2024 오픈소스프로그래밍 팀프로젝트 보고서

프로젝트명	영상처리 기술 기반 얼굴인식 기술 연구 및 챗봇 구현			
팀명	현민수	팀장	학번	이름
			2313177	오현서
수행기간	2024. 05. 15 ~ 2024. 06. 21 (학기말까지)	팀원	2316452	김민경
			2315426	김수연

가. 목표 및 기대효과

- 기술개발 배경

◦최근 몇십 년간 컴퓨터 비전 기술은 급속히 발전하여, 얼굴인식 및 객체 인식 등의 분야에서 상당한 성과를 거둠.
◦얼굴인식 분야는 시각 장애인 및 안면 인식 장애 환자들이 보다 독립적으로 생활할 수 있는 기회를 제공하는 데 중요한 역할을 함.
◦안면 인식 장애 환자들은 사람들의 얼굴을 인식하고 식별하는데 어려움을 겪음, 이에 카메라로 인식한 인물의 정보를 채팅(글) 등을 통해 제공하기 위함
◦인공지능 기술의 진보는 개인정보 보호와 안전 문제에도 긍정적인 영향을 미침.
예) 얼굴인식 기술을 이용해 잠재적 범죄자나 접근이 제한되어야 하는 지역에 대한 접근을 제한하는 등의 안전 및 보안 조치가 가능해짐
◦이는 안면 인식 장애인뿐만 아니라 인지능력이 부족한 미취학 아동들이 범죄에 노출될 확률이 낮아지는 효과를 기대할 수 있음.
◦전통적인 시각 보조 도구나 안전장치는 한계가 존재함. 예로 점자나 음성 안내 시스템, 아동을 위한 보호 장치는 사용자의 의도에 따라 즉각적으로 대응하기 어려움. 얼굴 인식 기술은 이러한 한계를 극복하고 보다 효율적이고 즉각적인 지원을 제공할 수 있을 것이라 기대됨.

- 기술개발 기대효과

[기술적 기대효과]
◦영상처리 기술을 통한 안면인식 기술 및 알림 시스템 기술 확보
◦비대면 수업 등의 다양한 인터랙션 서비스에 적용 가능한 기술
◦얼굴 인식과 그 사람의 정보와 매치 기술
[사회적 기대효과]
◦시각 장애인, 알츠하이머 환자 등 안면 인식 장애를 지닌 약자에게 타인 도움 없이 가족, 직장 동료 등 주변인을 화면이나 음성을 통해 인지할 수 있도록 함.
◦낯선 사람을 인지하기 어려운 어린 아이들을 범죄로부터 보호 가능성을 기대할 수 있음.
◦생위를 통해 지병을 지닌 사람과 그렇지 않은 사람의 차이를 좁힘.

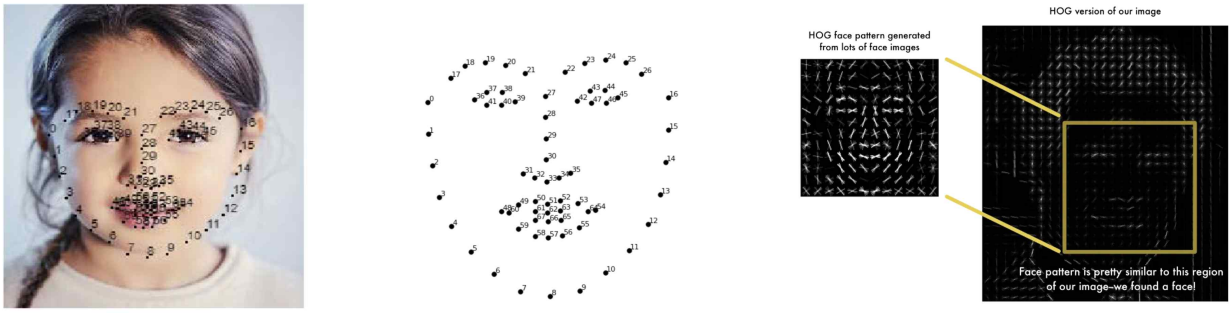
- 기술개발 목표

(개발 서비스 목표 및 기능 중심으로 간략하게 서술할 것)
◦서비스 목표
- 얼굴인식 기반 상대인지 챗봇을 통해 시각 장애인과 인지능력 부족 아동이 주변 사람들을 쉽게 인식하고 식별할 수 있도록 하여 시각 장애인 및 인지능력 부족 아동의 독립성 증진시킬 수 있음.
- 안전 및 보안
인식된 얼굴과 이름 등의 정보를 제공하여 시각 장애인 및 인지 능력 부족 아동이 안전하게 사회적 상호작용을 할 수 있도록 지원.
- 정보 접근성 향상:
얼굴인식과 챗봇 기술을 결합하여 사용자에게 유용한 정보를 제공하고, 이들의 정보 접근성

을 높임.

