

```

1 # Python Face_Recognition
2 # https://face-recognition.readthedocs.io/en/latest/readme.html#
3 # conda create -n facer opencv pillow cmake scipy scikit-learn natsort
4 # conda activate facer
5 # conda install -c conda-forge dlib
6 # conda install -c akode face_recognition_models
7 # pip install --no-dependencies face_recognition
8
9 import face_recognition
10 import os, sys, cv2, natsort
11
12 FRAME_THICKNESS = 3
13 FONT_THICKNESS = 2
14 MODEL = "cnn" # "hog"
15 COLOR_GRAY = (200,200,200)
16
17 def recog_face_from_directory(known_faces_dir="known", unknown_faces_dir="unknown", tolerance=0.6) :
18     # tolerance = 0.6 # 낮을수록 엄밀한 비교
19     print("알고 있는 얼굴 로딩...")
20     known_faces = []
21     known_names = []
22
23     for name in os.listdir(known_faces_dir):
24         for filename in os.listdir(f"{known_faces_dir}/{name}"):
25             image = face_recognition.load_image_file(f"{known_faces_dir}/{name}/{filename}")
26             # 영상내의 각각의 얼굴에 대하여 128차원의 face encoding 정보 구하기
27             encoding = face_recognition.face_encodings(image)[0]
28             known_faces.append(encoding)
29             known_names.append(name)
30
31     print("Wt폴더명:", known_names)
32     print("모르는 얼굴 로딩...")
33     for filename in os.listdir(unknown_faces_dir):
34         print("Wt", filename, ".....")
35         image = face_recognition.load_image_file(f"{unknown_faces_dir}/{filename}")
36         locations = face_recognition.face_locations(image) # (top, right, bottom, left)
37         print("얼굴영역:", locations)
38         encodings = face_recognition.face_encodings(image, locations)
39         image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
40
41         for face_encoding, face_location in zip(encodings, locations):
42             results = face_recognition.compare_faces(known_faces, face_encoding, tolerance)
43             distances = face_recognition.face_distance(known_faces, face_encoding)
44             print("거리:", distances)
45
46             match = None
47             if True in results:
48                 match = known_names[results.index(True)]
49                 print(f"Match 발견: {match}")
50
51                 top_left = (face_location[3], face_location[0])
52                 bottom_right = (face_location[1], face_location[2])
53                 color = [0, 255, 0]
54                 cv2.rectangle(image, top_left, bottom_right, color, FRAME_THICKNESS)
55
56                 top_left = (face_location[3], face_location[2])
57                 bottom_right = (face_location[1], face_location[2]+20)
58                 cv2.rectangle(image, top_left, bottom_right, color, cv2.FILLED)
59                 cv2.putText(image, match, (face_location[3]+10, face_location[2]+15),
60                             cv2.FONT_HERSHEY_SIMPLEX, 3.0, COLOR_GRAY, FONT_THICKNESS)
61
62     cv2.namedWindow(filename, cv2.WINDOW_NORMAL)
63     cv2.resizeWindow(filename, 1024, 768)
64     cv2.moveWindow(filename, 100, 50)
65

