

# “IoT 기술을 이용한 스마트 주거환경 서비스”

- 풀커버 ( Full Cover) -

---

32151648 박동학

32150781 김승준

32152057 방승환

32155068 홍승기

2020. 05. 18

# Index

- 1. 프로젝트 개요**
- 2. 얼굴 인식 개선**
- 3. 물품 수령 및 하드웨어 설치**
- 4. 위험, 이상 탐지**
- 5. 단기 개발 목표**

# 프로젝트 개요

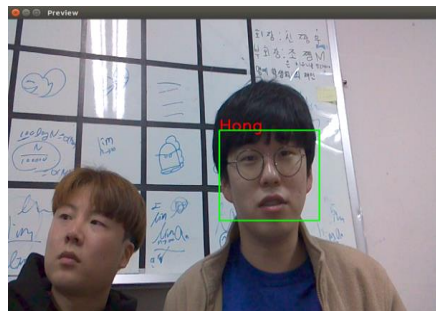


## 1. 차량 번호 인식 Python OpenCV

- ✓ 입주자 식별
- ✓ 주차공간 예약 안내



디스플레이  
사용자 GUI 제공



## 2. 안면 인식 Python OpenCV

- ✓ 입주자 식별
- ✓ 공동 현관문 열림



Jetson Nano  
- 중앙 처리 시스템



Raspberry Pi  
- 센서 데이터 처리



카메라



거리 측정 센서

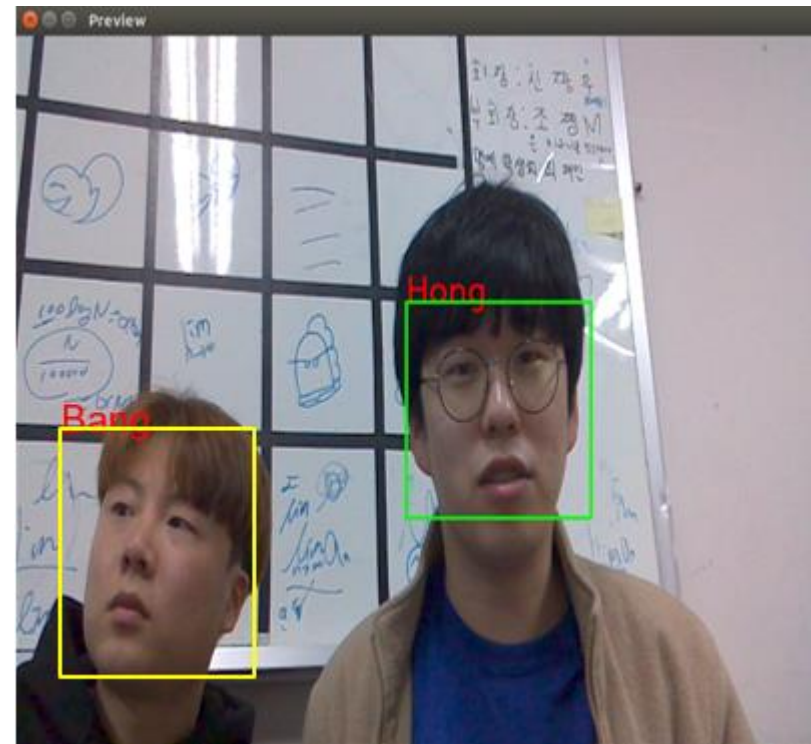
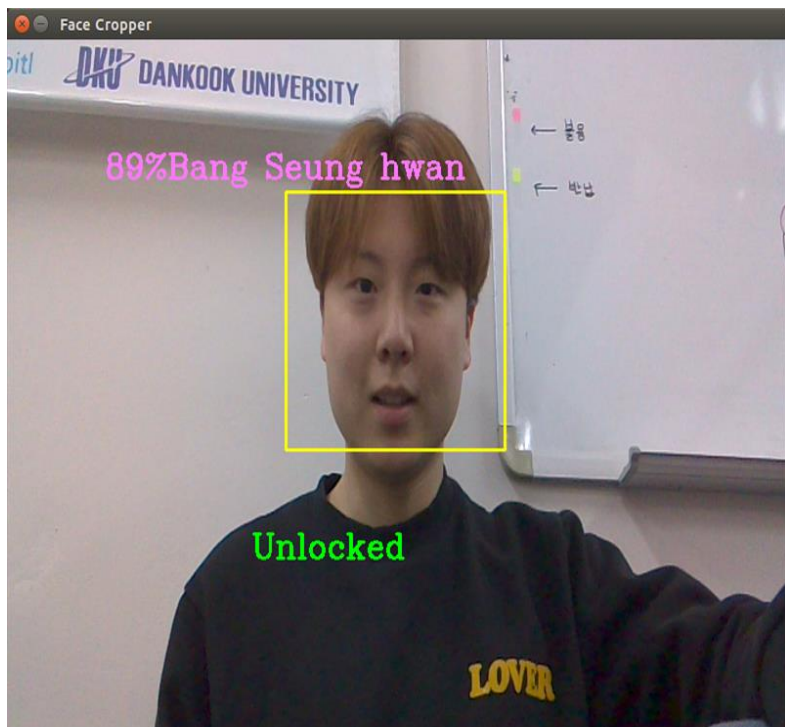