나. 프로젝트 개발 내용

- 연구개발 내용



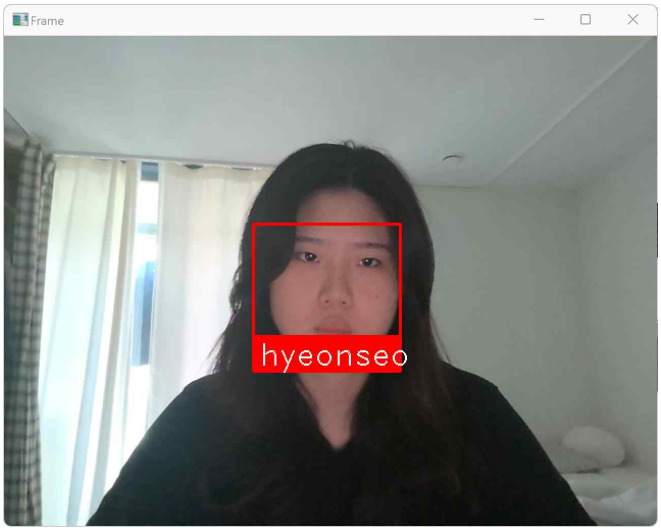
[그림 1] 얼굴 랜드마크

본 주제에서는 [그림1]과 같이 얼굴의 랜드마크를 기반으로 얼굴인식 시스템을 개발하고자 함. opencv-python, opencv-contrib-python, dlib, face_recognition, flask 등의 패키지를 사용하여 얼굴의 영역을 찾아내어 Face Recognition 모듈과 Unknown Face Classifier 모듈을 개발하고자 함.

○ **얼굴 인식**

1) known (이미 학습된 데이터)

opencv-python, opencv-contrib-python, dlib, face_recognition, flask등의 패키지를 설치하고 이를 활용해 얼굴 영역을 찾아내어 이미 학습 되어있는 정보를 통해 해당 인물의 이름을 표시한다.



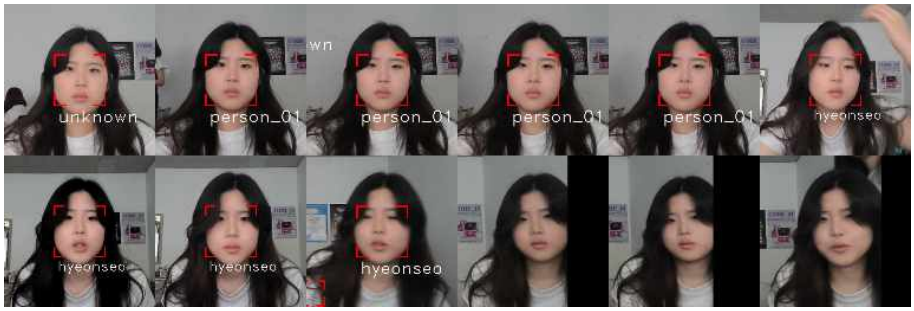
[그림 2] face_recog.py 파일 얼굴 인식 결과

2) Unknown (학습되지 않은 데이터)

opencv-python, opencv-contrib-python, dlib, face_recognition, imutils 등의 패키지를 설치하고 이를 활용해 얼굴 영역을 찾아내어 학습되어 있지 않은 사람들의 얼굴을 인식한다.

사전에 학습되어 있던 사람은 face_encoding 값을 가지고 있다. 사진에서 얼굴을 인식하고, 그 얼굴을 사전에 알고 있는 사람의 face_encoding과 비교하면, 가장 거리가 가까운 (즉, 가장 비슷한) 사람을 찾아낼 수 있다. face_encoding은 128 차원의 벡터이므로 수학적으로 유클리드 거리를 이용하면 두 벡터의 거리를 구할 수 있고, 이를 파이썬에 적용하면 numpy의 inalg.norm() 함수를 이용할 수 있다.

face_recognition의 face_distance() 함수는 numpy의 linalg.norm() 함수의 wrapper로 구현되어 있다. 두 face_encoding의 거리는 0 ~ 1 사이의 값으로 구해지고, 이 값은 두 얼굴이 얼마나 유사 한지에 대한 척도로 사용된다.



