[별첨6]

DU-도전학기제 결과보고서

성명		**학번**	
단과대학		**학과(전공)**	
도전학기 과제명	**(한글)꼼짝마!** **(영문)Don't move!**		
지도교수 의견	가정 내 침입자를 탐지하여 탐지된 침입자를 스마트폰으로 실시간 통보하는 과제로 개발 초기에 아두이노를 사용하였으나 영상처리 성능의 문제로 라즈베리파이를 사용하였음. 각각의 기능은 잘 구현했으며, 개발기간 부족으로 인한 아쉬움이 있으나 영상처리와 안드로이드, 아두이노/라즈베리파이 개발에 대한 충분한 학습이 되었으리라 사료됨		
전공 인정 여부에 대한 학과장 의견	인정 교과목인 “오픈소스하드웨어”, “소프트웨어프로세스”, “인간컴퓨터상호작용”의 내용이 프로젝트 개발에 대부분 포함되어 있음. 작품의 기능이 다소 부족한 부분이 있지만 확장의 여지가 있음. 이 프로젝트를 통해 충분히 전공 역량을 키웠다고 판단됨.		

1. 도전과제의 목표

○ 도전과제의 최종목표

- 가정 내 침입자 탐지

->탐지를 위해 아두이노를 통해 카메라와 침입자 탐지를 위한 각종 센서의 정보를 더해서 인식률을 높일 생각입니다.

- 탐지된 침입자를 스마트폰으로 실시간 통보

->가정에 허용되지 않은 침입자가 탐지된다면, 스마트폰을 통해 실시간으로 정보를 제공합니다. 뿐만 아니라 단순한 통보가 아닌, 영상처리를 통한 침입자의 외관에 대한 정보를 추가로 제공할 예정입니다.

○ 세부목표

- ANDROID를 이용한 가정 내에 있는 아두이노 카메라 실시간 확인 기능

->아두이노에 와이파이 모듈을 탑재하여, 스마트폰과 계속해서 통신하면서 사용자가 원한다면 현재 집의 상태를 카메라로 확인할 수 있도록 할 예정입니다.

- 아두이노의 다양한 모듈을 이용한 인식률 강화
->기본적으로 카메라를 이용하지만, 그 한계를 극복하기 위해, 소리센서, 열감지센서 등을 추가로 부착하여

- 사용자를 이용한 다양한 편리성 제공
->누구나 가지고 있는 스마트폰을 이용해서 UI를 제공합니다. 언제 어디서나 집안의 상황을 확인할 수 있고, 데이터에 연결되어 있으면, 침입 정보를 제공 받을 수 있습니다. 뿐만 아니라, 아두이노의 회전 모듈을 이용해서, 집 안의 상황을 360도 확인 할 수 있도록 시스템을 구축할 예정입니다.

2. 도전 과제 내용

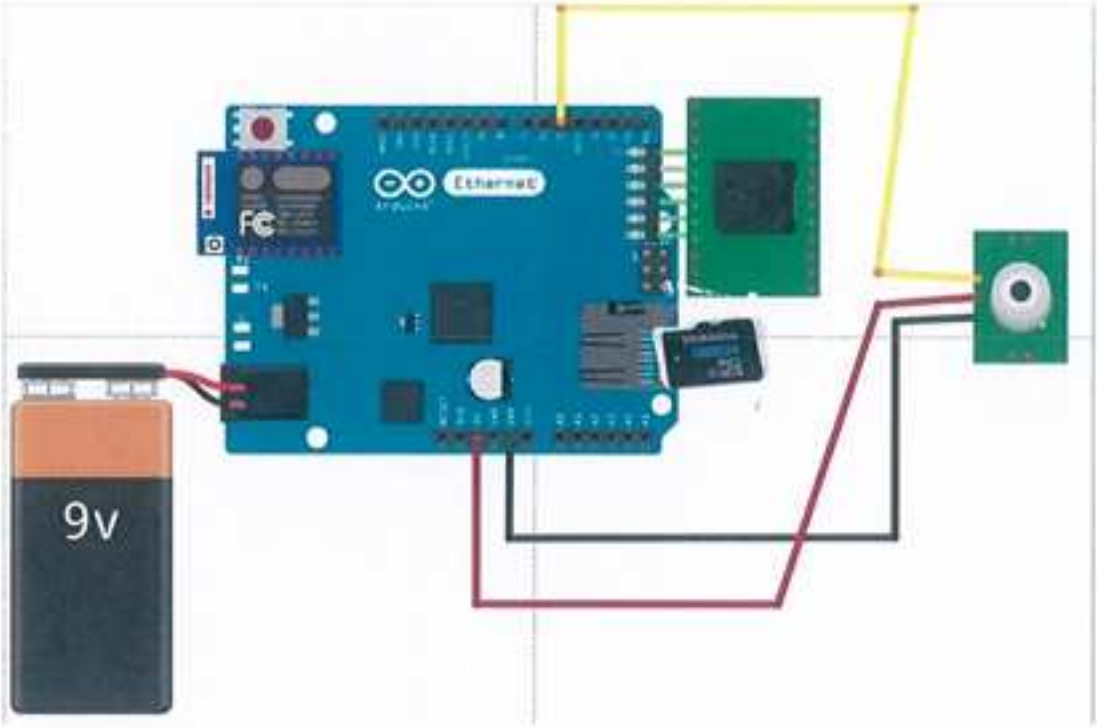
○ 효율적인 침입 탐지를 위한 시스템 개발
- 전체적인 시스템 구성

->아두이노에 부착된 카메라 모듈을 통한 영상을 감지하기 위해 OEPNCV를 이용합니다. OPENCV를 통해 침입자를 감지하는 알고리즘을 제작하여 침입자를 탐지합니다. 추가적으로 아두이노에 부착된 센서들을 이용해 침입자 탐지율을 올립니다. 만약 침입자가 탐지된다면 안드로이드 스마트폰을 이용해서 사용자에게 그 정보를 알려줍니다.

○ 사용자의 부담 최소화를 위한 설계
- OPENCV를 통해 탐지된 침입자의 키와 복장 등의 외관정보를 계산해서 알려주도록 설계
- 침입자가 감지된 순간부터 동영상 촬영을 시작해서 동영상의 캡쳐사진을 사용자에게 전송
= 촬영된 동영상은 공유기를 통한 내부 스토리지에 저장될 수 있도록 설계

○ 사용자를 위한 편의성 제공
- 단순히 침입탐지용도로 사용하는 것을 넘어서 아두이노 스피커 등을 활용해서 집안의 애완동물과 소통할 수 있는 부가기능 제공
- CCTV의 연결과 해지 그리고 동기화 되었다는 스마트폰 푸시 알림 서비스 제공

-아두이노 모형



1) 아두이노와 카메라를 연결
2) 아두이노와 PIR 모션센서를 연결
3) 아두이노에는 미리 작성한 소스코드를 컴파일 시켜 넣어 놓음
3) 작동시키기 위해 9V 배터리를 통해 전원을 공급함

-공유기 배선

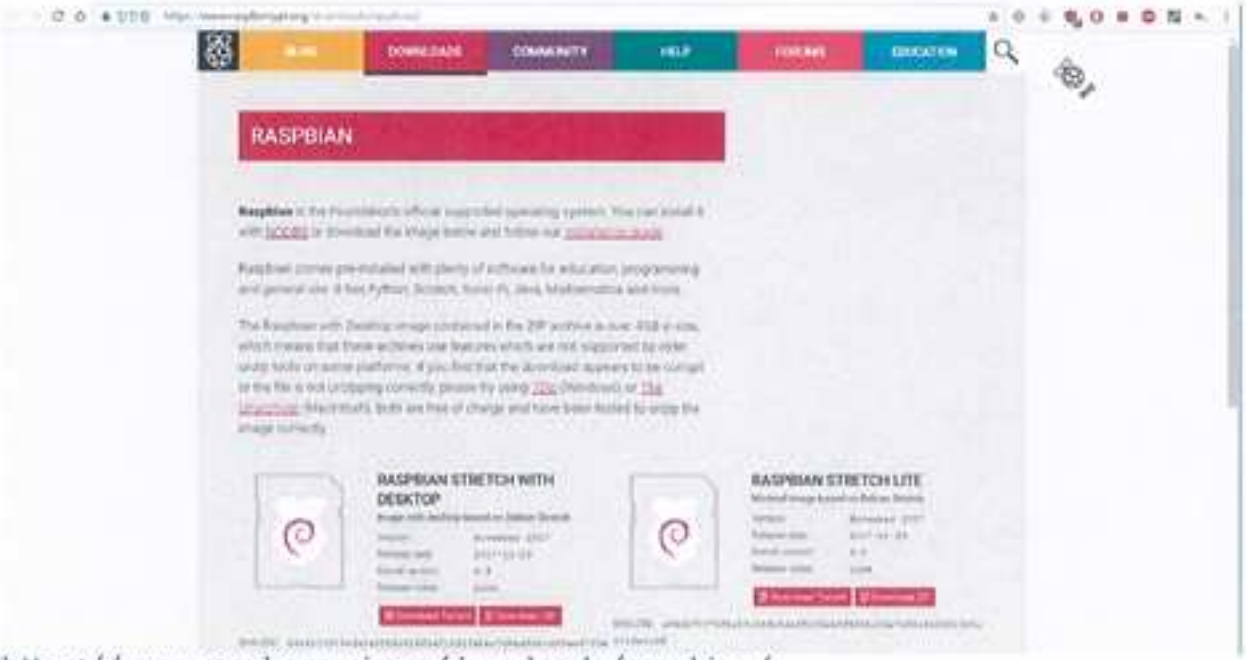
모뎀과 공유기를 연결하였고, PC에 공유기를 물려 테스트 환경을 구축하였습니다.

수정버전-라즈베리파이를 추가 사용한 침입탐지 시스템 개발

1) 개발환경 구성

-라즈비안 설치

· 라즈비안 이미지파일 다운로드



https://www.raspberrypi.org/downloads/raspbian/

에서 데스크탑 버전으로 다운로드하였음

· 다운로드한 이미지파일을 Micro SD카드에 쓰기

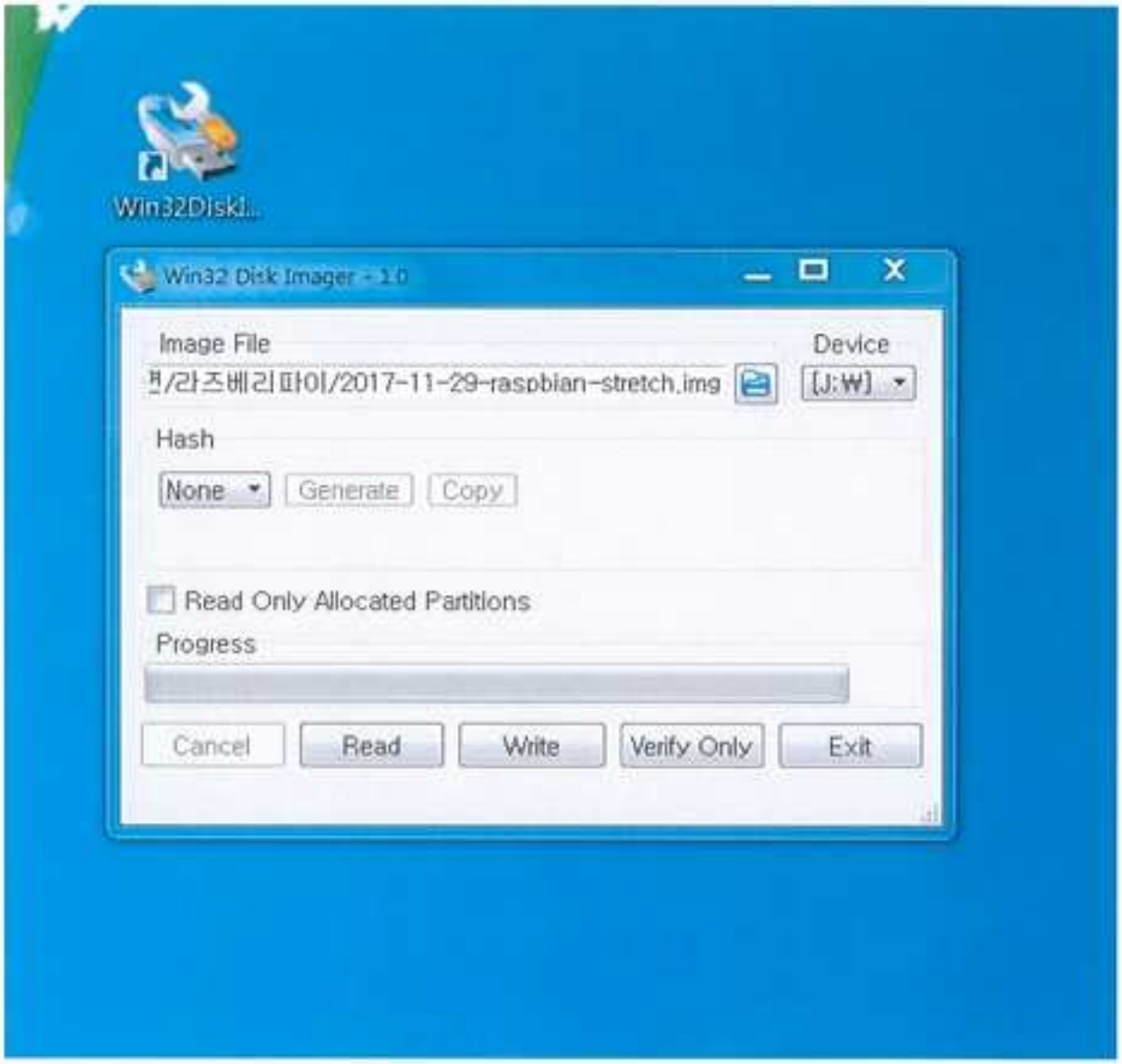
PC에 독자적인 리더기가 없다면, 따로 판매하는 Micro SD 리더기가 필요하다.

라즈베리파이는 Micro SD카드를 지원함

· SD카드에 이미지파일을 쓰기 위해 전용 툴 다운로드 필요

라즈베리파이 이미지파일을 SD카드에 쓸 수 있게해주는 툴
(Win32 Disk Imager)

· Win32 Disk Imager 실행 화면



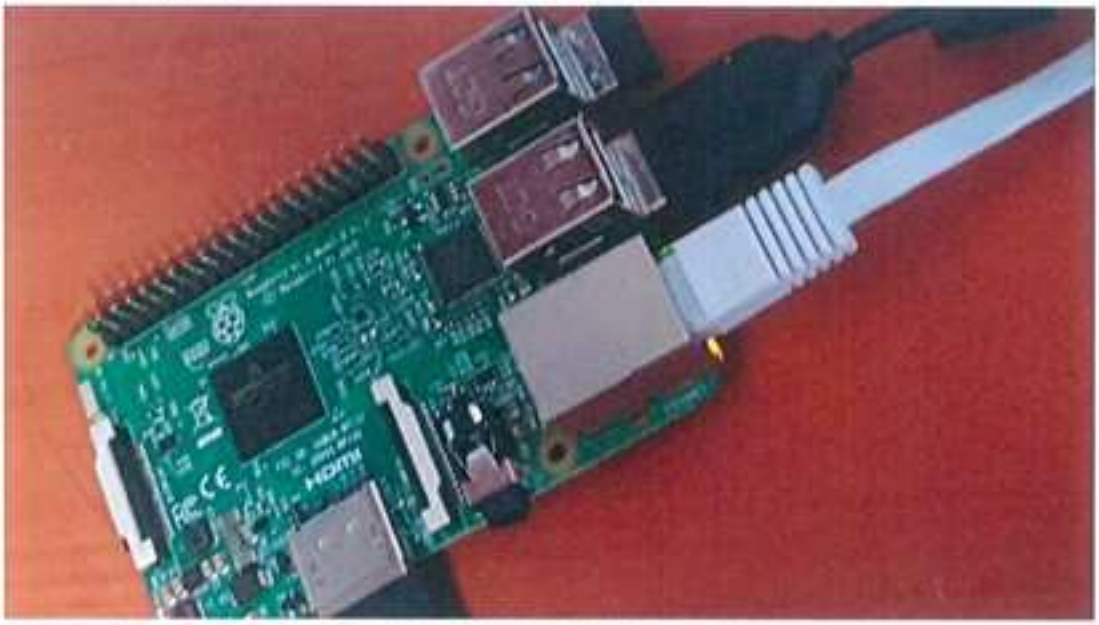
Win32 Disk Imager를 실행시킨 화면으로 이미지파일의 경로를 찾아서 넣어준 뒤, Micro SD카드의 드라이브 이름을 확인하고 Read 버튼을 눌러주면 라즈베리파이가 Micro SD카드에 자동으로 쓰여진다. 시간이 꽤 오래걸리는 작업이다.