미세 먼지 측정 센서

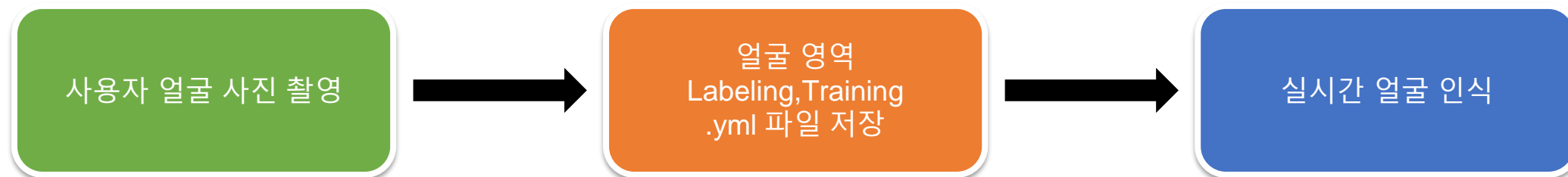
## 3. 이상 탐지 Isolation Forest

- ✓ 위험 상황 탐지
- ✓ 알림 요청
- ✓ 사고 처리 요청

## 개선 사항 : 단일 사용자 인식에서 다중 사용자 인식으로 개발 진행



## 얼굴 인식



Face\_Capture.py

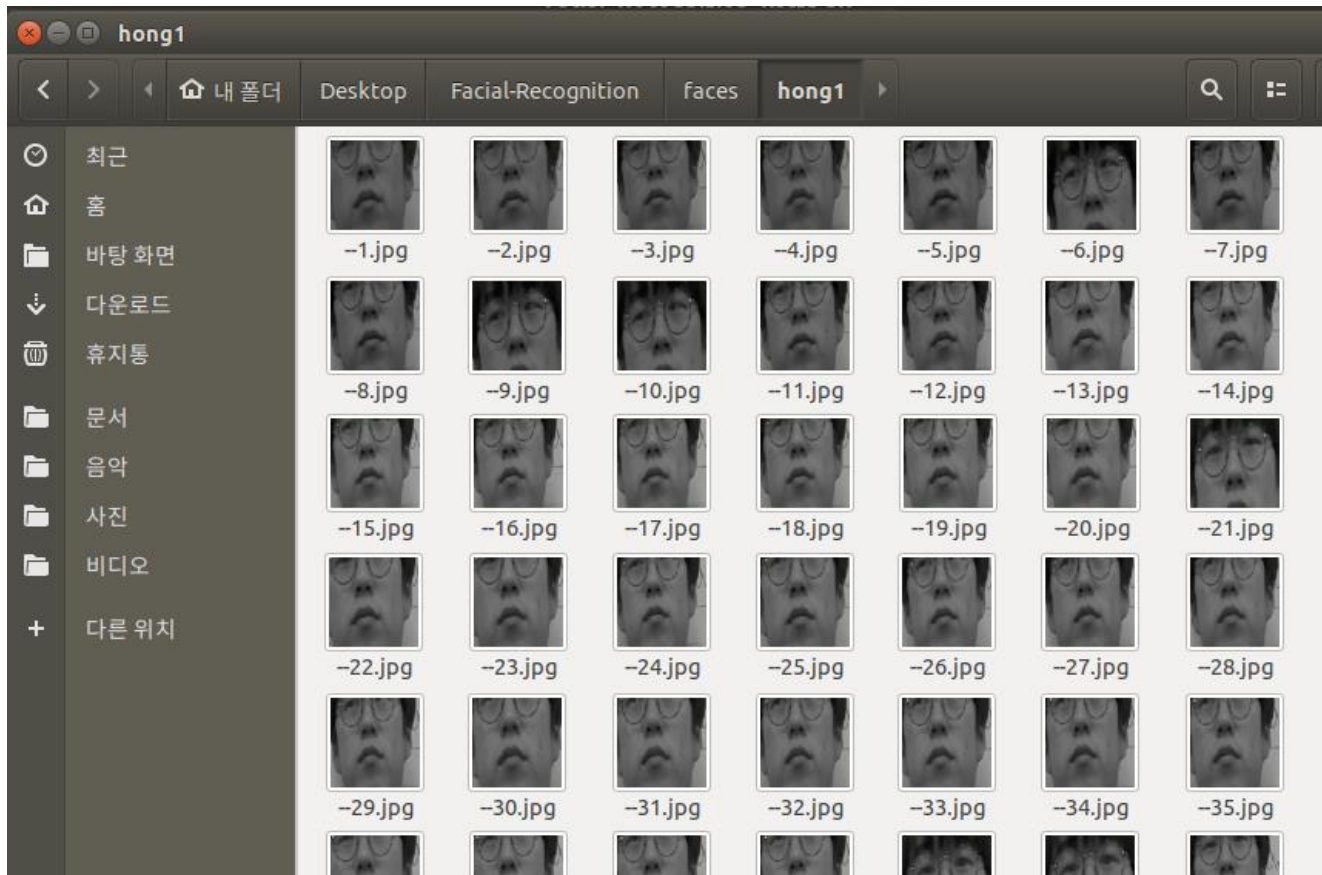


face\_training.py



face\_recog\_realtime.py

## Face\_capture.py



## Directory 구성

얼굴 사진 폴더

Faces

Label (User Name)

Hong

Image File

1.jpg  
2.jpg  
.

Dong

1.jpg  
2.jpg  
.



## Face\_training.py

```
(0, array([[388, 330, 170, 170]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/38.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/1.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/9.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/33.jpg', 'Hong')
(0, array([[365, 305, 185, 185]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/32.jpg', 'Hong')
(0, array([[382, 303, 181, 181]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/29.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/17.jpg', 'Hong')
(0, array([[382, 395, 198, 198]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/7.jpg', 'Hong')
(0, array([[361, 308, 209, 209]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/27.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/19.jpg', 'Hong')
(0, array([[364, 319, 196, 196]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/22.jpg', 'Hong')
(0, array([[383, 332, 166, 166]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/16.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/39.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/20.jpg', 'Hong')
(0, ())
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/18.jpg', 'Hong')
(0, array([[369, 327, 190, 190]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/21.jpg', 'Hong')
(0, array([[371, 327, 181, 181]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/34.jpg', 'Hong')
(0, array([[367, 308, 179, 179]], dtype=int32))
('/home/fullcover/Desktop/Facial-Recognition/faces/Hong/26.jpg', 'Hong')
(0, array([[387, 329, 170, 170]], dtype=int32))

```

- 얼굴 영역을 인식
- 영역에 존재하는 이미지를 recognizer를 통해서 학습
- 학습한 내용을 .yml에 저장

## 형식

(경로, label)  
(영역, DataType)

## Example

('/home/fullcover/Desktop/Facial- Recognition/faces/Hong/38.jpg', 'Hong')

(0, array([[388, 330, 170, 170]], dtype=int32))