```

```

66     cv2.imshow(filename, image)
67     cv2.waitKey(1)
68     cv2.destroyAllWindows(filename)
69
70 def recog_face_from_cam(cam_no, known_faces_dir = "known", tolerance=0.6) :
71     # tolerance = 0.6 # 낮을수록 엄밀한 비교
72     cap = cv2.VideoCapture(cam_no)
73
74     print("알고 있는 얼굴 로딩...")
75     known_faces = []
76     known_names = []
77
78     for name in os.listdir(known_faces_dir):
79         for filename in os.listdir(f"{known_faces_dir}/{name}"):
80             image = face_recognition.load_image_file(f"{known_faces_dir}/{name}/{filename}")
81             # 영상내의 각각의 얼굴에 대하여 128차원의 face encoding정보 구하기
82             encoding = face_recognition.face_encodings(image)[0]
83             known_faces.append(encoding)
84             known_names.append(name)
85
86     while True :
87         ret, frame = cap.read()
88
89         # 카메라 영상, BGR형식에서 RGB형식으로 변환
90         image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
91         locations = face_recognition.face_locations(image, model="hog") # (top, right, bottom, left)
92         encodings = face_recognition.face_encodings(image, locations)
93
94         print("얼굴영역:", locations)
95         encodings = face_recognition.face_encodings(image, locations)
96         image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
97
98         for face_encoding, face_location in zip(encodings, locations):
99             results = face_recognition.compare_faces(known_faces, face_encoding, tolerance)
100             distances = face_recognition.face_distance(known_faces, face_encoding)
101             print("거리:", distances)
102
103             match = None
104             if True in results:
105                 match = known_names[results.index(True)]
106                 print(f"Match 발견: {match}")
107
108                 top_left = (face_location[3], face_location[0])
109                 bottom_right = (face_location[1], face_location[2])
110                 color = [0, 255, 0]
111                 cv2.rectangle(image, top_left, bottom_right, color, FRAME_THICKNESS)
112
113                 top_left = (face_location[3], face_location[2])
114                 bottom_right = (face_location[1], face_location[2]+20)
115                 cv2.rectangle(image, top_left, bottom_right, color, cv2.FILLED)
116                 cv2.putText(image, match, (face_location[3]+10, face_location[2]+15),
117                             cv2.FONT_HERSHEY_SIMPLEX, 3.0, COLOR_GRAY, FONT_THICKNESS)
118
119             cv2.namedWindow("CAM", cv2.WINDOW_NORMAL)
120             cv2.resizeWindow("CAM", 1024, 768)
121             cv2.moveWindow("CAM", 100, 50)
122
123             cv2.imshow("CAM", image)
124             inKey = cv2.waitKey(1)
125             if inKey == 27 :
126                 cv2.destroyAllWindows("CAM")
127                 break
128
129 def capture_face_from_cam(camNo, capture_faces_dir = "captured", tolerance=0.6) :
130     # tolerance = 0.6 # 낮을수록 엄밀한 비교
131     cap = cv2.VideoCapture(camNo)

```

```

132
133     if not os.path.isdir(capture_faces_dir):
134         os.mkdir(capture_faces_dir)
135
136     fileList = os.listdir(capture_faces_dir)
137     if len(fileList) == 0 :
138         nextNo = 1
139     else :
140         orderedFileList = natsort.natsorted(fileList)
141         print("orderedFileList",orderedFileList)
142         nextNo = int(orderedFileList[len(orderedFileList)-1][:4])+1
143
144 while True :
145     ret, frame = cap.read()
146
147     # 카메라 영상, BGR형식에서 RGB형식으로 변환
148     image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
149
150     cv2.imshow("CAM", frame)
151     inKey = cv2.waitKey(1)
152     if inKey == 27 :
153         cv2.destroyAllWindows("CAM")
154         break
155     elif inKey == ord('s') :
156         filename = capture_faces_dir+"/"+str(nextNo)+".png" # JPG 말고 PNG로
157         print(filename," is saved")
158         cv2.imwrite(filename, frame)
159         nextNo += 1
160
161 if __name__ == "__main__" :
162     if len(sys.argv) > 1:
163         if len(sys.argv)>2 :
164             camNo = int(sys.argv[2])
165         else :
166             camNo = 0
167
168         if sys.argv[1] == 'dir':
169             recog_face_from_directory("known", "unknown", 0.6)
170         elif sys.argv[1] == 'cam' :
171             recog_face_from_cam(camNo, "known", 0.6)
172         elif sys.argv[1] == 'capture' :
173             capture_face_from_cam(camNo, "captured")
174     else :
175         print("사용법: python facerecog_cam.py dir/cam/capture")
176         print("사용법: python facerecog_cam.py capture cam_no")

```