-네트워크 구성

· 공유기

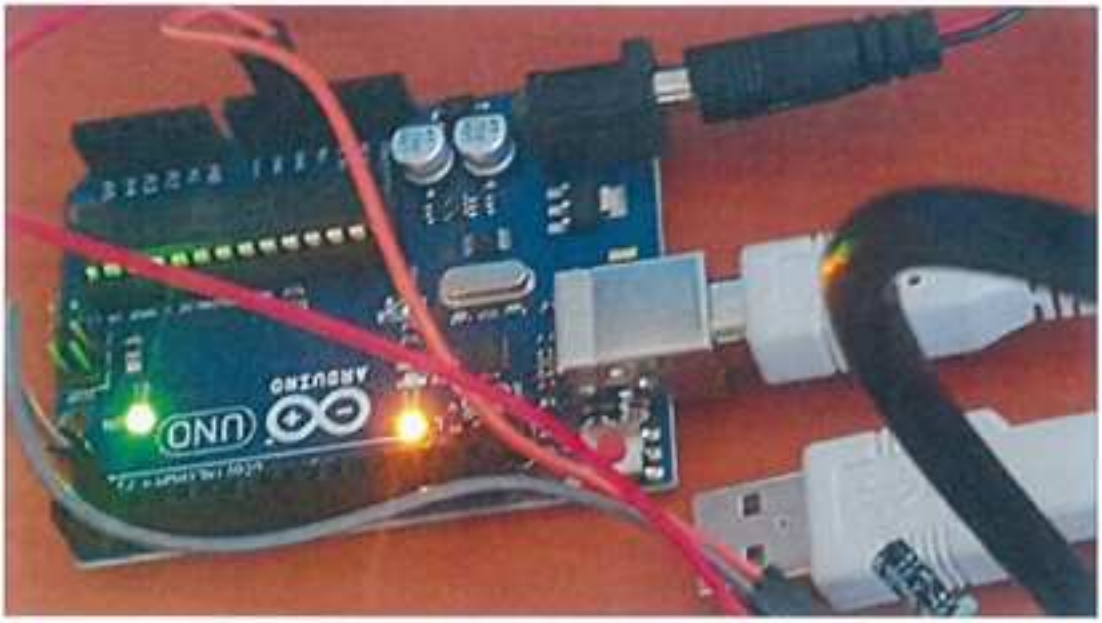
오른쪽 맨 아래 랜선이 외부에서 들어오는 공유기와의 데이터 연결선
오른쪽 맨 위 랜선이 PC와 연결되는 랜선
오른쪽 중간 랜선이 라즈베리파이와 연결되는 랜선

· 라즈베리파이



공유기에 물려있는 랜선을 가져와서 유선연결이 되어있는 상태
와이파이로도 인터넷에 접속이 가능하지만, 심각한 성능의 저하로 인해 너무나 불편해서 개발 중에는 유선랜을 사용함

· 아두이노



라즈베리파이와 USB 연결이 되어 있는 상태
아두이노또한 ESP8266 와이파이 모듈로 인터넷에 연결이 가능함

하지만, 라즈베리파이에 훨씬 못 미치는 성능으로 인해, 아두이노에서 굳이 인터넷 연결을 할 필요가 없다고 판단함

위 모듈은 아두이노 ESP8266 모듈로 아두이노 UNO에 적절한 배선을 통해 Wifi 연결을 가능하게 해 주는 모듈임(현재 사용하지 않음)

-초기설정

· 최초 부팅상태

개발하는 동안, USB 키보드가 없어서 우여곡절 끝에 VNCSERVER에 연결시켰음
최초에 라즈베리파이 연결시 화상키보드 또한 설치되어있지 않았음

Micro SD카드에 화상키보드 설치 명령어가 담긴 텍스트파일을 넣음
한줄 한줄 copy and paste를 통해 화상키보드를 설치함

설정변경을 위해서는 GUI화면 상단 메뉴에서 터미널을 실행, sudo raspi-config 명령어를 입력
Change user password에서 내가 사용할 패스워드를 2차례 입력하여 적절히 설정해야 함
Localization options에서 지역정보(로케일), 타임존 등을 설정할 수 있음
지역정보의 경우 en_GB(UTF8), en_US(UTF8), ko_KR(UTF8), ko_KR(EUC)를 선택하였고, 기본으로 ko_KR(UTF8)을 설정해 주었다.
시간정보의 경우 ASIA-Seoul을 설정하였다.

Interfacing options은 라즈베리파이와 통신하는 기기 및 프로토콜 방법들에 대한 허용 여부를 설정할 수 있도록 하는 기능임, 원격접속을 할 것이기 때문에 SSH와 VNC기능만 enable로 하고, 나머지는 모두 disable로 함

-VNCSERVER

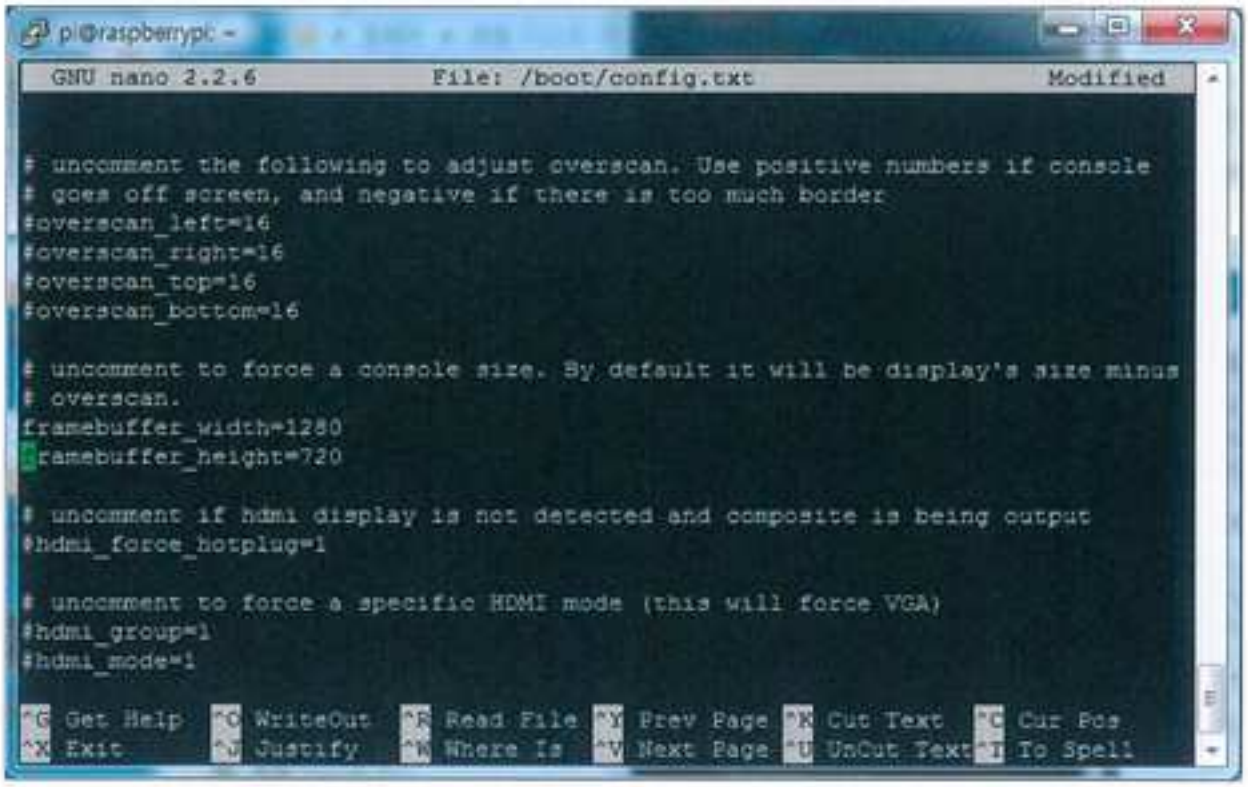
PC에서 원격으로 라즈베리파이에 접속해서 개발을 할 수 있도록 도와주는 원격 프로그램이며 USB키보드가 없는 지금환경에서 반드시 필요한 도구이다.

이전에 설치해두었던 화상키보드를 통해 마우스로 일일이 클릭하면서 진행
(VNC Viewer 실행 이후에는 화상키보드 쓸 일이 거의 없음)

· x11vnc 설치
pi@raspberrypi:~ $ sudo apt-get install x11vnc xinetd

· /boot/config.txt 파일을 편집기로 열어서 해상도 수정

pi@raspberrypi:~ $ sudo nano /boot/config.txt



· 바뀐 설정을 적용하기 위해 라즈베리파이 재부팅

pi@raspberrypi:~ $ sudo reboot

VNC Server 프로그램을 실행한다.

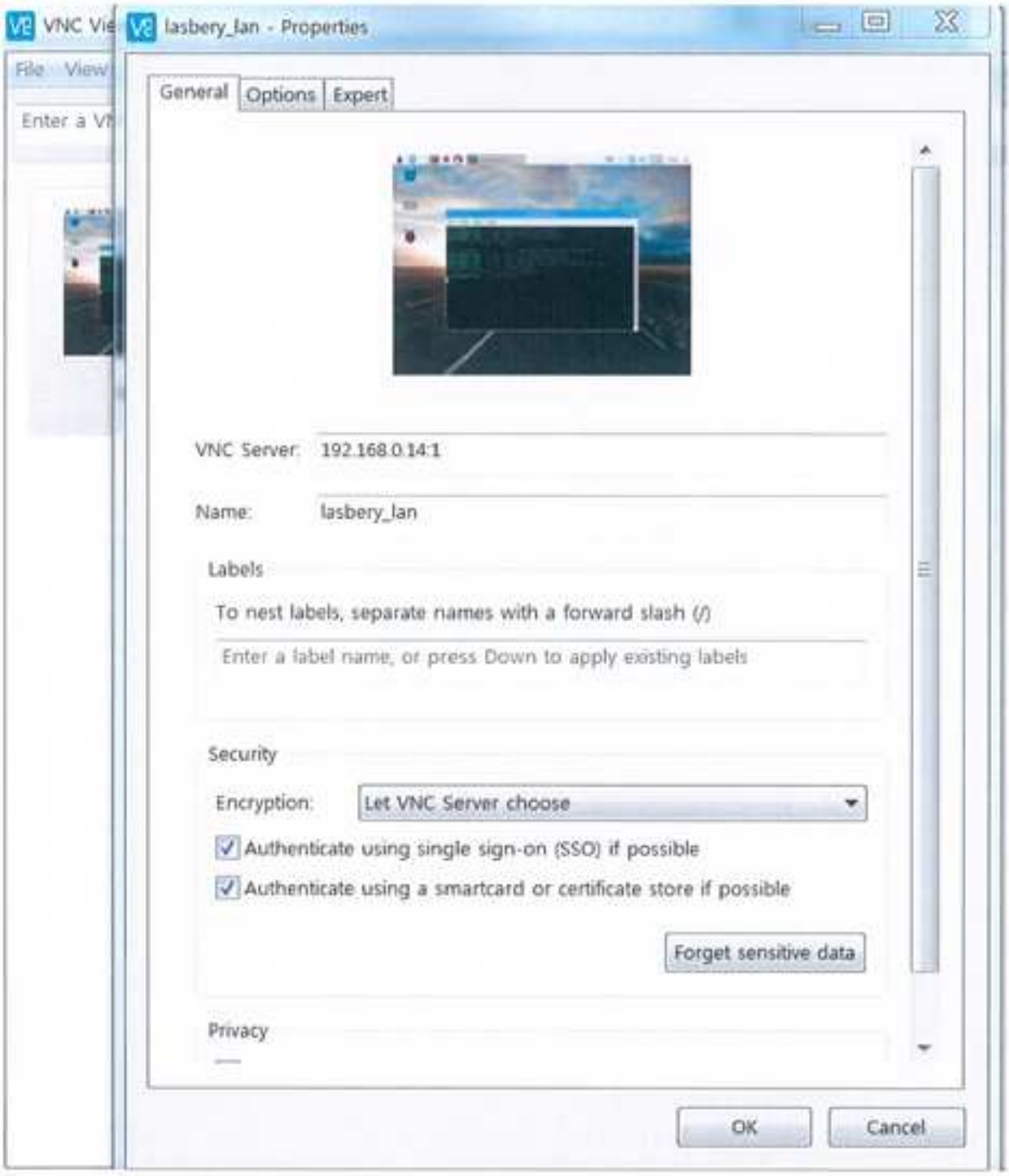
pi@raspberrypi:~ $ x11vnc

· VNC클라이언트 설치

PC상에는 VNC Viewer 프로그램을 실행해서 원격 제어가 가능함

VNC Verwer는 https://www.realvnc.com/download/viewer/ 에서 다운로드

· VNC 프로그램을 실행하고 라즈베리파이 IP로 접속



PC와 라즈베리파이는 같은 네트워크상에 묶여있기 때문에 내부IP로 연결이 가능함

· VNC viewer에서 사전에 설정해둔 비밀번호를 치고 접속에 들어감

· VNC viewer에서 원격 접속에 성공한 상태

2) motion
-침입감지를 위한 라즈베리파이 전용 패키지
-웹 서버의 구현도 같이 담당함

motion 패키지를 설치
sudo apt-get install motion

또는 Motion을 컴파일하기위해 필요한 패키지들 설치
pi@raspberrypi:~ $
sudo apt-get install libavutil-dev libavcodec-dev libavformat-dev libswscale-dev libjpeg-dev libmp-dev pkg-config libtool automake autoconf build-essential

Motion 소스 코드를 다운로드
pi@raspberrypi:~ $
wget https://github.com/Motion-Project/motion/archive/release-4.0.1.tar.gz

- Motion 소스 코드의 압축을 해제
pi@raspberrypi:~ $ tar xvf release-4.0.1.tar.gz

- Motion 소스 코드 폴더로 이동
pi@raspberrypi:~ $ cd motion-release-4.0.1

- configure 스크립트를 생성
pi@raspberrypi:~ $ autoreconf -fiv

- Motion 컴파일 환경을 설정
pi@raspberrypi:~ $ ./configure

- Motion의 10초 걸린 이벤트를 수정하기위해 alg.c 파일을 수정
pi@raspberrypi:~ $ sudo nano alg.c

- Ctrl+W을 눌러 아래줄을 찾아서 10을 2로 바꿔줌
#define ACCEPT_STATIC_OBJECT_TIME 2 /* Seconds */

Motion의 8프레임 걸린 이벤트를 수정하기위해 motion.c 파일을 수정
pi@raspberrypi:~ $ nano motion.c

Ctrl+W를 눌러 아래줄을 찾아서 8을 0으로 바꿔줌
cnt->moved = 0;

- 오류없이 컴파일 하기위해 Header 파일들을 복사
pi@raspberrypi:~ $ sudo cp /opt/vc/include/interface/vcos/pthreads/*
/opt/vc/include/interface/vcos/

pi@raspberrypi:~ $
sudo cp /opt/vc/include/interface/vmcs_host/linux/vchost_config.h
/opt/vc/include/interface/vmcs_host/

- Motion을 컴파일
pi@raspberrypi:~ $ make -j3

- Motion을 설치
pi@raspberrypi:~ $ sudo make install

Motion 환경설정 파일을 만듦
pi@raspberrypi:~ $ sudo cp /usr/local/etc/motion/motion-dist.conf
/usr/local/etc/motion/motion.conf

- 설정해서 모션을 감지하고 저장한 사진을 이메일로 보내도록 Motion 환경설정 파일을 수정
pi@raspberrypi:~ $ sudo nano /usr/local/etc/motion/motion.conf

daemon = off 에서 on 으로 설정
daemon - 옵션을 프로그램을 백그라운드에서 별도로 돌리기 위한 옵션 설정

webcam_localhost = on 에서 off 으로 설정

-웹캠을 로컬에서만 사용할지를 설정하는 것으로 우리는 다른 컴퓨터 및 장비로부터 접속할 예정이기에 Off로 설정

- motion 패키지 구동
pi@raspberrypi:~ $ sudo motion

3) mutt

-이메일을 보내기 위한 패키지

· 패키지정보를 갱신

pi@raspberrypi:~ $sudo apt-get update

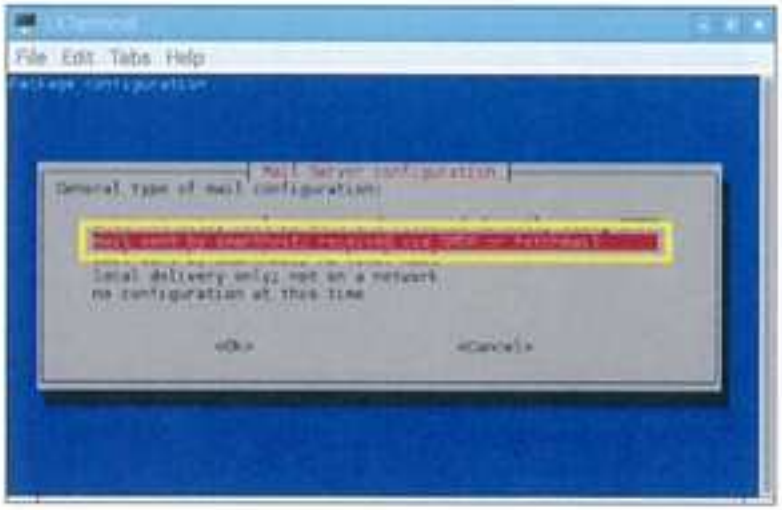
· mutt 이메일 클라이언트와 exim4 메일 서버를 설치

pi@raspberrypi:~ $ sudo apt-get install mutt exim4

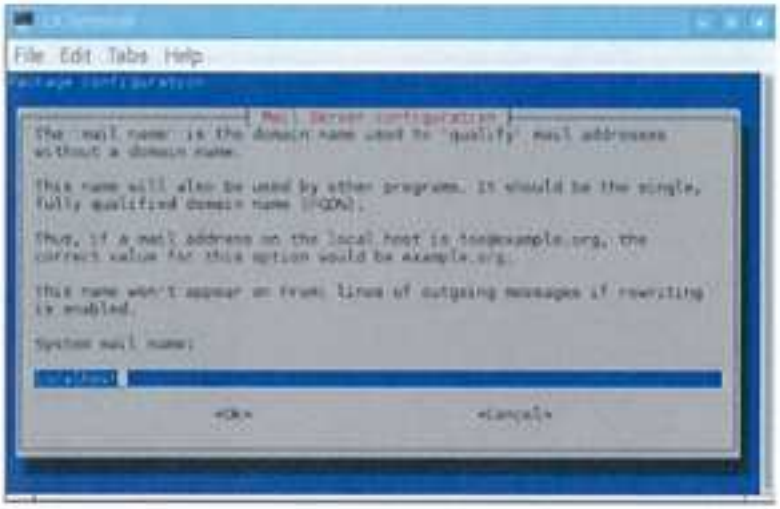
· exim4 메일 서버의 환경을 설정

pi@raspberrypi:~ $ sudo dpkg-reconfigure exim4-config

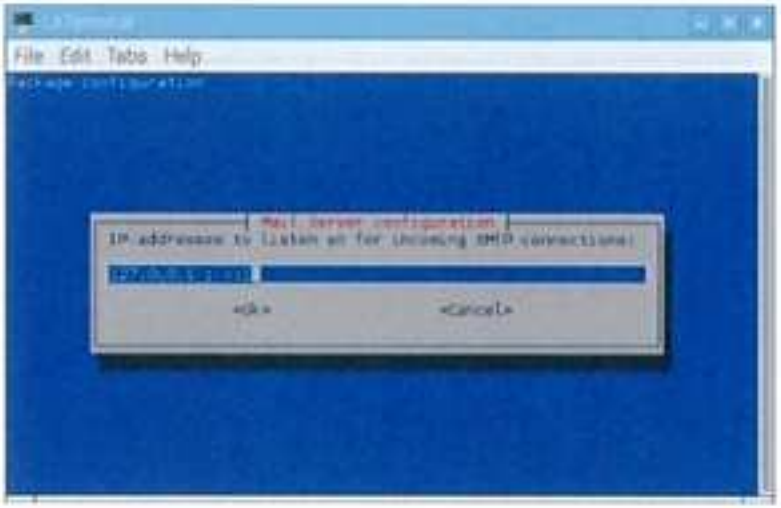
· 지메일 SMTP 서버를 이용하기 위해 "mail sent by smarthost; received via SMTP or fetchmail"를 선택



· "System mail name"에 "locahost"를 입력. Enter키를 눌러 다음으로 넘어감.