[그림 3] unknown_face_classifier result montage 사진

```
def compare_with_known_persons(self, face, persons):
    # distance를 구함
    encodings = [person.encoding for person in persons]
    distances = face_recognition.face_distance(encodings, face.encoding)
    index = np.argmin(distances)
    min_value = distances[index]    # distance의 최소값을 구함
    if min_value < self.similarity_threshold:
        # distance의 최소값이 simalarity_threshold보다 작으면 같은 사람의 얼굴임
        persons[index].add_face(face)
        persons[index].calculate_average_encoding()   # 얼굴의 face_encoding의 평균
        return persons[index]
```

[그림 4] 비교 알고리즘

[그림4]는 인식된 얼굴을 기존에 아는 사람의 얼굴과 비교하는 알고리즘

1. 새로 얼굴이 인식되면, 아는 사람들의 face_encoding과의 거리를 구한다.
2. 가장 가까운 사람과의 거리가 similarity_threshold보다 작으면 그 사람의 얼굴이라고 판단하고, 그 사람의 얼굴 DB에 얼굴을 추가한다. (similarity_threshold: 비슷하게 생겼지만 다른 사람이라는 의미)
3. 새로 추가된 얼굴을 포함하여 그 사람의 face_encoding을 업데이트한다. 얼굴이 추가될 때마다 그 사람의 모든 얼굴의 face_encoding의 평균을 구하여 놓는다.

밑에 [그림 5]은 모르는 사람의 얼굴을 비교하여 새로운 사람을 찾아내는 알고리즘

1. 모르는 얼굴은 unknown_faces에 따로 저장해 놓는다.
2. 새로 인식된 얼굴과 모르는 얼굴들(unknown_faces)과의 거리를 구한다.
3. 가장 가까운 얼굴와의 거리가 similarity_threshold보다 작으면 두 얼굴은 같은 사람의 얼굴이라고 판단하고, 새로운 사람을 만든다.
4. 가장 가까운 얼굴와의 거리가 similarity_threshold보다 크면 unknown_faces에 추가한다.
5. 새로 만들어진 사람의 이름은 person_## 으로 자동으로 부여되고, ##은 사람이 추가될 때마다 증가한다.

○ **텔레그램 봇**

```python
def compare_with_unknown_faces(self, face, unknown_faces):
    # distance를 구함
    encodings = [face.encoding for face in unknown_faces]
    distances = face_recognition.face_distance(encodings, face.encoding)
    index = np.argmin(distances)
    min_value = distances[index]    # distance의 최소값을 구함
    if min_value < self.similarity_threshold:
        # two faces are similar - create new person with two faces
        person = Person()
        newly_known_face = unknown_faces.pop(index)
        person.add_face(newly_known_face)
        person.add_face(face)
        person.calculate_average_encoding()   # 얼굴의 face_encoding의 평균
        return person
    else:
        # unknown face
        unknown_faces.append(face)
        return None
```

[그림 5] 새로운 인물 탐색 알고리즘

1) 기능

비디오에서 얼굴을 인식 후 인식된 얼굴을 사람 별로 분류. 모르는 사람의 얼굴이라도 비슷한 얼굴끼리 분류하여 처음 보는 사람이 나타나거나, 알고 있는 사람이 다시 나타나면 텔레그램으로 알림을 보낸다. 즉 실시간으로 얼굴을 인식하여 얼굴에 대한 정보를 사용자에게 전송한다.

주요 알고리즘

```
(py3) $ python visitor_alarm_telegram_bot.py -h
usage: visitor_alarm_telegram_bot.py [-h] --token TOKEN [--srcfile SRCFILE]
                                     [--threshold THRESHOLD] [--sbf SBF]
                                     [--resize-ratio RESIZE_RATIO]
                                     [--appearance-interval APPEARANCE_INTERVAL]

optional arguments:
  -h, --help            show this help message and exit
  --token TOKEN         Telegram Bot Token
  --srcfile SRCFILE     Video file to process. If not specified, web cam is
                        used.
  --threshold THRESHOLD
                        threshold of the similarity (default=0.42)
  --sbf SBF             second between frame processed (default=0.5)
  --resize-ratio RESIZE_RATIO
                        resize the frame to process (less time, less accuracy)
  --appearance-interval APPEARANCE_INTERVAL
                        alarm interval second between appearance (default=10)
```