## Face\_recog\_realtime.py

정확도

```
(0, 66.30830477746575)
(0, 64.45734821696624)
(0, 67.68581930564399)
(0, 64.23286697543719)
(0, 64.71226889803098)
(0, 63.92023926385611)
(0, 65.58454902445779)
(0, 64.98782918027763)
(0, 65.25418125118729)
(0, 67.00914941993862)
(0, 64.84995411493787)
(0, 67.1021755208254)
(0, 64.9170321936994)
(0, 66.81714813062543)
(0, 66.52373113475316)
(0, 64.46540613896025)
(0, 67.38971284323024)
(0, 63.683149007542596)
(0, 65.14290619007151)
(0, 66.32875194984476)
(0, 67.47567875613504)
(0, 66.76398632212855)
(0, 65.1970605890028)
(0, 65.14665784303699)
(0, 59.30282330095366)
(0, 67.50473114332117)
(0, 61.34626434673399)
(0, 66.53615365146534)
(0, 67.0455505235384)
(0, 67.9800698518331)
(0, 64.95396318872932)
(0, 68.45199083308816)
(0, 68.61041861980651)
(0, 67.22259559850072)
(0, 68.04895389761566)
(0, 68.8370072743015)
(0, 67.8317733752687)
```





## 앞으로의 개선 사항

- **다양한 배경으로 찍은 얼굴 사진 DataSet 준비 ( 팀원 中 2명 학습 예정)**
- **GUI에 실시간 인식 프로그램 탑재**
- **코드 최적화 진행**

## 하드웨어 추가 수령

: 무선통신을 위한 WIFI-Module

: 안테나

: 7인치 Jetson Nano Display



## Jetson Nano WIFI-Module 설치

문제점 : 주문한 랜 카드가 Jetson Nano OS Booting 과정에서 충돌 발생

2 MONTHS LATER

**huyijfmqp**

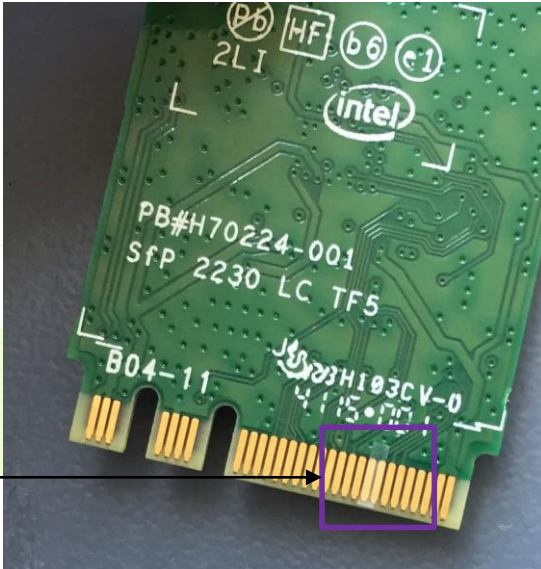
For those who have intel 8260 and still want it to work with nano, you can just mask out the I2C pins on the wifi card with tape.

The I2C signals use 8th and 9th pins on the back (counting from right to left).

[https://1drv.ms/u/s!ApXJrfCKTCq\\_gQAVFeDSbXuwPwa4](https://1drv.ms/u/s!ApXJrfCKTCq_gQAVFeDSbXuwPwa4)