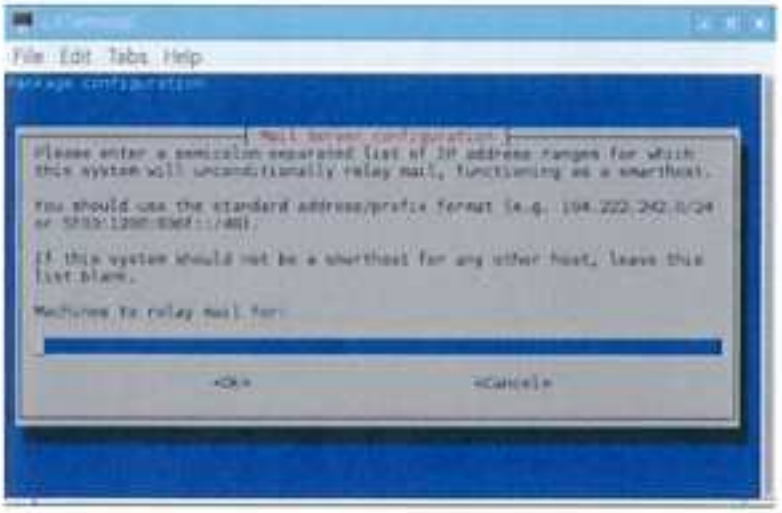


· "IP-addresses to listen on for incoming SMTP connections"이 "127.0.01 : ::1"로 되어 있는지 확인 후 Enter키를 눌러 다음으로 넘어감.



· "Other destinations for which mail is accepted"를 공백으로 만듦. Enter키를 눌러 다음으로 넘어감.

· "Machines to relay mail for"가 공백으로 되어 있는지 확인. Enter키를 눌러 다음으로 넘어감.



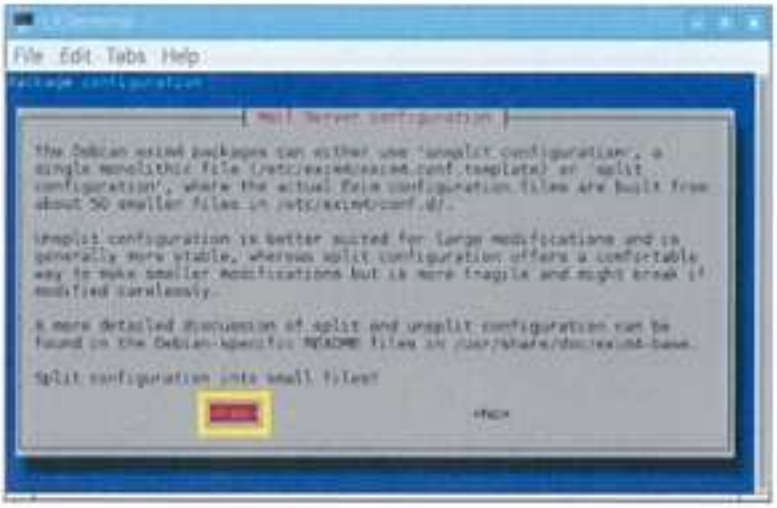
· "IP address or host name of the outgoing smarthost"에 "smtp.gmail.com::587"을 입력함. Enter키를 눌러 다음으로 넘어감.

· "Hide local mail name in outgoing mail?"에 "No"를 선택.

· "Keep number of DNS-queries minimal (Dial-on-Demand)?"에 "No"를 선택.

· "Delivery method for local mail"을 "mbox format in /var/mail/"로 선택.

· "Split configuration into small files?"에 "Yes"를 선택.



· "Root and postmaster mail recipient"에 공백으로 되어 있는지 확인.
Enter키를 눌러 환경설정을 종료.

· exim4 메일 서버가 지메일 SMTP 서버에 로그인하기 위한 계정정보를 입력
pi@raspberrypi:~ $ sudo nano /etc/exim4/passwd.client

· passwd.client 파일 마지막 줄에 아래 줄을 추가. your_gmail_id에 사용할 Gmail 아이디, your_gmail_password에 사용할 Gmail 패스워드를 입력 ctrl+X를 눌러서 종료할 수 있음

*.google.com:your_gmail_id@gmail.com:your_gmail_password
(개인정보라서 입력하지 않음)

· exim4 메일 서버가 지메일 SMTP 서버의 IPv6주소와 연결되지 않는 문제를 방지.
pi@raspberrypi:~ $
sudo nano /etc/exim4/conf.d/main/02_exim4-config_options

· "02_exim4-conifg_options 파일"의 마지막 줄에 아래 줄을 추가.
disable_ipv6 = true

· mutt 이메일 클라이언트가 지연없이 이메일을 보내기 위해 환경을 설정.

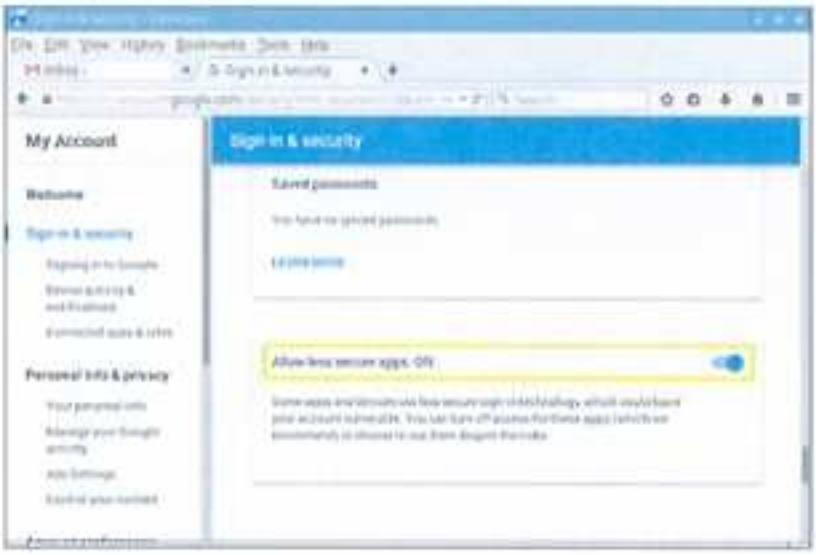
```
pi@raspberrypi:~ $ nano ~/.muttrc
```

· .muttrc 파일 마지막 줄에 아래 줄을 추가.

```
set sendmail_wait = -1
```

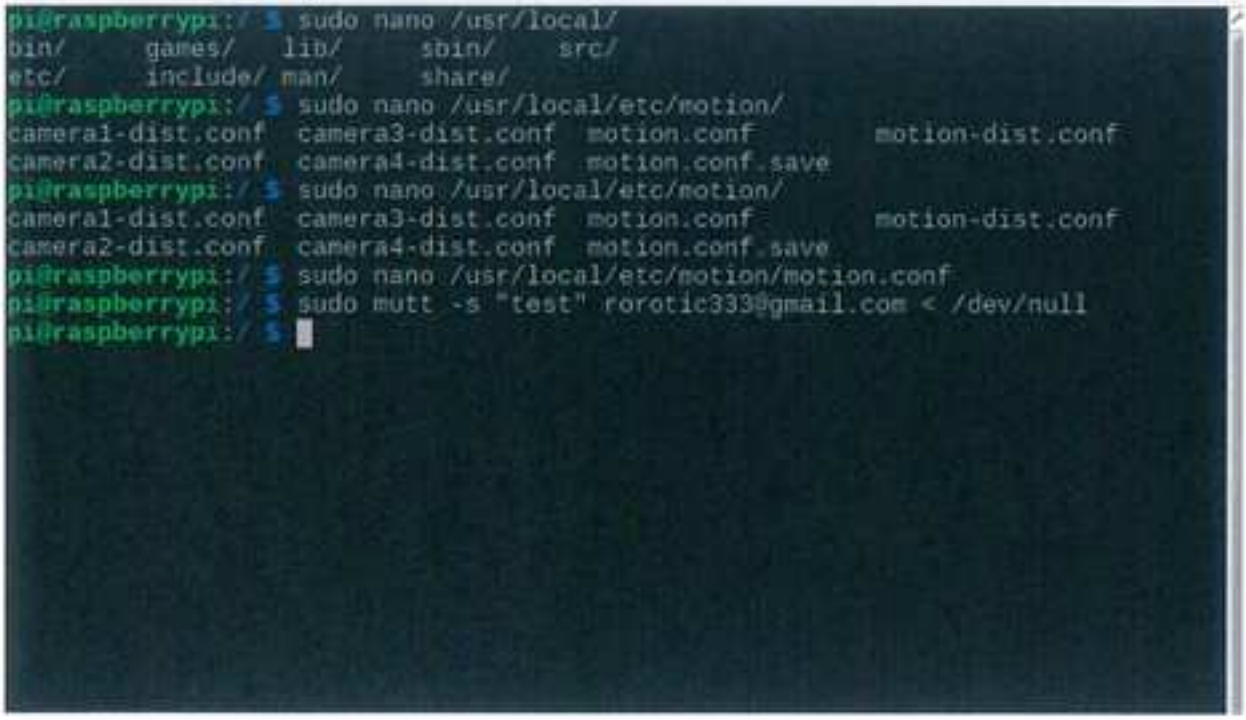
· exim4같은 보안 수준이 낮은 앱에서 Gmail SMTP 서버를 이용해서 이메일 보내기 위해 Gmail 설정하는 방법.

· "보안 수준이 낮은 앱 허용(Allow less secure apps)"을 "ON"으로 선택.



· mutt 이메일 클라이언트를 이용해서 이메일이 정상적으로 가는지 테스트

pi@raspberrypi:~ $ sudo mutt -s "test" rorotic333@gmail.com < /dev/null



· 정상적으로 메일이 수신된 모습.

※ 참고한 프로젝트(회사에서 진행)- 3D 신체검사기 비전관련 프로젝트

(1) 개요

신체 단면의 굴곡을 표시해서 의료의 목적으로 사용할 수 있는 3D 신체검사기 머신

(2) 동작과정

C#, C++을 이용 .basler Camera 5M pixcel 120fps 급 사용

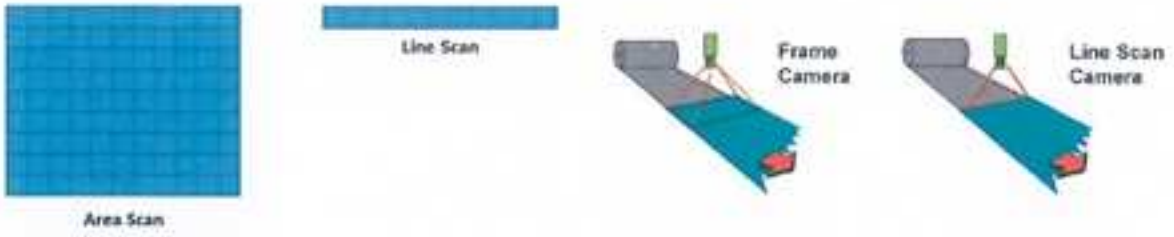
모터에 basler카메라와 레이저를 부착시킴

부착된 카메라가 모터에 의해 밑으로 내려오면서 신체에 쏘아진 레이저의 굴곡을 촬영함

모터가 일정한 속도로 밑으로 내려오면서 카메라로 촬영한 굴곡의 데이터를 화면에 뿌려 주고 그 데이터는 축적됨

기능은 Line Scan Camera와 동일한 기능을 가짐

※ Line Scan Camera

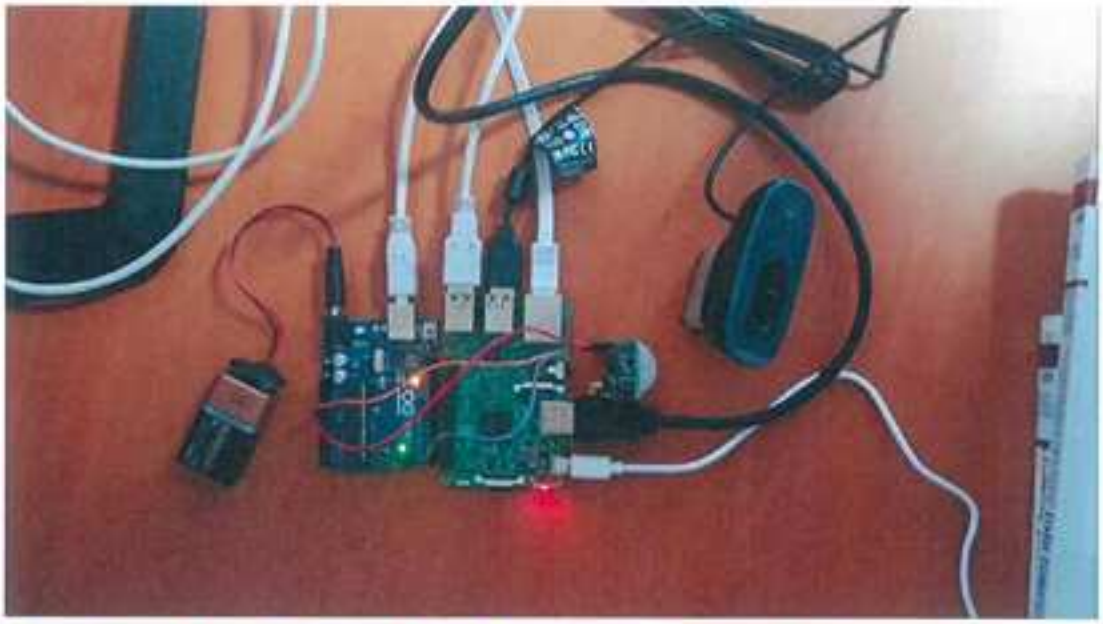


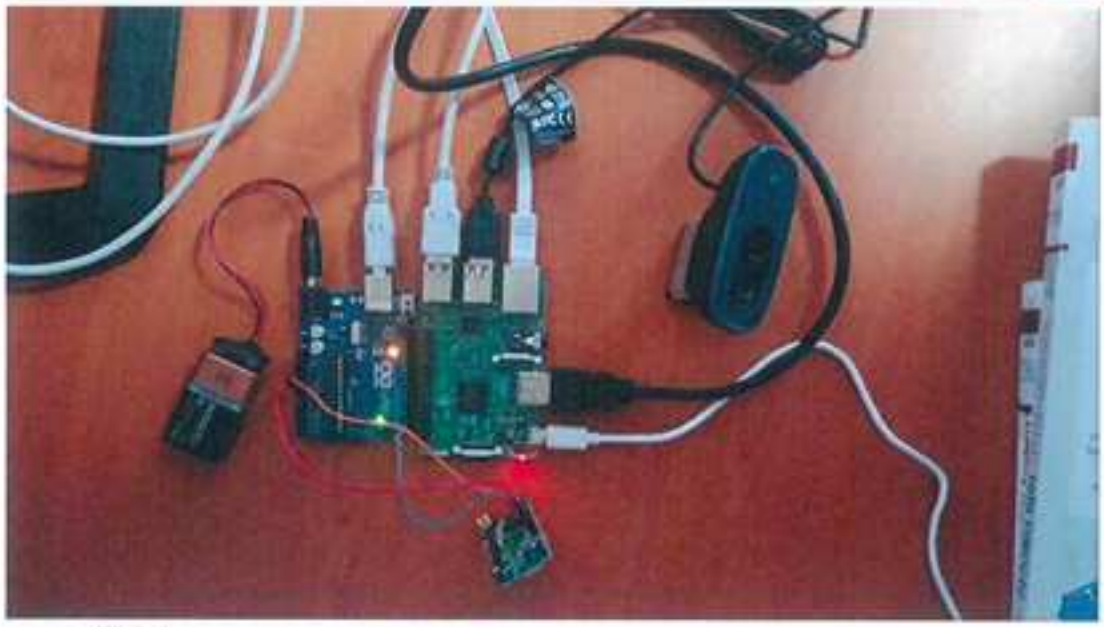
Line scan 카메라는 개체 또는 카메라 자체가 움직이면서 촬영하며, 정지된 화상을 촬영하는데 적합함.
Area scan 카메라보다 컨베이어 벨트와 같이 움직이는 개체를 촬영하는데 적합하고, Line Scan 카메라는 픽셀의 수에 따라 높이에 대한 제약이 있는 Area scan 카메라에 비해 Line scan 카메라는 높이에 대한 제약이 없다는 장점을 가짐.
이러한 특징으로 인해, Line scan 카메라를 사용하면 카메라의 수를 줄일 수 있는 장점을 가짐.