[그림 6] 텔레그램 봇 코드 파일 -h 결과

--token: 텔레그램 봇의 토큰

--srcfile: 입력 비디오로 웹캠이 아니라 비디오 파일이 사용

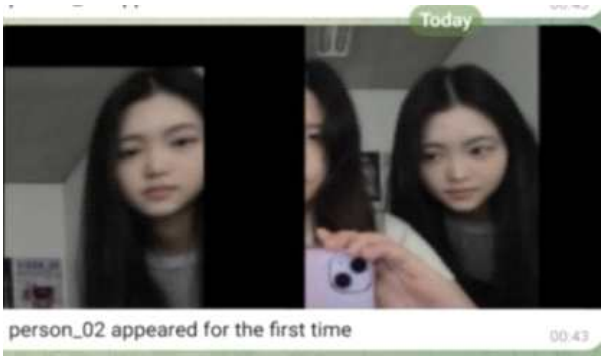
--threshold: 인식된 얼굴을 동일인으로 분류할 기준.

--sbf: 처리할 비디오 프레임의 시간 간격 (초)

--resize-ratio: 비디오 크기를 줄여서 처리하도록 함. 하지만 처리 시간이 줄어드는 대신에 정확도는 낮아짐.

--appearance-interval: 어떤 사람의 얼굴이, 이 시간보다 길게 보이지 않다가 다시 인식되면,

재출현으로 판단



[그림 7 텔렘그램봇이 전송한 사진]

1) 처음 보는 사람이 출현할 때

Face classifier가 인식된 얼굴을 처음 보는 사람으로 분류하면,

"2024-06-18 10:22:40,659 - person_01 appeared for the first time"

PC에서는 위와 같은 로그가 출력되고 텔레그램에는 [그림 7]과 같이 전달된다.

[그림 8 다시 인식된 얼굴 전송]

2) 이전에 출현했던 사람이 출현할 때

PC에서는 아래와 같은 로그가 출력되고

"2024-06-18 10:23:06,830 - person_01 appeared again in 22 seconds since 2024-06-21 10:22:43"

텔레그램에는 [그림 8]과 같이 전달된다.

다. 추진일정 및 업무분장

1) 역할 분담 계획

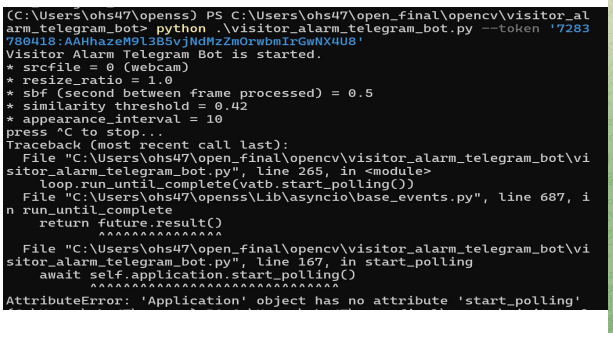
구 분	성 명	담당 업무	세부개발 사항
팀장	오현서		
팀원-1	김민경		
팀원-2	김수연		

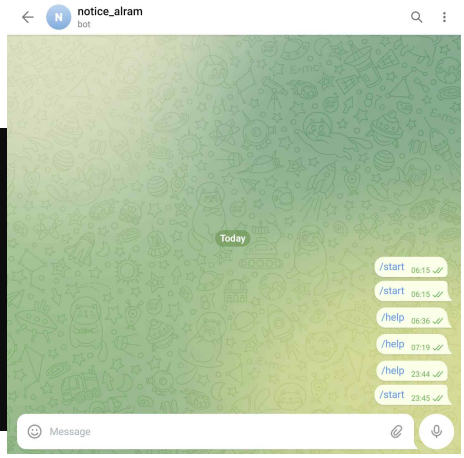
2) 일정 계획

세부 개발내용	과제 수행 기간				
	5월2주	5월 3주	5월 4주	6월 1주	6월 2주
1. 개발 환경 설정					
2. 이미지 얼굴 인식 모듈 개발					
3. 비지오 얼굴 인식 모듈 개발					
4. 텔레그램봇과 연결					

