibration given object points and image points* ret, mtx, dist, rvecs, tvecs = cv2.calibrateCamera(objpoints, imgpoints,  img\_size, None,None) dst = cv2.undistort(img, mtx, dist, None, mtx) cv2.imwrite('camera\_cal/result/test\_undist.jpg',dst)  *# Save the camera calibration result for later use (we won't worry about rvecs / tvecs)* dist\_pickle = {} dist\_pickle["mtx"] = mtx dist\_pickle["dist"] = dist  print("파라미터 값\n",mtx,dist) |

|  |
| --- |
| 3번 결과 |
|  |

|  |
| --- |
| 4번 코드 |
| import cv2 import numpy as np import glob import pickle  def undistort(img, cal\_dir='camera\_cal/wide\_dist\_pickle.p'):  with open(cal\_dir, mode='rb') as f:  file = pickle.load(f)  mtx = file['mtx']  dist = file['dist']  dst = cv2.undistort(img, mtx, dist, None, mtx)   return dst  cap=cv2.VideoCapture("video/drive.mp4") if cap.isOpened()==False:  print("카메라를 열 수 없습니다") while True:  rt, img = cap.read()   *#동영상이 끝나면 재생되는 프레임의 위치를 0으로 다시 지정* if cap.get(cv2.CAP\_PROP\_POS\_FRAMES) == cap.get(cv2.CAP\_PROP\_FRAME\_COUNT):  cap.set(cv2.CAP\_PROP\_POS\_FRAMES, 0)  img\_dst = undistort(img)  videos=cv2.hconcat([img,img\_dst])  cv2.imshow("original video&calibrate video",videos)  key = cv2.waitKey(20)  if key == 27:  break  cv2.destroyAllWindows() |

|  |
| --- |
| 4번 결과 |
|  |