3. 도전 과제의 성과

라즈베리파이를 이용한 침입탐지 시스템 개발

1) 완성도 사진





· 아두이노

9V 배터리를 이용해서 독립 전원 구성

모션 센서를 이용해서 라즈베리파이에게 모션감지정보 전달

라즈베리파이와는 usb로 연결되어 있음

· 라즈베리파이

휴대폰 충전기와 같은 규격의 전원 공급 케이블이 필요함
독자 전원이 반드시 필요하며, 임의로 연결 해제시에는 강제로 부팅이 종료됨
아두이노와는 USB로 연결되어서 데이터를 주고받음
USB포트는 4개가 달려있지만, 웹캠의 전력을 받혀주는데도 모자라서 독자전원을 공급해주는 허브가 반드시 필요함(전력 공급 부족으로 인한 웹캠 오작동)
사진에는 가려서 나오지 않지만, 무선마우스를 연결해놓은 상태

· 웹캠

아두이노 ov2640 카메라 성능이 너무 좋지않아(2M 급, 초점수동조절) USB카메라 추가로 사용
라즈베리파이에 USB로 연결되어 있으며, 외부전원이 공급되는 허브에 물려있지 않아서 전력부족으로 인해 가끔 오작동할 때가 있음

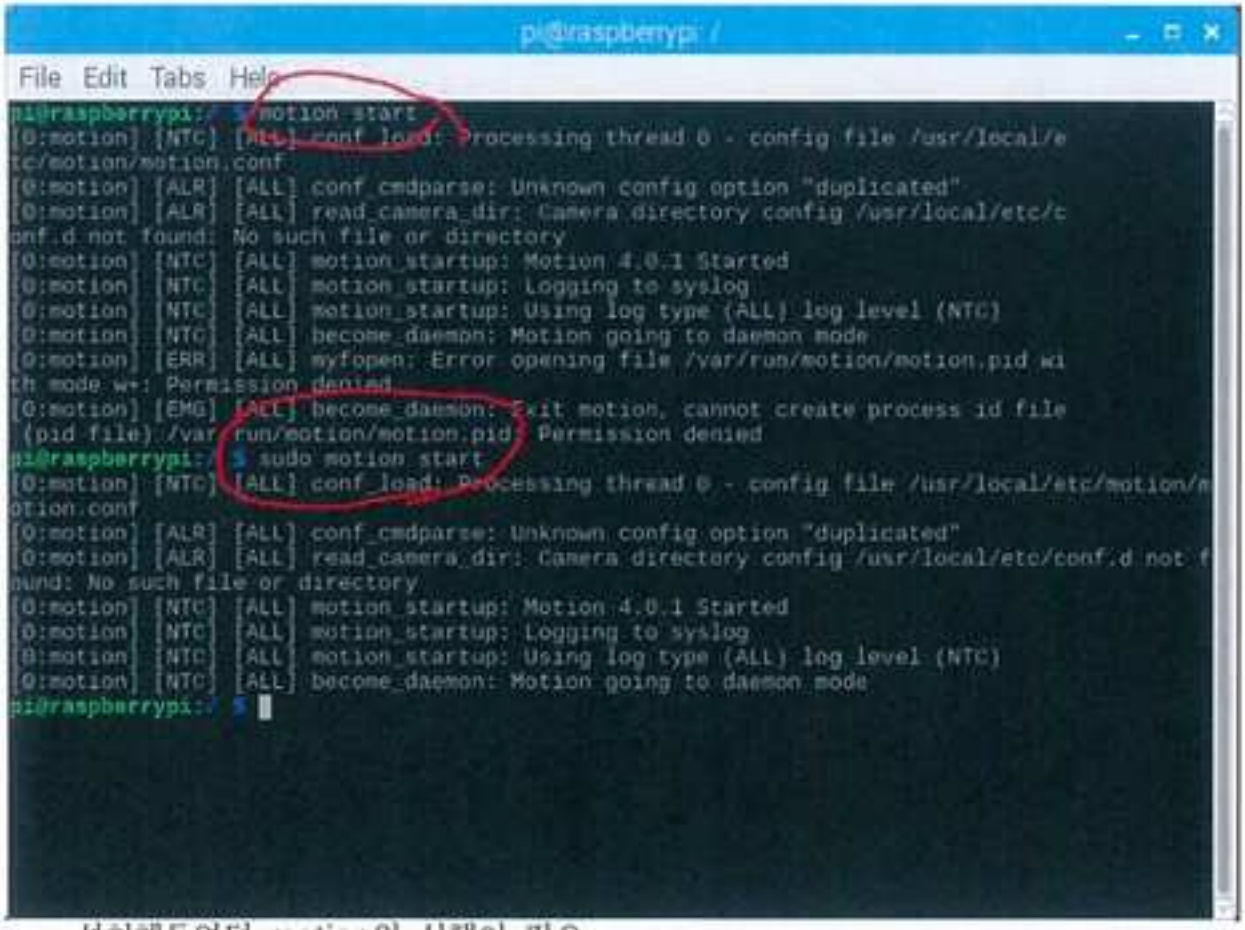
※ (유전원 허브)

왼쪽에는 USB 케이블로 연결되는 부분이며, 오른쪽은 220V전원이 공급되는 코드가 있음

2) 실행순서

-라즈베리파이 초기 설정

(1) motion 실행



· 설치해두었던 motion의 실행이 필요

· 첫 번째 라인에서는 관리자 명령어인 sudo가 빠져있어서 에러가 난 모습

· 다음 입력에는 sudo motion start를 통해 정상적으로 motion이 실행된 것을 알 수 있다.

(2) 웹 브라우저에서 실행한 원격 접속 모습

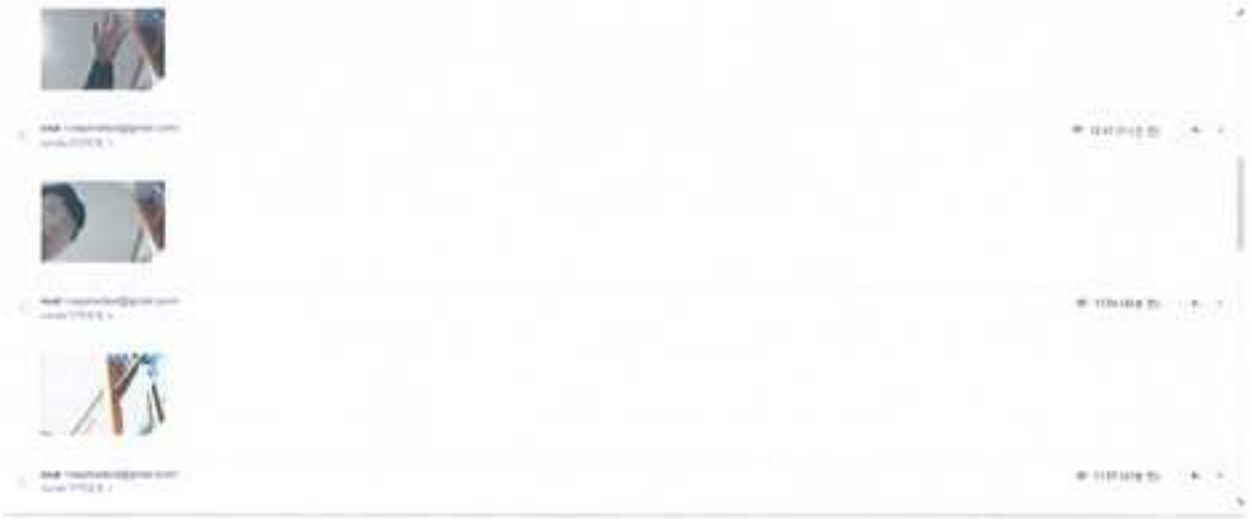
· 웹 브라우저에서도 동일 네트워크 안이기 때문에 ip주소대역과 포트번호를 통해서 간단하게 motion 웹 서버에서 접속이 가능함

· 접속하게 되면 실시간으로 카메라에서 뿌려주는 영상을 확인할 수 있음

· 움직임이 관찰될 시 redbox가 쳐지고 이때의 사진은 라즈베리파이 내부 sd카드에 저장됨

· 또한 저장된 사진은 이메일로 자동 전송됨 (백업기능과 통지기능을 동시에)

라즈베리파이에서 자동으로 전송된 이미지파일들.



· 웹 서버상에서 조정할 수 있는 옵션들

<- back

Camera 0

- daemon = on
- process_id_file = /var/run/motion/motion.pid
- setup_mode = off
- camera_name = (not defined)
- logfile = (not defined)
- log_level = 6
- log_type = all
- videodevice = /dev/video0
- v4l2_palette = 17
- input = -1
- norm = 0
- frequency = 0
- rotate = 0
- width = 640
- height = 480
- framerate = 10
- minimum_frame_time = 0
- netcam_url = (not defined)
- netcam_userpass = (not defined)
- netcam_keepalive = off
- netcam_proxy = (not defined)
- netcam_tolerant_check = off
- rtsp_uses_tcp = on
- mmalcam_name = (not defined)
- mmalcam_control_params = (not defined)
- auto_brightness = off
- brightness = 0
- contrast = 0
- saturation = 0
- hue = 0
- power_line_frequency = -1
- roundrobin_frames = 1
- roundrobin_skip = 1
- switchfilter = off
- threshold = 100
- threshold_tune = off
- noise_level = 24
- noise_tune = on
- despeckle_filter = EedDl
- area_detect = (not defined)
- mask_file = (not defined)
- smart_mask_speed = 0
- lightswitch = 0

모두 쓰는 기능은 아니며, 일부 필요한 기능 외에는 손대지 않는 것이 좋다.
width : 웹상에서 보여줄 가로 픽셀 너비
height : 웹상에서 보여줄 세로 픽셀 길이
threshold : 침입감지를 위해 현재사진과 다음사진간의 픽셀간 차이의 정도를 나타냄
이 값이 낮을수록 민감하게 침입을 탐지하며, 이 값이 높다면 장면이 크게 바뀌지 않는 이상 침입했다고 판단하지 않을 것이다.

Daemon: Motion이 백그라운드에서 동작할 것인지 안 할것인지 구분하는 상태
현재 Motion은 백그라운드에서 동작하고 있음

· 웹 서버상에서 조정할 수 있는 옵션들 2

- minimum_motion_frames = 1
- pre_capture = 0
- post_capture = 0
- event_gap = 60
- max_movie_time = 0
- emulate_motion = off
- output_pictures = best
- output_debug_pictures = off
- quality = 75
- camera_id = 0
- picture_type = jpeg
- ffmpeg_output_movies = off
- ffmpeg_output_debug_movies = off
- ffmpeg_timelapse = 0
- ffmpeg_timelapse_mode = daily
- ffmpeg_bps = 400000
- ffmpeg_variable_bitrate = 0
- ffmpeg_video_codec = mpeg4
- ffmpeg_duplicate_frames = off
- use_extpipe = off
- extpipe = (not defined)
- snapshot_interval = 0
- locate_motion_mode = on
- locate_motion_style = redbox
- text_right = %Y-%m-%dWn%T-%q
- text_left = (not defined)
- text_changes = on
- text_event = %Y%m%d%H%M%S
- text_double = off
- exif_text = (not defined)
- target_dir = /home/pi/Detection
- snapshot_filename = %v-%Y%m%d%H%M%S-snapshot
- picture_filename = %v-%Y%m%d%H%M%S-%q
- movie_filename = %v-%Y%m%d%H%M%S
- timelapse_filename = %Y%m%d-timelapse
- ipv6_enabled = off
- stream_port = 8081
- stream_quality = 50
- stream_motion = off
- stream_maxrate = 20
- stream_localhost = off
- stream_limit = 0
- stream_auth_method = 0
- stream_authentication = username:password
- stream_preview_scale = 100
- stream_preview_newline = off
- webcontrol_port = 8080

quality: 사진의 품질 정도. 높을수록 용량이 커짐
target_dir: Micro_SD카드가 저장될 경로
picture_filename: 사진파일이 저장될 이름.
%v: 연속되는 침입의 한 장면

%y: 년 %m: 월 %d: 일 %h: 시 %m: 분 %s: 초

%q: 장면내에서 촬영되는 사진의 넘버링 개수

· 웹 서버상에서 조정할 수 있는 옵션들 3

- stream_motion = off
- stream_maxrate = 20
- stream_localhost = off
- stream_limit = 0
- stream_auth_method = 0
- stream_authentication = username:password
- stream_preview_scale = 100
- stream_preview_newline = off
- webcontrol_port = 8080
- webcontrol_localhost = off
- webcontrol_html_output = on
- webcontrol_authentication = username:password
- track_type = 0
- track_auto = off
- track_port = (not defined)
- track_motorx = 0
- track_motorx_reverse = off
- track_motory = 0
- track_motory_reverse = off
- track_maxx = 0
- track_minx = 0
- track_maxy = 0
- track_miny = 0
- track_homex = 128
- track_homey = 128
- track_iomojo_id = 0
- track_step_angle_x = 10
- track_step_angle_y = 10
- track_move_wait = 10
- track_speed = 255
- track_stepsize = 40
- quiet = on
- on_event_start = (not defined)
- on_event_end = (not defined)
- on_picture_save = sudo mutt -s "Motion Detected New New" -a "%f" -- rcrotic333@gmail.com< /dev/null
- on_motion_detected = (not defined)
- on_area_detected = (not defined)
- on_movie_start = (not defined)
- on_movie_end = (not defined)
- on_camera_lost = (not defined)
- video_pipe = (not defined)
- motion_video_pipe = (not defined)
- camera = No threads
- thread = No threads
- camera_dir = /usr/local/etc/conf.d

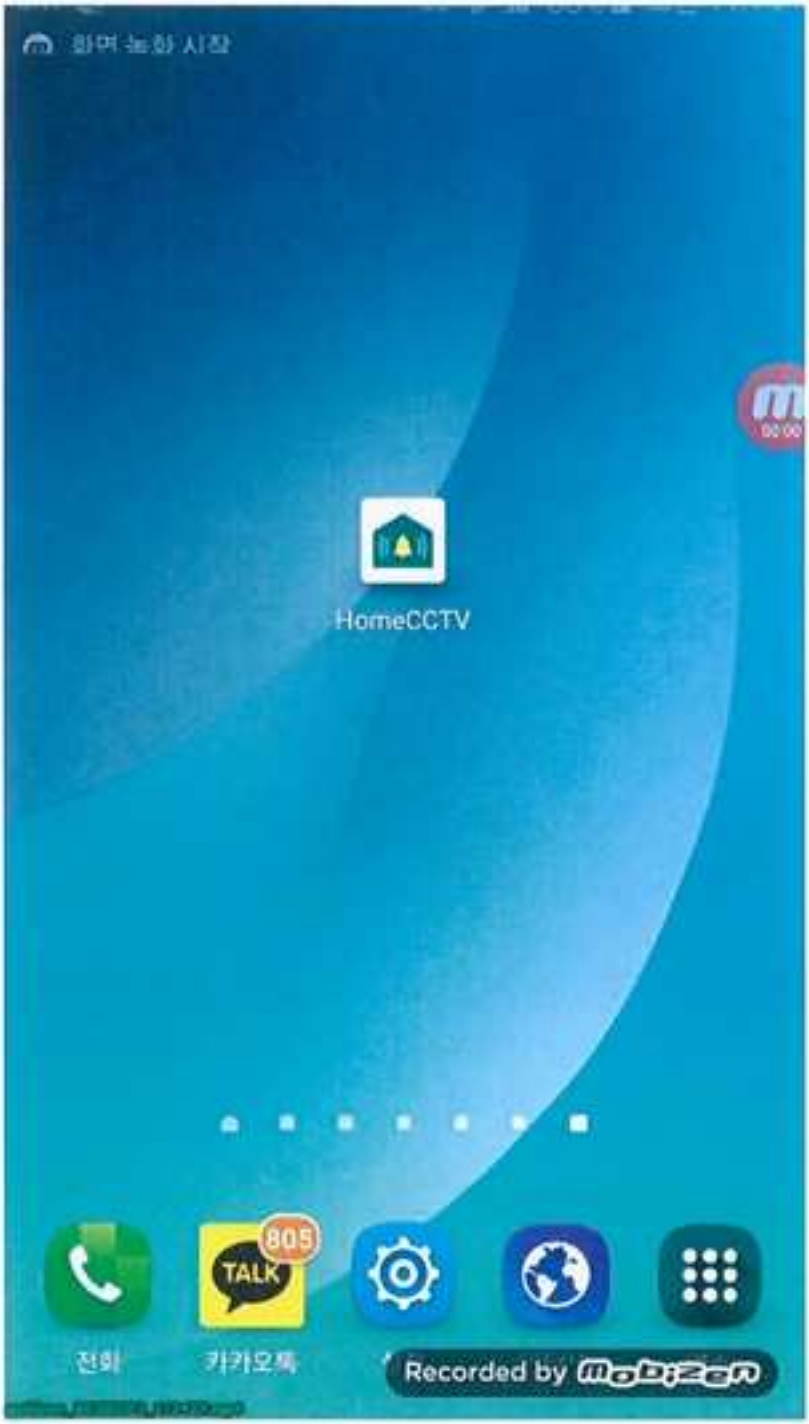
<- back

stream_localhost: 스트림으로 진행되는 웹캠영상을 내부에서만 볼 수 있게 설정할 것인지 정하는 것.
즉 라즈베리파이 내의 웹브라우저에서만 사용가능하게 설정하는 것

webcontrol_localhost: 현재 페이지의 설정에 대한내용을 내부에서만 사용할 수 있게 설정하는 것.
즉 라즈베리파이 내의 웹브라우저에서만 사용가능하게 설정하는 것

on_picture_save: 사진이 저장되면, 저장된 사진을 이메일로 보내주는 부분 받는 사람을 여러명 추가할 수도 있고, 내용을 추가할 수 있고, 제목을 변경할 수 있다.