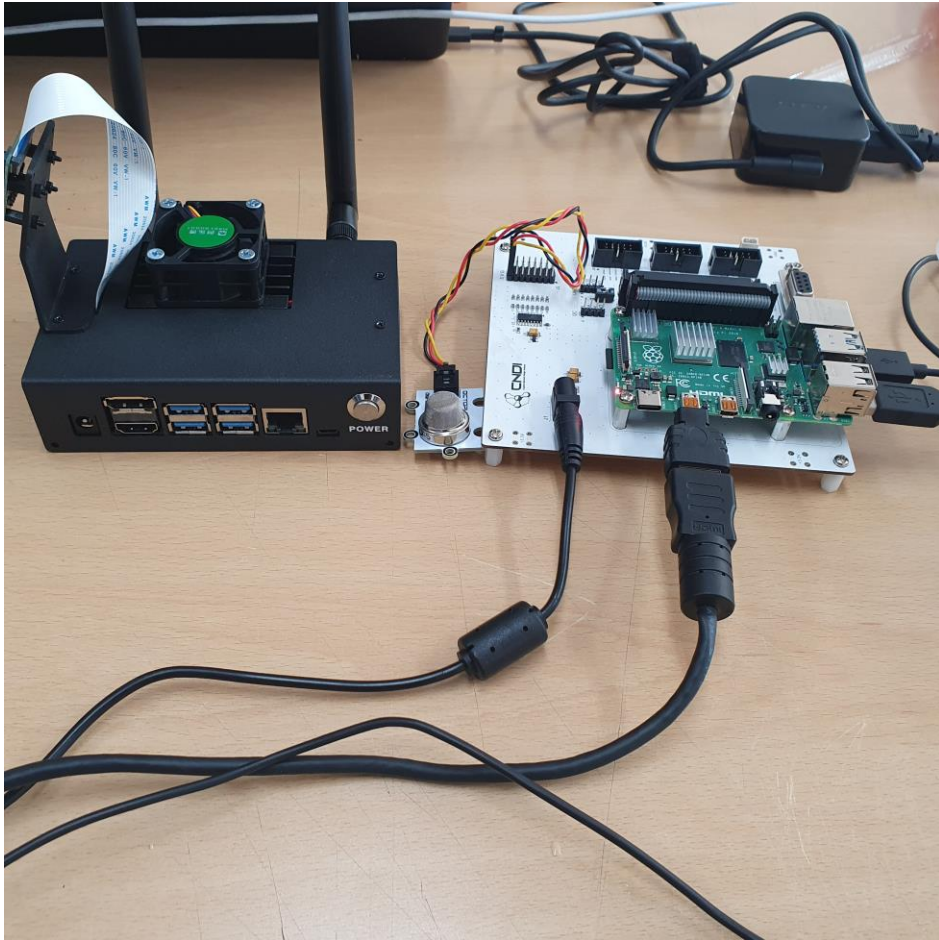
**Trumany** Employee

Hi huyijfmqp,  
Great post. Believe this can help others.



인터넷 검색 후 8번 9번 핀을 마스킹 작업 결정

## 하드웨어 완성 모습





## 위험, 이상 탐지

### 위험, 이상 탐지 – GasData

**위험 탐지 : Thresholding 기법을 이용한 가스 누수 탐지**

**이상 탐지 : isolation Forest Algorithm을 이용한 외부 공격, 서버 이상 탐지**



gasLeakage.py

센서 작동 및 데이터 Read Python Module



fire\_server.py

Server Communication Python Module



## 데이터 Format ( 0.5 sleep )

```
data = [[datetime.datetime.now(), getData]]
submission = pd.DataFrame(data)
submission.to_csv('./Gas_DataSet.csv', header = False, mode = 'a', index = False)
time.sleep(0.5)
```

TIMESTAMP	GAS_DATA
27:12.8	165
27:13.3	166
27:13.8	167
27:14.3	167
27:14.8	166
27:15.3	166
27:15.8	166
27:16.3	166
27:16.8	165
27:17.3	165
27:17.8	165
27:18.4	165
27:18.9	165

정상시

44:06.6	269
44:07.1	264
44:07.6	262
44:08.1	315
44:08.6	444
44:09.1	424
44:09.6	396
44:10.2	359
44:10.7	332
44:11.2	314
44:11.7	301
44:12.2	290
44:12.7	281

가스 감지시



Gas\_DataSet.csv

## 앞으로의 개선 사항

- 2~3 가지 센서 데이터 수집 ( Gas, 수분, 미세먼지 )
- 실시간 데이터 흐름 및 경고 GUI 개발
- 임계값 설정 및 isolation Forest Test (random으로 이상 데이터 생성)

- 1. 다중 사용자 인식에 있어서 DataSet 확보 및 정확도 개선 작업 필요**
- 2. 사용자에게 직관적인 사용법을 제시 할 수 있는 GUI 디자인 고려**
- 3. 센서 데이터를 효율적으로 관리할 수 있는 Module 개발**
- 4. 현실적인 한계를 보완할 필요성 제시**  
**[ ex. 번호 인식 실질 테스트, 생활 환경에서의 sensor 값]**

- 1. 지속적인 Sensor Data 수집**
- 2. GUI 보완 및 지속 개발**
- 3. Image Processing 파트에 있어 정확도 개선**