다. 프로젝트 개발 결과

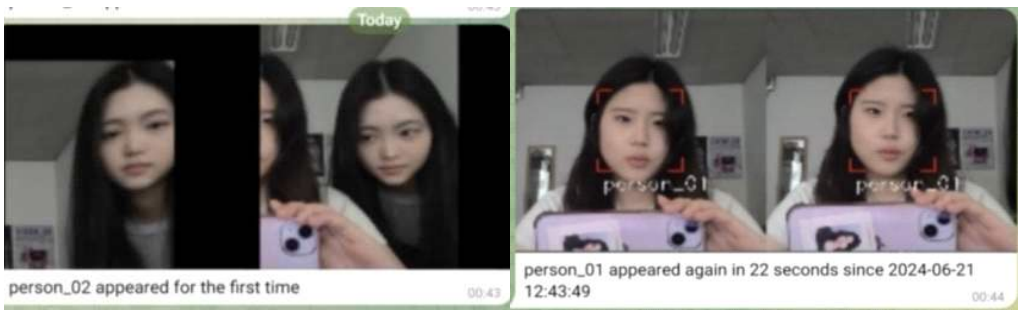
- 최종결과물 내용
  - 결과물 예시





->실패한 결과: 연결 과정은 나오지만 자동으로 종료되고 텔레그램 봇과의 연결 실패 후 에러발생

----------오류 해결 후----------



  - Github 사이트
    - https://github.com/Florakimm2/OpensourceTeamProject.git

- 프로젝트 수행 결과의 한계점
  - 개발 기술의 한계점
    - 이름을 직접 바꿔야 한다는 점에서 서비스의 타켓층으로 한 대상에게 적절성이 떨어진다

는 것을 인지함.

- 텔레그램 봇과의 연결이 불안정하다는 점을 고려한다면 다른 프론트(UI)를 개발하는 것을 더 나은 연결성과 결과를 보여줄 수 있을 것 같다.
- 현재 이 프로그램은 실시간으로 얼굴을 인식하는 매커니즘을 가지고 있다. 화면에 처음 인식된 사람은 unknown으로 인식되었다가 person0x로 변경된다. 이는 내가 아는 사람이 아니더라도 여러번 인식된다는 오점이 있는데 이를 보완하기 위해 사용자가 미리 아는 사람을 '사진'으로 등록하여 내가 등록한 사람이 아니면 그냥 모르는 사람으로 값을 받을 수 있도록하고, 사람이 인식될 때 rename을 하지 않아도 내가 아는 사람이면 바로 아는 사람임을 알 수 있도록 한다면 더 나은 프로그램이 될 것 같다.

○ 프로젝트 수행의 어려움 및 느낀 점

-개발시 어려움 및 느낀 점

- 패키지 설치시 반복되는 오류 -> conda install로 진행하여 패키지 설치가 필요함을 인지하고 conda install로 설치가 불가능한 opencv-python과 opencv-contrib-python은 pip install로 오류 해결
- runtime error발생 -> BRG에서 RGB로 바꾸는 코드 추가 및 수정하여 오류 해결.
- 각자 pc의 종류와 환경이 달라 실행 과정이나 코드 수정, 패키지 다운 등에서 차이가 있어 개인 pc 종류의 맞는 해결 방안들을 따로 모색해야 한다는 점이 어려웠다.
- 위와 같은 오류들을 오픈형 ai와 검색을 통해 해결하면서 많은 정보들을 습득할 수 있었고 이 과정들로 인해"오픈소스프로그래밍"해당 과목을 더 깊게 이해할 수 있었다.
- 버전에 따라 코드를 수정해야한다는 점에서 오픈소스를 활용하는 데에 어려움을 느낌 최신 버전을 수행 한 사람이 없는 경우 구글링도 어려웠고, 생성형 AI를 사용하고 싶어도 알려진 것들이 별로 없거나 영어로 나오는 내용이 많아 오류를 수정하는 데에도 어려움을 느낌.

-협력시 어려움 및 느낀 점

- 각자 일정을 조율하는 과정과 소통하는 과정에서도 오해가 생기고 어려웠다.
- 위와 같은 팀프로젝트 특성상 개인의 능력만으로 좋은 결과물을 도출하기에는 한계가 있고 이 한계를 해결하기 위해 소통이 무조건적으로 필요하다. 계속적인 소통과 이해, 배려 등을 통해 함께 더 성장할 수 있는 기회였다.