(3) 스마트폰에서 어플리케이션에서 확인한 모습

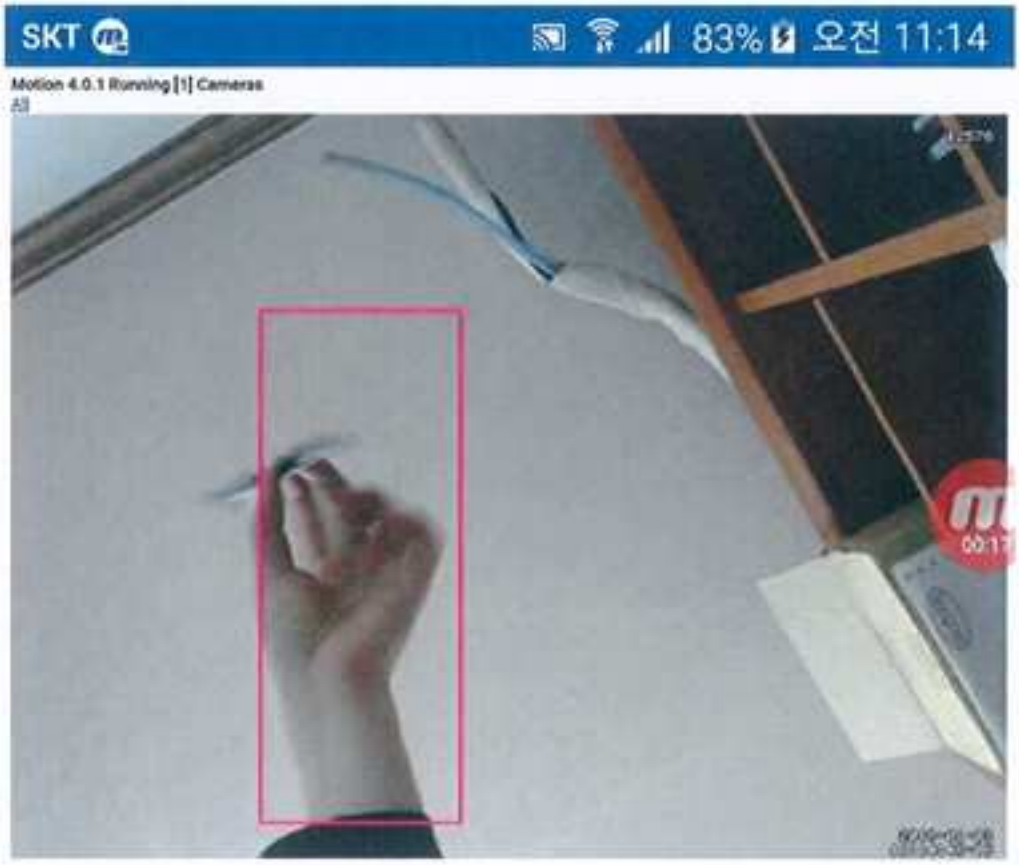


생성된 안드로이드 어플리케이션 모습

(3) 스마트폰에서 어플리케이션에서 확인한 모습

생성된 어플리케이션 최초 실행모습
실시간으로 영상을 보여줌

(3) 스마트폰에서 어플리케이션에서 확인한 모습



카메라 촬영 영역에 손가락과 볼펜을 통해 움직임이 제대로 감지되고 있음을 보여줌

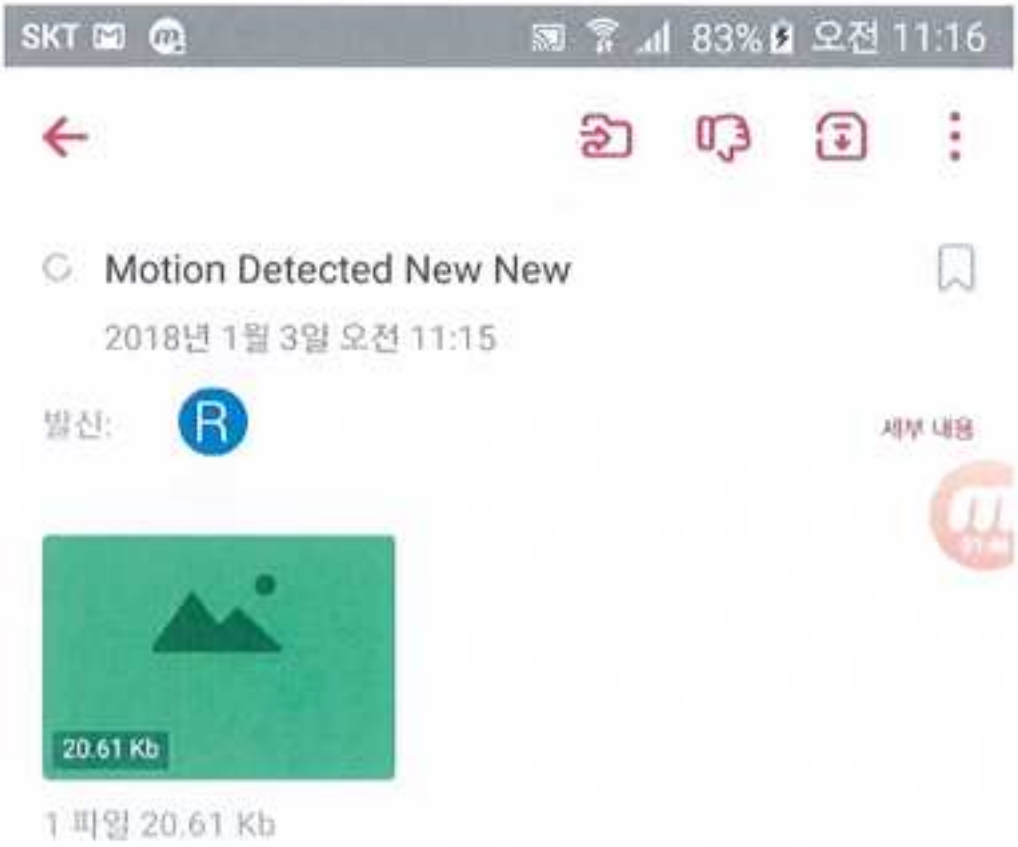
(3) 스마트폰에서 어플리케이션에서 확인한 모습

침입 탐지가 되어서 메일이 온 모습
상단바에 메일이 온 것을 확인할 수 있음
(push 서비스 응답이 이상해서 메일 관련 어플리케이션 두 개로 테스트함)

(3) 스마트폰에서 어플리케이션에서 확인한 모습

창을 내려서 push알림이 제대로 도착한 모습을 알 수 있음
메일의 도착주소는 @gmail.com이다.

(3) 스마트폰에서 어플리케이션에서 확인한 모습



push알림을 터치해서 메일을 확인하는 모습.
첨부된 사진파일이 존재함을 알 수 있다.

(3) 스마트폰에서 어플리케이션에서 확인한 모습

첨부된 사진을 확인한 모습.
현재 모션촬영 옵션은 침입탐지가 적절하게 이루어진 한 장만을 저장하고 메일로 보내는 형태로, 옵션을 통해 frame의 수만큼 침입탐지가 이루어진 장면을 가지고 통째로 저장하는 방법도 있음
(부하가 많이 걸리고, 제대로 탐지되지 못한 사진또한 저장되는 단점)

(4) 보안 기능

(1) 안드로이드 httpAuthRequest 사용

```
//zoom 허용
mWebView.getSettings().setBuiltInZoomControls(true);
mWebView.getSettings().setSupportZoom(true);

//javascript의 window.open 허용
mWebView.getSettings().setJavaScriptCanOpenWindowsAutomatically(true);
//javascript 허용
mWebView.getSettings().setJavaScriptEnabled(true);
mWebView.getSettings().setSupportMultipleWindows(true);

//meta태그의 viewport사용 가능
mWebView.getSettings().setUseWideViewPort(true);

class MyWebViewClient extends WebViewClient {
    @Override
    public void onReceivedHttpAuthRequest(WebView view,
                                          HttpAuthHandler handler, String host, String realm) {

        handler.proceed("username", "password");

    }
}

mWebView.setWebViewClient(new MyWebViewClient ());


mWebView.setWebChromeClient(new WebChromeClient());
mWebView.loadUrl("http://192.168.0.14:8080");
```

Motion에서 보안연결을 지원하기 때문에 안드로이드 어플리케이션에서도 이에 해당하는 보안연결을 지원해야함.

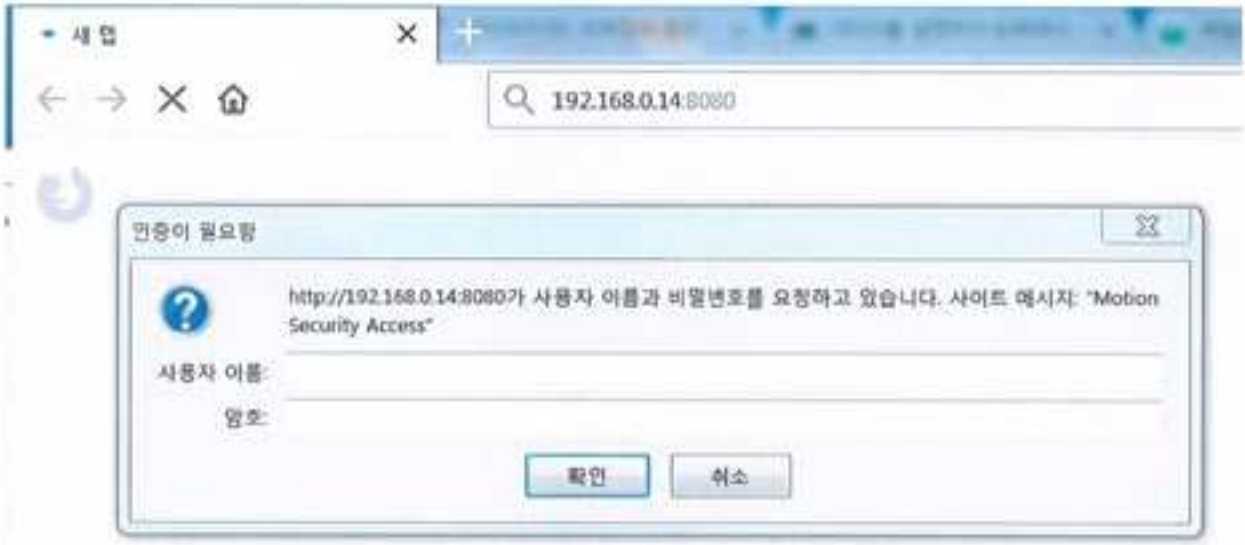
따라서 AuthRequest가 발생했을 때 처리하는 함수를 만들고 계정과 비밀번호를 사용할 수 있도록 어플리케이션을 수정함

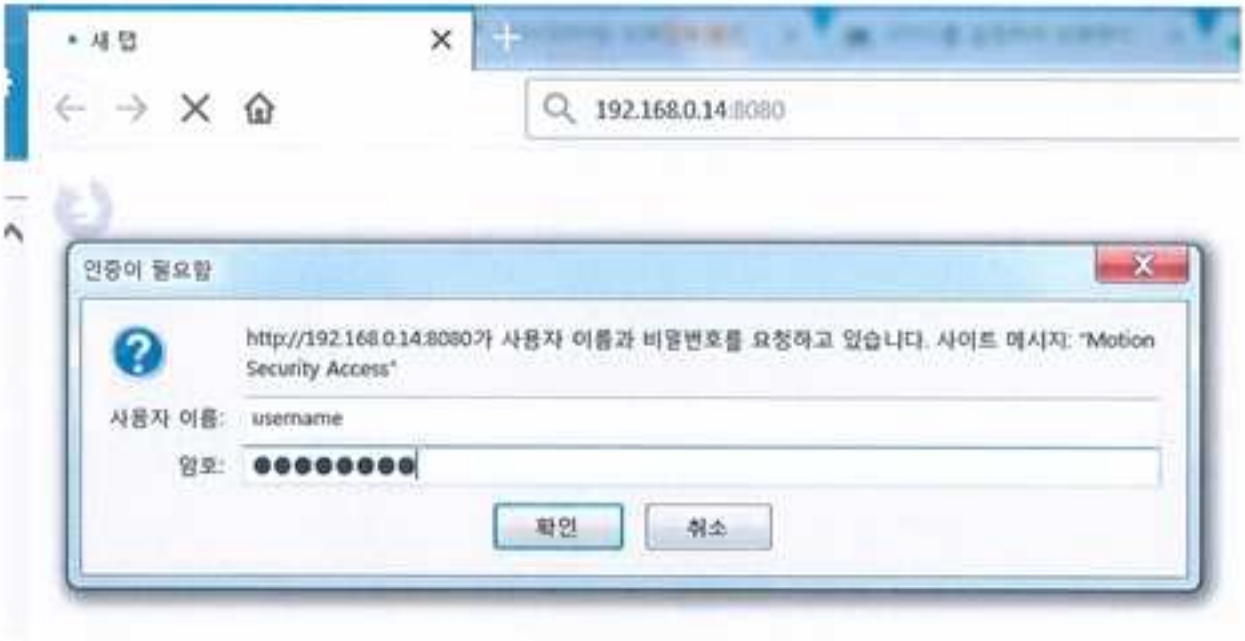
(2) Motion에서의 보안 연결

- stream_auth_method = 0
- stream_authentication = username:password
- stream_preview_scale = 100
- stream_preview_newline = off
- webcontrol_port = 8080
- webcontrol_localhost = off
- webcontrol_html_output = on
- webcontrol_authentication = username:password
- track_type = 0

stream auth method의 지정된 숫자를 변경시켜 보안 레벨을 올릴 수 있음
0은 보안기능이 없음X(disabled)
1은 한번 연결되면 더 이상 패스워드와 계정을 묻지 않음(최초인증)
2는 접속할 때 마다 패스워드와 계정을 물음

(3) 웹서버에서 보안 인증 요구





미리 지정해두었던 암호를 입력

암호를 입력하고 들어간 보안 연결이 된 모습

4. 자기평가

-문제점
1) 아두이노 만으로 opencv를 통해 영상처리를 하는 것은 불가능
2) 개발 도중 필요한 지식이 생각보다 많았음
- 아두이노 크탕
- 영상처리 관련 지식
- 안드로이드
- C++문법
- Wifi 및 시리얼 통신
- 전기 제어
- 웹
- 서버

3) 카메라 성능이 생각보다 부족함

-추후 보완사항
1) 라즈베리 파이를 통해 영상처리를 하고, 그 결과를 아두이노로 전송받을 수 있음

2) 또는 컴퓨터와 아두이노를 케이블로 연결해서 컴퓨터에서 처리를 하고 아두이노로 전송하는 형태를 취함

3) 이 경우에는 아두이노가 필요없어져서 실제 사용에는 상당한 제약이 따름

라즈베리 파이 + 웹캠 세팅 이후 최종 자기평가

1) 라즈베리파이와 원격 시스템(스마트폰 또는 컴퓨터)은 동일 네트워크에서 동작하기 때문에 외부접속을 위해서는 포트포워딩과 DDNS 설정이 필요함

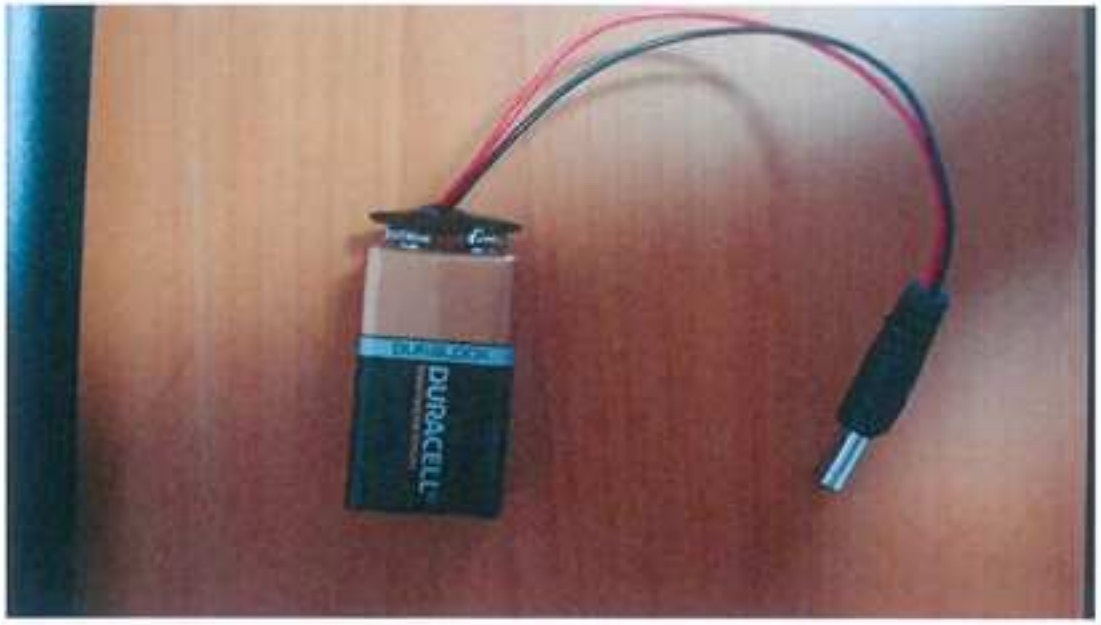
2) 라즈베리파이에 존재하는 웹캠은 별도전원이 공급되는 허브가 반드시 필요함
(usb 전력공급 부족으로 오작동하는 경우가 자주 생김)

3) 아두이노 단독으로 사용할 때보다 훨씬 안정적이며, 인터넷의 연결도 훨씬 원활했음(유선연결)

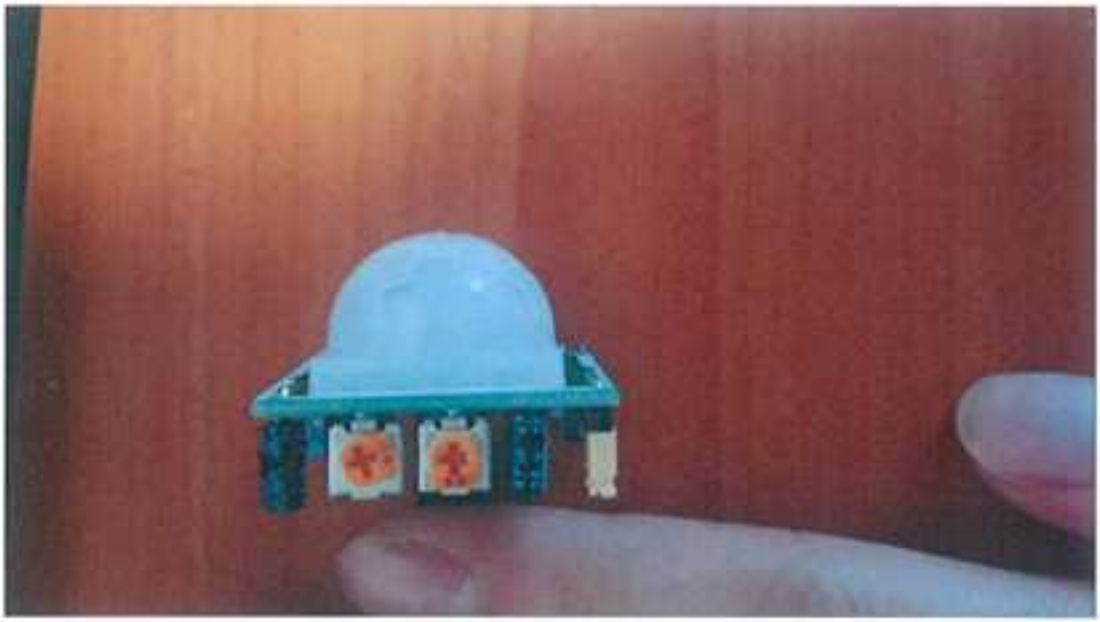
5. 최종 결과물

-아두이노 결과물 사진

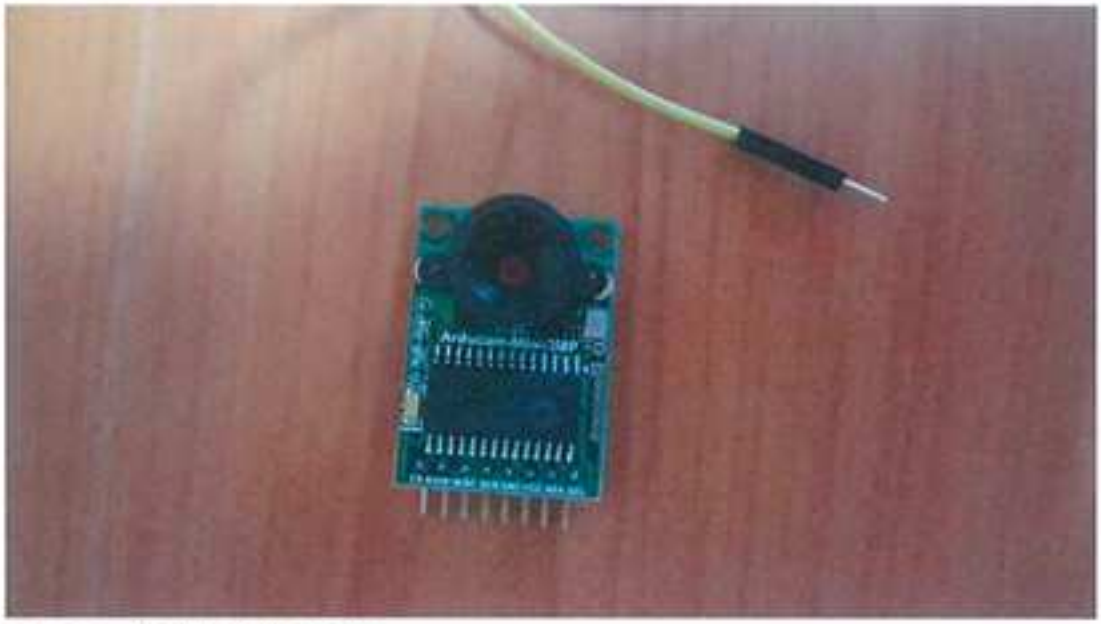
Arducam esp8266 uno V2



전원공급을 위한 9V배터리와 케이블

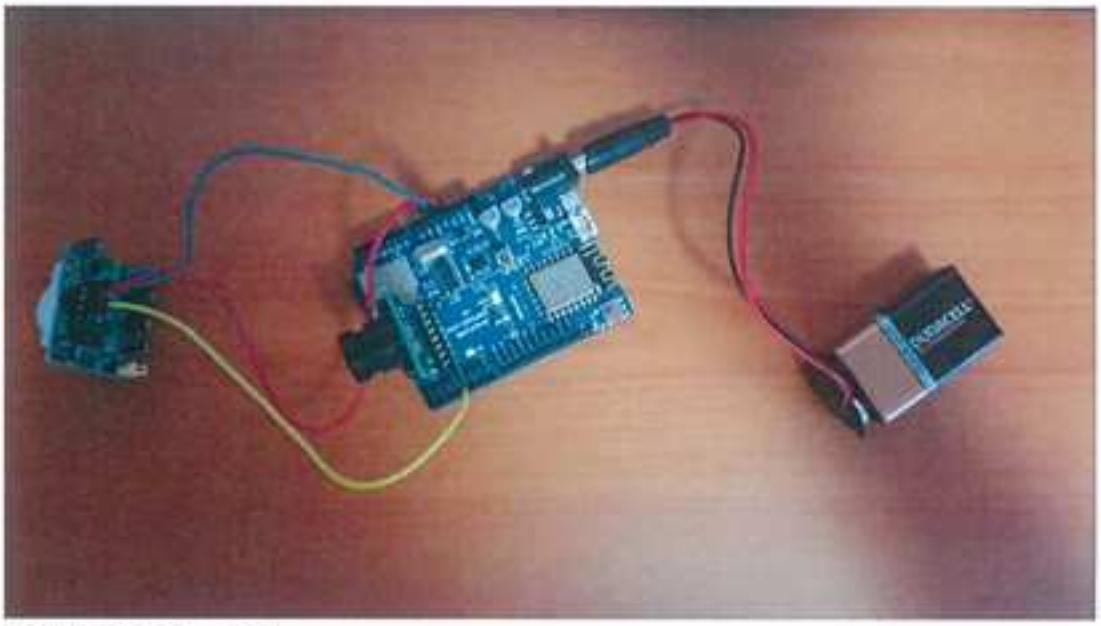


PIR 모션감지 센서



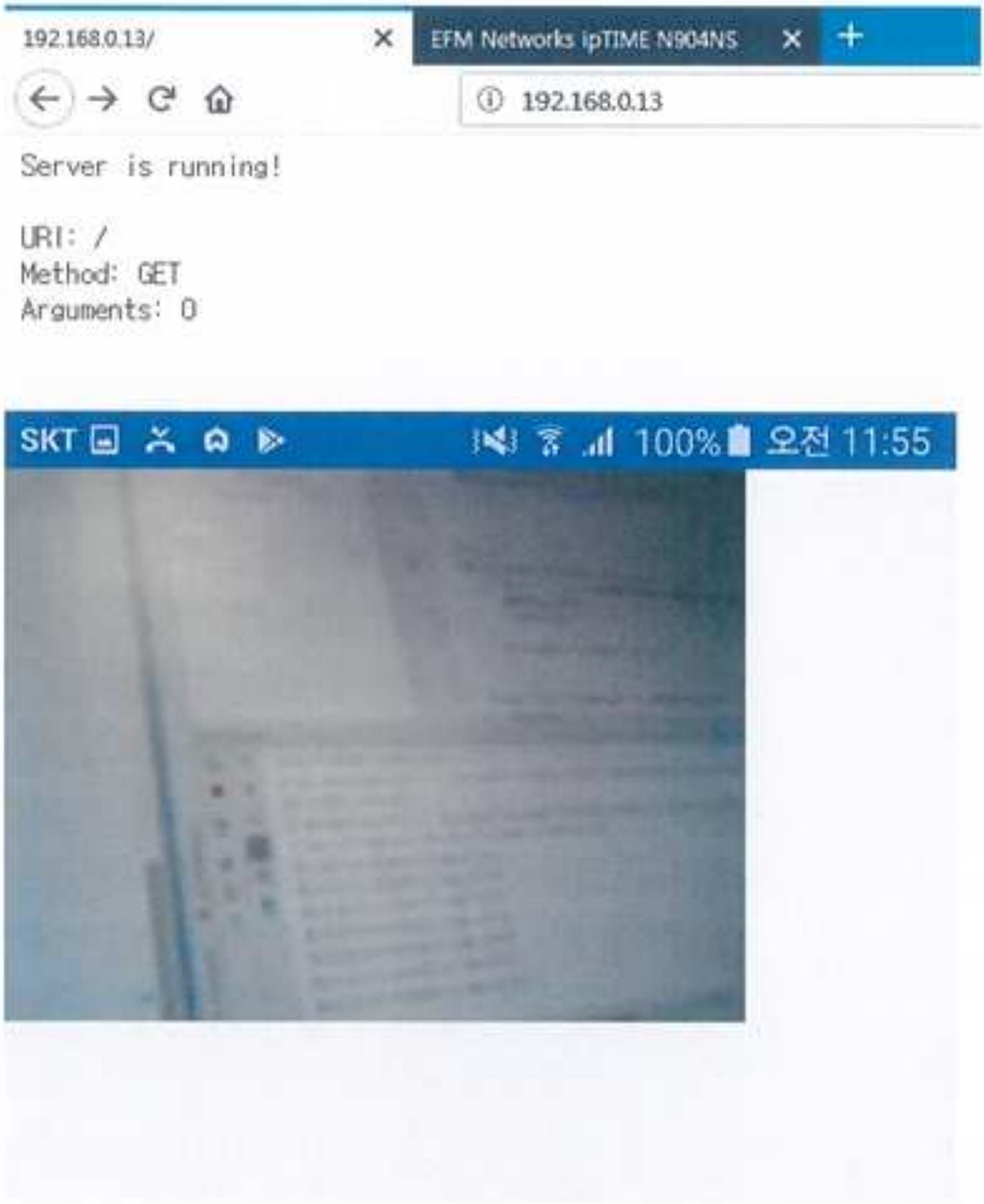
아두이노 카메라 OV2640 2M

데이터 저장을 위한 32GB Micro SD 카드



연결된 아두이노 모형

-시연 사진


192.168.0.13/
EFM Networks ipTIME N904NS
192.168.0.13
Server is running!
URI: /
Method: GET
Arguments: 0
SKT
100%
오전 11:55

-소스코드

1)안드로이드 어플리케이션 소스코드

File - C:\Users\cho\AndroidStudioProjects\MyApplication\app\src\main\AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.example.cho.myapplication" >
4      <uses-permission android:name="android.permission.INTERNET"/>
5
6      <application
7          android:allowBackup="true"
8          android:icon="@mipmap/ic_launcher"
9          android:label="@string/app_name"
10         android:roundIcon="@mipmap/ic_launcher_round"
11         android:supportsRtl="true"
12         android:theme="@style/AppTheme" >
13         <activity
14             android:name=".MainActivity"
15             android:label="@string/app_name"
16             android:theme="@style/AppTheme.NoActionBar" >
17             <intent-filter>
18                 <action android:name="android.intent.action.MAIN" />
19
20                 <category android:name="android.intent.category.LAUNCHER"
    />
21             </intent-filter>
22         </activity>
23     </application>
24
25 </manifest>
```

Page 1 of 1

File - C:\Users\cho\AndroidStudioProjects\MyApplication\app\src\main\res\layout\content_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:showIn="@layout/activity_main"
    tools:context="com.example.cho.myapplication.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>

```

Page 1 of 1

File - C:\Users\cho\AndroidStudioProjects\MyApplication\app\src\main\res\layout\activity_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/
   android"
3      xmlns:tools="http://schemas.android.com/tools"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:paddingBottom="@dimen/activity_vertical_margin"
7      android:paddingLeft="@dimen/activity_horizontal_margin"
8      android:paddingRight="@dimen/activity_horizontal_margin"
9      android:paddingTop="@dimen/activity_vertical_margin"
10     tools:context="com.example.cho.myapplication.MainActivity">
11
12     <WebView
13         android:layout_width="fill_parent"
14         android:layout_height="fill_parent"
15         android:id="@+id/webView"
16         android:layout_alignParentBottom="true"
17         android:layout_alignParentLeft="true"
18         android:layout_alignParentStart="true"
19         android:layout_alignParentTop="true"
20         android:layout_alignParentRight="true"
21         android:layout_alignParentEnd="true" />
22 </RelativeLayout>
```

Page 1 of 1

File - C:\Users\cho\AndroidStudioProjects\MyApplication\app\src\main\res\values\colors.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#3F51B5</color>
    <color name="colorPrimaryDark">#303F9F</color>
    <color name="colorAccent">#FF4081</color>
</resources>

```

File - C:\Users\cho\AndroidStudioProjects\MyApplication\app\src\main\res\values\dimens.xml

```
1  <resources>
2      <dimen name="fab_margin">16dp</dimen>
3      <dimen name="activity_horizontal_margin">0dp</dimen>
4      <dimen name="activity_vertical_margin">0dp</dimen>
5  </resources>
6
```

Page 1 of 1

File - C:\Users\cho\AndroidStudioProjects\MyApplication\app\src\main\res\values\styles.xml

```
 1  <resources>
 2
 3      <!-- Base application theme. -->
 4      <style name="AppTheme" parent="Theme.AppCompat.Light.
    DarkActionBar">
 5          <!-- Customize your theme here. -->
 6          <item name="colorPrimary">@color/colorPrimary</item>
 7          <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
 8          <item name="colorAccent">@color/colorAccent</item>
 9      </style>
10      <style name="AppTheme.NoActionBar">
11          <item name="windowActionBar">false</item>
12          <item name="windowNoTitle">true</item>
13      </style>
14      <style name="AppTheme.AppBarOverlay" parent="ThemeOverlay.
    AppCompat.Dark.ActionBar" />
15      <style name="AppTheme.PopupOverlay" parent="ThemeOverlay.
    AppCompat.Light" />
16
17  </resources>
18
```

Page 1 of 1

2)아두이노 소스코드

```
// ArduCAM Mini demo (C)2017 Lee
// Web: http://www.ArduCAM.com
// This program is a demo of how to use most of the functions
// of the library with ArduCAM ESP8266 2MP/5MP camera.
// This demo was made for ArduCAM ESP8266 2MP/5MP Camera.
// It can take photo and send to the Web.
// It can take photo continuously as video streaming and send to the Web.
// The demo sketch will do the following tasks:
// 1. Set the camera to JPEG output mode.
// 2. if server.on("/capture", HTTP_GET, serverCapture),it can take photo and send to the Web.
// 3.if server.on("/stream", HTTP_GET, serverStream),it can take photo continuously as video
//streaming and send to the Web.

// This program requires the ArduCAM V4.0.0 (or later) library and ArduCAM ESP8266 2MP/5MP camera
// and use Arduino IDE 1.6.8 compiler or above

#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>
#include <Wire.h>
#include <ArduCAM.h>
#include <SPI.h>
#include "memorysaver.h"
#if !(defined ESP8266 )
#error Please select the ArduCAM ESP8266 UNO board in the Tools/Board
#endif

//This demo can only work on OV2640_MINI_2MP or ARDUCAM_SHIELD_V2 platform.
#if !(defined (OV2640_MINI_2MP)||defined (OV5640_MINI_5MP_PLUS) || defined (OV5642_MINI_5MP_PLUS) \
    || defined (OV5642_MINI_5MP) || defined (OV5642_MINI_5MP_BIT_ROTATION_FIXED) \
    ||(defined (ARDUCAM_SHIELD_V2) && (defined (OV2640_CAM) || defined (OV5640_CAM) || defined
(OV5642_CAM))))
#error Please select the hardware platform and camera module in the
../libraries/ArduCAM/memorysaver.h file
#endif
// set GPIO16 as the slave select :
const int CS = 16;


//you can change the value of wifiType to select Station or AP mode.
//Default is AP mode.
int wifiType = 0; // 0:Station  1:AP
```

```
//AP mode configuration
//Default is arducam_esp8266.If you want,you can change the AP_aSSID  to your favorite name
const char *AP_ssid = "arducam_esp8266";
//Default is no password.If you want to set password,put your password here
const char *AP_password = "";

//Station mode you should put your ssid and password
const char *ssid = "Home_2.4Ghz"; // Put your SSID here
const char *password = "gosehl1!"; // Put your PASSWORD here

static const size_t bufferSize = 4096;
static uint8_t buffer[bufferSize] = {0xFF};
uint8_t temp = 0, temp_last = 0;
int i = 0;
bool is_header = false;

ESP8266WebServer server(80);
#if defined (OV2640_MINI_2MP) || defined (OV2640_CAM)
  ArduCAM myCAM(OV2640, CS);
#elif defined (OV5640_MINI_5MP_PLUS) || defined (OV5640_CAM)
  ArduCAM myCAM(OV5640, CS);
#elif defined (OV5642_MINI_5MP_PLUS) || defined (OV5642_MINI_5MP) || defined
(OV5642_MINI_5MP_BIT_ROTATION_FIXED) ||(defined (OV5642_CAM))
  ArduCAM myCAM(OV5642, CS);
#endif


void start_capture(){
  myCAM.clear_fifo_flag();
  myCAM.start_capture();
}

void camCapture(ArduCAM myCAM){
WiFiClient client = server.client();
uint32_t len  = myCAM.read_fifo_length();
if (len >= MAX_FIFO_SIZE) //8M
{
  Serial.println(F("Over size."));
}
if (len == 0 ) //0 kb
{
  Serial.println(F("Size is 0."));
}
myCAM.CS_LOW();
```

```
myCAM.set_fifo_burst();
if (!client.connected()) return;
String response = "HTTP/1.1 200 OK\r\n";
response += "Content-Type: image/jpeg\r\n";
response += "Content-Len: " + String(len) + "\r\n\r\n";
server.sendContent(response);
i = 0;
while (len--)
{
temp_last = temp;
temp = SPI.transfer(0x00);
//Read JPEG data from FIFO
if ( (temp == 0xD9) && (temp_last == 0xFF) ) //If find the end ,break while,
{
buffer[i++] = temp;  //save the last  0XD9
//Write the remain bytes in the buffer
if (!client.connected()) break;
client.write(&buffer[0], i);
is_header = false;
i = 0;
myCAM.CS_HIGH();
break;
}
if (is_header == true)
{
//Write image data to buffer if not full
if (i < bufferSize)
buffer[i++] = temp;
else
{
//Write bufferSize bytes image data to file
if (!client.connected()) break;
client.write(&buffer[0], bufferSize);
i = 0;
buffer[i++] = temp;
}
}
else if ((temp == 0xD8) & (temp_last == 0xFF))
{
is_header = true;
buffer[i++] = temp_last;
buffer[i++] = temp;
}
}
```

```
}

void serialCapture(){
start_capture();
Serial.println("OK Capturing");

int total_time = 0;

total_time = millis();
while (!myCAM.get_bit(ARDUCHIP_TRIG, CAP_DONE_MASK));
total_time = millis() - total_time;
Serial.println("Capture total_time used (in ms):" + total_time);
Serial.println(total_time, DEC);

total_time = 0;

Serial.println("OK Capture Done.");
total_time = millis();
readCapturedImg();
total_time = millis() - total_time;
Serial.println("send total_time used (in ms):" + total_time);
Serial.println(total_time, DEC);
Serial.println("OK send Done.");
}

void serialRead(){
uint32_t len1, len2, len3, length = 0;

String response = "HTTP/1.1 200 OK\r\n";
response += "Content-Type: multipart/x-mixed-replace; boundary=frame\r\n\r\n";
server.sendContent(response);

while (1){
start_capture();
while (!myCAM.get_bit(ARDUCHIP_TRIG, CAP_DONE_MASK));
size_t len = myCAM.read_fifo_length();
if (len >= MAX_FIFO_SIZE) //8M
{
Serial.println("Over size.");
continue;
}
if (len == 0 ) //0 kb
{
Serial.println("Size is 0.");
```

```
continue;
}
myCAN.CS_LOW();
myCAN.set_fifo_burst();
if (!client.connected()) break;
response = "--frame\r\n";
response += "Content-Type: image/jpeg\r\n\r\n";
server.sendContent(response);
while (1==1)
{
temp_last = temp;
temp = SPI.transfer(0x00);

//Read JPEG data from FIFO
if ( (temp == 0xD9) && (temp_last == 0xFF) ) //If find the end ,break while,
{
buffer[++i] = temp;  //save the last  0xD9
//Write the remain bytes in the buffer
myCAN.CS_HIGH();
if (!client.connected()) break;
client.write(&buffer[0], i);
is_header = false;
i = 0;
}
if (is_header == true)
{
//Write image data to buffer if not full
if (i < bufferSize)
buffer[++i] = temp;
else
{
//Write bufferSize bytes image data to file
myCAN.CS_HIGH();
if (!client.connected()) break;
client.write(&buffer[0], bufferSize);
i = 0;
buffer[++i] = temp;
myCAN.CS_LOW();
myCAN.set_fifo_burst();
}
}
else if ((temp == 0xD8) && (temp_last == 0xFF))
{
is_header = true;
```

```
buffer[i++] = temp_last;
buffer[i++] = temp;
}
}
if (!client.connected()) break;
}
}

void handleNotFound(){
String message = "Server is running!\n\n";
message += "URI: ";
message += server.uri();
message += "\nMethod: ";
message += (server.method() == HTTP_GET)?"GET":"POST";
message += "\nArguments: ";
message += server.args();
message += "\n";
server.send(200, "text/plain", message);

if (server.hasArg("ql")){
int ql = server.arg("ql").toInt();
#if defined (OV2640_MINI_2MP) || defined (OV2640_CAM)
myCAM.OV2640_set_JPEG_size(ql);
#elif defined (OV5640_MINI_5MP_PLUS) || defined (OV5640_CAM)
myCAM.OV5640_set_JPEG_size(ql);
#elif defined (OV5642_MINI_5MP_PLUS) || defined (OV5642_MINI_5MP_BIT_ROTATION_FIXED) ||(defined
(OV5642_CAM))
myCAM.OV5642_set_JPEG_size(ql);
#endif
delay(1000);
Serial.println("QL change to: " + server.arg("ql"));
}
}

void setup() {
uint8_t vid, pid;
uint8_t temp;
#if defined(__SAM3X8E__)
Wire1.begin();
#else
Wire.begin();
#endif
Serial.begin(115200);
Serial.println(F("ArduCAM Start!"));
```

```
// set the CS as an output:
pinMode(SS, OUTPUT);

// initialize SPI:
SPI.begin();
SPI.setFrequency(80000000);  //40m

//Check if the ArduCAM SPI bus is OK
myCAM.write_reg(ARDUCHIP_TEST1, 0x55)
temp = myCAM.read_reg(ARDUCHIP_TEST1);
if (temp != 0x55){
Serial.println(F("SPI1 interface Error!"));
while(1);
}
#if defined (OV2640_MINI_2MP) || defined (OV9640_CAM)
//Check if the camera module type is OV2640
myCAM.wrSensorReg8_8(0xff, 0x01);
myCAM.rdSensorReg8_8(OV2640_CHIPID_HIGH, &vid);
myCAM.rdSensorReg8_8(OV2640_CHIPID_LOW, &pid);
if ((vid != 0x26 ) && (( pid != 0x41 ) || ( pid != 0x42 )))
Serial.println(F("Can't find OV2640 module!"));
else
Serial.println(F("OV2640 detected."));
#elif defined (OV5640_MINI_5MP_PLUS) || defined (OV5640_CAM)
//Check if the camera module type is OV5640
myCAM.wrSensorReg16_8(0xff, 0x01);
myCAM.rdSensorReg16_8(OV5640_CHIPID_HIGH, &vid);
myCAM.rdSensorReg16_8(OV5640_CHIPID_LOW, &pid);
if((vid != 0x56) || (pid != 0x40))
Serial.println(F("Can't find OV5640 module!"));
else
Serial.println(F("OV5640 detected."));
#elif defined (OV5642_MINI_5MP_PLUS) || defined (OV5642_CAM) || defined
(OV5642_MINI_5MP_BIT_ROTATION_FIXED) || defined (OV5642_320X240)
//Check if the camera module type is OV5642
myCAM.wrSensorReg16_8(0xff, 0x01);
myCAM.rdSensorReg16_8(OV5642_CHIPID_HIGH, &vid);
myCAM.rdSensorReg16_8(OV5642_CHIPID_LOW, &pid);
if((vid != 0x56) || (pid != 0x42)){
Serial.println(F("Can't find OV5642 module!"));
}
else
Serial.println(F("OV5642 detected."));
```

```
#endif

//Change to JPEG capture mode and initialize the OV2640 module
myCAM.set_format(JPEG);
myCAM.InitCAM();
#if defined (OV2640_MINI_2MP) || defined (OV2640_CAM)
myCAM.OV2640_set_JPEG_size(OV2640_320x240);
#elif defined (OV5640_MINI_5MP_PLUS) || defined (OV5640_CAM)
myCAM.write_reg(ARDUCHIP_TIM, VSYNC_LEVEL_MASK);   //VSYNC is active HIGH
myCAM.OV5640_set_JPEG_size(OV5640_320x240);
#elif defined (OV5642_MINI_5MP_PLUS) || defined (OV5642_MINI_5MP) || defined
(OV5642_MINI_5MP_BIT_ROTATION_FIXED) ||(defined (OV5642_CAM))
myCAM.write_reg(ARDUCHIP_TIM, VSYNC_LEVEL_MASK);   //VSYNC is active HIGH
myCAM.OV5642_set_JPEG_size(OV5642_320x240);
#endif

myCAM.clear_fifo_flag();
if (wifiType == 0){
if(!strcmp(ssid,"SSID")){
Serial.println(F("Please set your SSID"));
while(1);
}
if(!strcmp(password,"PASSWORD")){
Serial.println(F("Please set your PASSWORD"));
while(1);
}
// Connect to WiFi network
Serial.println();
Serial.println();
Serial.print(F("Connecting to "));
Serial.println(ssid);

WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(F("."));
}
Serial.println(F("WiFi connected"));
Serial.println("");
Serial.println(WiFi.localIP());
}else if (wifiType == 1){
Serial.println();
```

```
Serial.println();
Serial.print("Captive AP: ");
Serial.println(AP_ssid);
Serial.print("The password is: ");
Serial.println(AP_password);

WiFi.mode(WIFI_AP);
WiFi.softAP(AP_ssid, AP_password);
Serial.println("");
Serial.print("Server IP address: ");
}

// Start the server
server.on("/capture", HTTP_GET, serveCapture);
server.on("/stream", HTTP_GET, serveStream);
server.onNotFound(handleNotFound);
server.begin();
Serial.println("Server started");
}

void loop() {
server.handleClient();
}
```

```
// ArduCAM demo (C)2017 Lee
// Web: http://www.ArduCAM.com
// This program is a demo of how to use most of the functions
// of the library with a supported camera modules, and can run on any Arduino platform.
// This demo was made for Omnivision 2MP/5MP sensors
// It will run the ArduCAM (2/5MP) OV2640/OV5640 as a real 2MP/5MP digital camera, provide both JPEG
capture.
// The demo sketch will do the following tasks:
// 1. Set the sensor to JPEG mode
// 2. Capture and buffer the image to FIFO every 5 seconds
// 3. Store the image to Micro SD/TF card with JPEG format in sequential
// 4. Resolution can be changed by myCAM.set_JPEG_size() function
// This program requires the ArduCAM V4.0.0 (or later) library and ArduCAM ESP8266 2MP/5MP shield
// and use Arduino IDE 1.6.8 compiler or above
#include <ArduCAM.h>
#include <Wire.h>
#include <SPI.h>
#include <SD.h>
#include "memorysaver.h"
#if !(defined ESP8266 )
#error Please select the ArduCAM ESP8266 UNO board in the Tools/Board
#endif

//This demo can only work on
OV2640_MINI_2MP/OV5640_MINI_5MP_PLUS/OV5642_MINI_5MP_PLUS/OV5642_MINI_5MP_PLUS
//OV3640_MINI_3MP_BIT_ROTATION_FIXED/ ARDUCAM_SHIELD_V2 platform.
#if !(defined (OV2640_MINI_2MP)||defined (OV5640_MINI_5MP_PLUS) || defined (OV5642_MINI_5MP_PLUS) ||
    || defined (OV5642_MINI_5MP) || defined (OV5647_MINI_5MP_BIT_ROTATION_FIXED) ||
    (defined (ARDUCAM_SHIELD_V2) && (defined (OV2640_CAM) || defined (OV5640_CAM) || defined
(OV5642_CAM))))
#error Please select the hardware platform and camera module in the
../libraries/ArduCAM/memorysaver.h !!!
#endif
// set GPIO16 as the slave select
const int CS = 16;
//Version 2.xxx: CS=10 as the slave select :
const int SD_CS = 5;
const int CAM_POWER_ON = 16;
const int size[][2] = {

#if defined (OV2640_MINI_2MP) || defined (OV2640_CAM)
ArduCAM myCAM(OV2640, CS);
#elif defined (OV5640_MINI_5MP_PLUS) || defined (OV5640_CAM)
ArduCAM myCAM(OV5640, CS);
```

```
#elif defined (OV5642_MINI_5MP_PLUS) || defined (OV5642_MINI_5MP) || defined
(OV5642_MINI_5MP_BIT_ROTATION_FIXED) ||(defined (OV5642_CAM))
ArduCAM myCAM(OV5642, CS);
#endif
void myCAMSaveToSDFile(){
  char str[8];
  byte buf[256];
  static int i = 0;
  static int k = 0;
  uint8_t temp = 0, temp_last = 0;
  uint32_t length = 0;
  bool is_header = false;
  File outFile;
  //Flush the FIFO
  myCAM.flush_fifo();
  //Clear the capture done flag
  myCAM.clear_fifo_flag();
  //Start capture
  myCAM.start_capture();
  Serial.println(F("Star Capture"));
 while(!myCAM.get_bit(ARDUCHIP_TRIG , CAP_DONE_MASK));
 Serial.println(F("Capture Done."));

 length = myCAM.read_fifo_length();
 Serial.print(F("The fifo length is :"));
 Serial.println(length, DEC);
  if (length >= MAX_FIFO_SIZE) //8M
  {
    Serial.println(F("Over size."));
  }
    if (length == 0 ) //0 kb
  {
    Serial.println(F("Size is 0."));
  }
 //Construct a file name
 k = k + 1;
 itoa(k, str, 10);
 strcat(str, ".jpg");
 //Open the new file
 outFile = SD.open(str, O_WRITE | O_CREAT | O_TRUNC);
 if(! outFile){
  Serial.println(F("File open faild"));
  return;
 }
```

```
i = 0;
myCAM.CS_LOW();
myCAM.set_fifo_burst();

while ( length-- )
  {
    temp_last = temp;
    temp = SPI.transfer(0x00);
    //Read JPEG data from FIFO
    if ( (temp == 0xD9) && (temp_last == 0xFF) ) //If find the end ,break while,
    {
        buf[i++] = temp;  //save the last  0XD9
       //Write the remain bytes in the buffer
        myCAM.CS_HIGH();
        outFile.write(buf, i);
       //Close the file
        outFile.close();
        Serial.println("OK");
        is_header = false;
        i = 0;
    }
    if (is_header == true)
    {
       //Write image data to buffer if not full
        if (i < 256)
        buf[i++] = temp;
        else
        {
          //Write 256 bytes image data to file
          myCAM.CS_HIGH();
          outFile.write(buf, 256);
          i = 0;
          buf[i++] = temp;
          myCAM.CS_LOW();
          myCAM.set_fifo_burst();
        }
    }
    else if ((temp == 0xD8) & (temp_last == 0xFF))
    {
      is_header = true;
      buf[i++] = temp_last;
      buf[i++] = temp;
    }
  }
```

```
}

void setup() {
  uint8_t vid, pid;
  uint8_t temp;
  Wire.begin();
  Serial.begin(115200);
  Serial.println(F("ArduCAM Start!"));

  //set the CS as an output:
  pinMode(CS,OUTPUT);
  pinMode(CAM_POWER_ON , OUTPUT);
  digitalWrite(CAM_POWER_ON, HIGH);

  //initialize SPI:
  SPI.begin();
  SPI.setFrequency(4000000); //4MHz

  delay(1000);
  //Check if the ArduCAM SPI bus is OK
  myCAM.write_reg(ARDUCHIP_TEST1, 0x55);
  temp = myCAM.read_reg(ARDUCHIP_TEST1);
  if (temp != 0x55){
    Serial.println(F("SPI1 interface Error!"));
    while(1);
  }

    //Initialize SD Card
  if(!SD.begin(SD_CS)){
    Serial.println(F("SD Card Error"));
  }
  else
  Serial.println(F("SD Card detected!"));

#if defined (OV2640_MINI_2MP) || defined (OV2640_CAM)
  //Check if the camera module type is OV2640
  myCAM.wrSensorReg8_8(0xff, 0x01);
  myCAM.rdSensorReg8_8(OV2640_CHIPID_HIGH, &vid);
  myCAM.rdSensorReg8_8(OV2640_CHIPID_LOW, &pid);
  if ((vid != 0x26 ) && (( pid != 0x41 ) || ( pid != 0x42 )))
   Serial.println(F("Can't find OV2640 module!"));
  else
   Serial.println(F("OV2640 detected."));
 #elif defined (OV5640_MINI_5MP_PLUS) || defined (OV5640_CAM)
```

3)openCV 소스코드

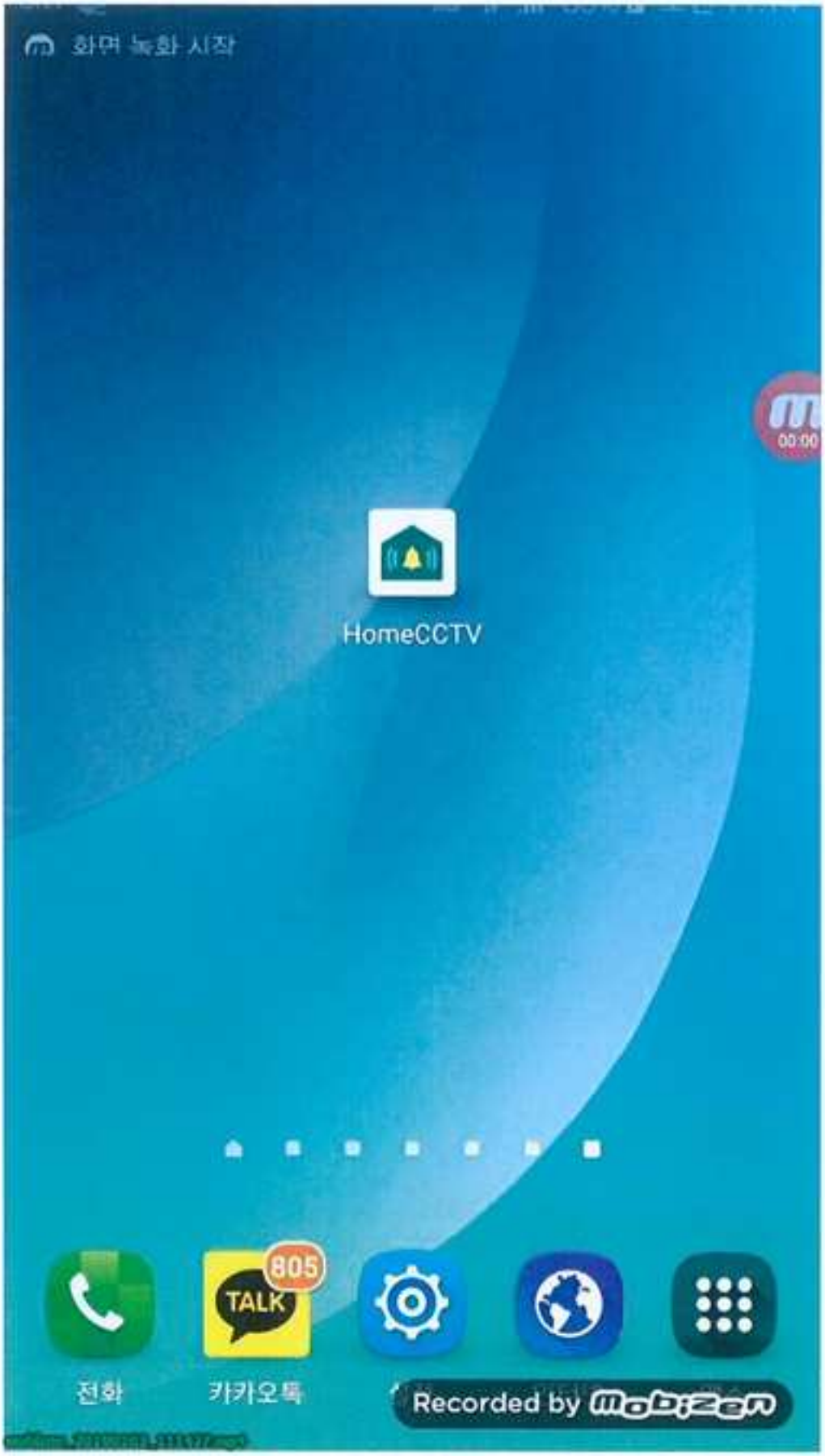
...rWOpenCV-IntruderDetection-masterWIntruderDetectionWmain.cpp 1

```
#include "opencv/cv.h"
#include "opencv/highgui.h"

#include <Windows.h>
#include <iostream>

#define WIN_NAME '침입자 탐지'
#define DIFF_THRESHOLD 0.1
#define FACE_CLASSIFIER_PATH "C:WWProgram FilesWWopencvWWsourcesWWdataW
  Whaarcascades₩₩haarcascade_frontalface_default.xml"
#define FACE_SEARCH_SCALE 1.1
#define MERGE_DETECTED_GROUP_CNTS 3
#define FACE_FRAME_WIDTH 50
#define FACE_FRAME_HEIGHT 50
#define FACE_FRAME_THICKNESS 1
#define INTRUDER_IMAGES_SAVE_PATH "..WWintruder_images₩₩"

/**
*   현재 시간정보를 문자열로 가져오는 메소드
*/
const std::string getCurrentTS2Str() {
    time_t now = time(0);
    struct tm tstruct;
    char buf[80];
    tstruct = *localtime(&now);
    strftime(buf, sizeof(buf), "%Y%m%d%H%M%S", &tstruct);

    return buf;
}

/**
*   메인 메소드
*/
int main(int argc, char *argv[]) {
    // 웹 캠 생성
    cv::VideoCapture capture(0);

    // 웹 캠을 실행하지 못한 경우 에러 출력 및 종료
    if (!capture.isOpened()) {
        std::cerr << "웹 캠 디바이스를 찾을 수 없습니다." << std::endl;
        return 0;
    }

    // 윈도우 생성
    cv::namedWindow(WIN_NAME, 1);

    // 얼굴인식 템플릿 설정
    cv::CascadeClassifier face_classifier;
    face_classifier.load(FACE_CLASSIFIER_PATH);

    std::cout << "'esc' 키를 이용하여 프로그램 종료 " << std::endl;

    // 이전 그레이 스케일 프레임을 저장하는 변수
    cv::Mat frameBeforeGrayScale;

    while (true) {
```

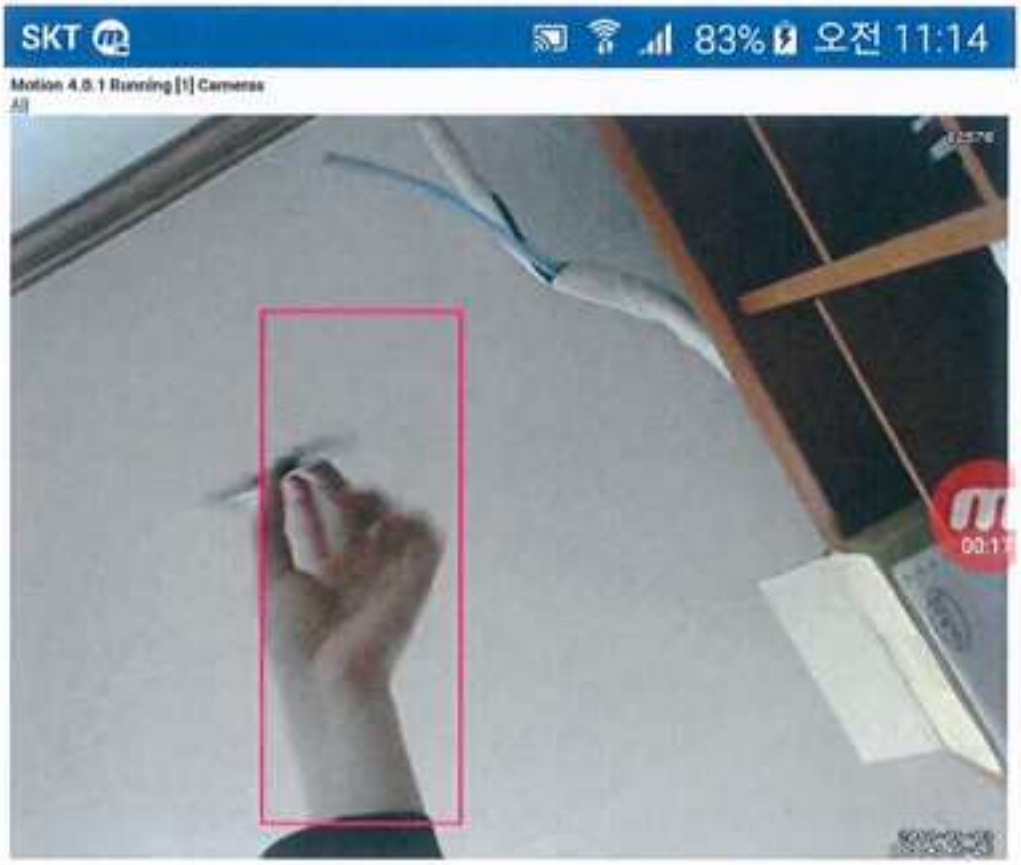
```
58          bool isFrameValid = true;
57          cv::Mat frameOriginalMat;
58          cv::Mat frame;
59
60          try {
61              // 웹 캠 프레임의 원본 크기 저장
62              capture >> frameOriginalMat;
63
64              // 원본 크기의 1/2로 축소 (프레임의 크기가 클 경우 연산시간이 증
                  가)
65              cv::resize(frameOriginalMat, frame, cv::Size
                  (frameOriginalMat.cols / 2, frameOriginalMat.rows / 2), 0, 0,
                  CV_INTER_NN);
66          } catch (cv::Exception& e) {
67              // 에러 출력
68              std::cerr << "프레임 축소에 실패했기에, 해당 프레임을 무시합니다." <<
                    e.err << std::endl;
69              isFrameValid = false;
70          }
71
72          // 현재 프레임을 그레이 스케일로 변경
73          cv::Mat frameCurrentGrayScale;
74          cv::cvtColor(frame, frameCurrentGrayScale, CV_BGR2GRAY);
75
76          if (frameCurrentGrayScale.size == frameBeforeGrayScale.size) {
77              // 정규화를 통해 이전 프레임과 현재 프레임의 차이의 정도를 구함
78              double errorL2 = cv::norm(frameCurrentGrayScale,
                    frameBeforeGrayScale, CV_L2);
79              double diff = errorL2 / (double)(frameCurrentGrayScale.rows *
                    frameBeforeGrayScale.rows);
80
81              // 임계치 값보다 프레임의 차이가 클 경우
82              if (diff >= DIFF_THRESHOLD) {
83                  // 얼굴영역을 저장 할 벡터 변수
84                  std::vector<cv::Rect> faces;
85
86                  // 얼굴인식 템플릿을 이용하여 얼굴인식
87                  face_classifier.detectMultiScale(
88                      frameCurrentGrayScale, faces,
89                      FACE_SEARCH_SCALE,
90                      MERGE_DETECTED_GROUP_CNTS,
91                      CV_HAAR_FIND_BIGGEST_OBJECT | CV_HAAR_SCALE_IMAGE,
92                      cv::Size(FACE_FRAME_WIDTH, FACE_FRAME_HEIGHT)
93                  );
94
95                  int facesSize = faces.size();
96
97                  // 침입자의 얼굴을 감지한 경우
98                  if (facesSize != 0) {
99                      for (int i = 0; i < facesSize; i++) {
100                         // 침입자의 얼굴 프레임이 전체 프레임 밖을 벗어나지 않
                          았을 경우에만
101                         if (
102                             0 <= faces[i].x
103                             && 0 <= faces[i].width
104                             && faces[i].x + faces[i].width <= frame.cols
```

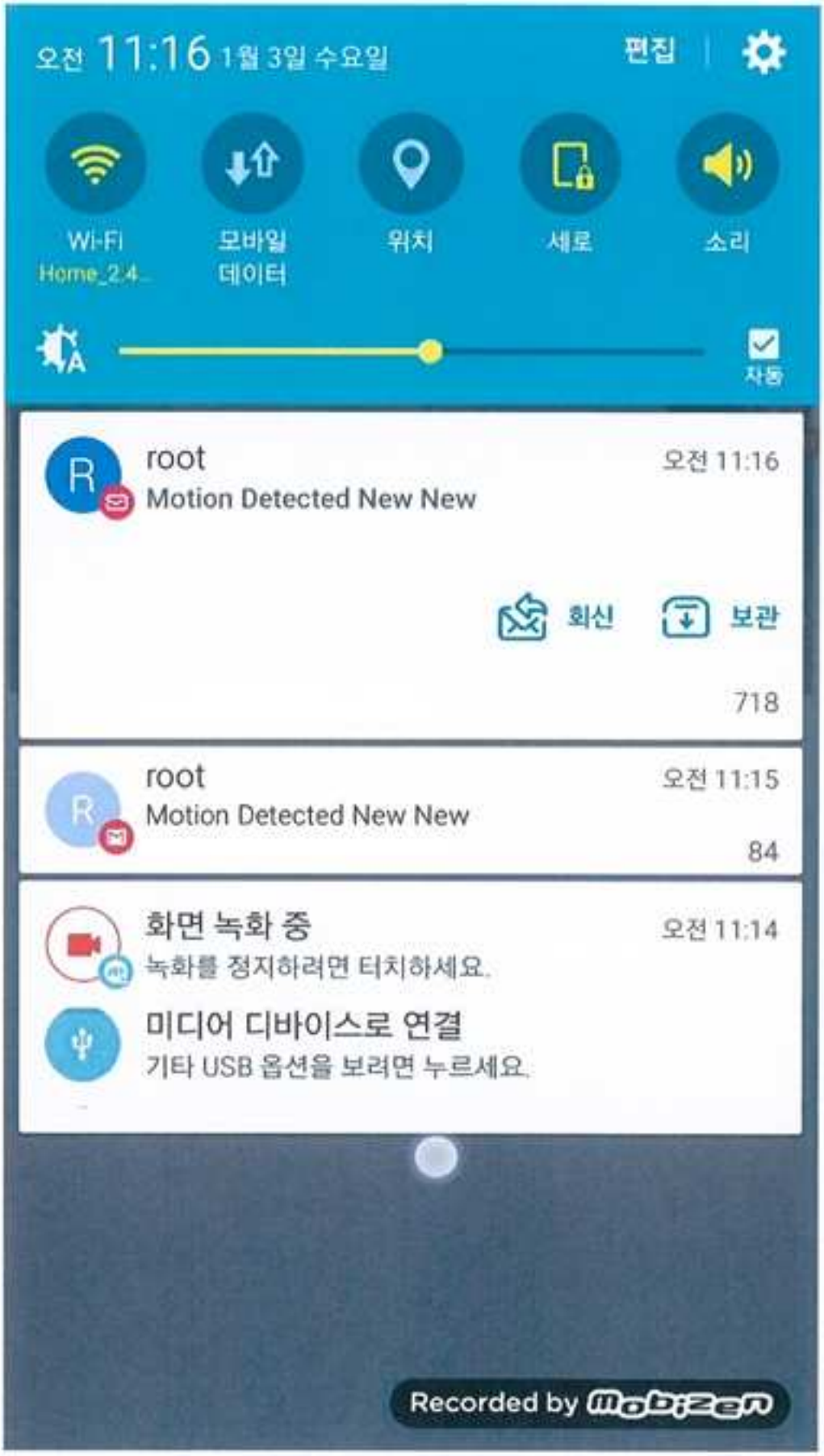
```
105                             && 0 <= faces[i].y
106                             && 0 <= faces[i].height
107                             && faces[i].y + faces[i].height <= frame.rows
108                         ) {
109                             // 침입자의 얼굴 부분만 프레임에서 잘라냄
110                             cv::Mat faceFrame = frame(cv::Rect(faces[i].x,
                        faces[i].y, faces[i].width, faces[i].height));
111
112                             // 침입자의 얼굴을 저장 할 디렉토리가 없을 경우 생
                        성
113                             CreateDirectory(INTRUDER_IMAGES_SAVE_PATH, NULL);
114
115                             // 침입자의 얼굴 프레임을 이미지로 저장
116                             cv::imwrite(INTRUDER_IMAGES_SAVE_PATH +
                        getCurrentTS2Str() + std::string("_") + std::to_string(i) +
                         std::string(".jpg"), faceFrame);
117
118                             // 침입자 탐지 메세지를 화면에 출력
119                             cv::putText(
120                                 frame,
121                                 "Intruder Detected!",
122                                 cv::Point(50, 230),
123                                 CV_FONT_HERSHEY_COMPLEX_SMALL,
124                                 1.0,
125                                 cv::Scalar(0, 0, 255)
126                             );
127                         }
128                     }
129                 }
130
131                 // 움직임은 있었지만 침입자 얼굴을 감지하지 못한 경우
132                 else {
133                     // 움직임 감지 메세지를 화면에 출력
134                     cv::putText(
135                         frame,
136                         "Moving Object Detected!",
137                         cv::Point(20, 230),
138                         CV_FONT_HERSHEY_COMPLEX_SMALL,
139                         1.0,
140                         cv::Scalar(255, 0, 0)
141                     );
142                 }
143             }
144         }
145
146         // 윈도우에 결과 출력
147         cv::imshow(WIN_NAME, frame);
148
149         // 현재 프레임을 이전 프레임 저장 변수에 옮김
150         frameBeforeGrayScale = frameCurrentGrayScale;
151
152         int keyCode = cv::waitKey(30);
153
154         // esc 키가 눌리면 프레임 캡쳐 종료
155         if (keyCode == 27) {
156             break;
```

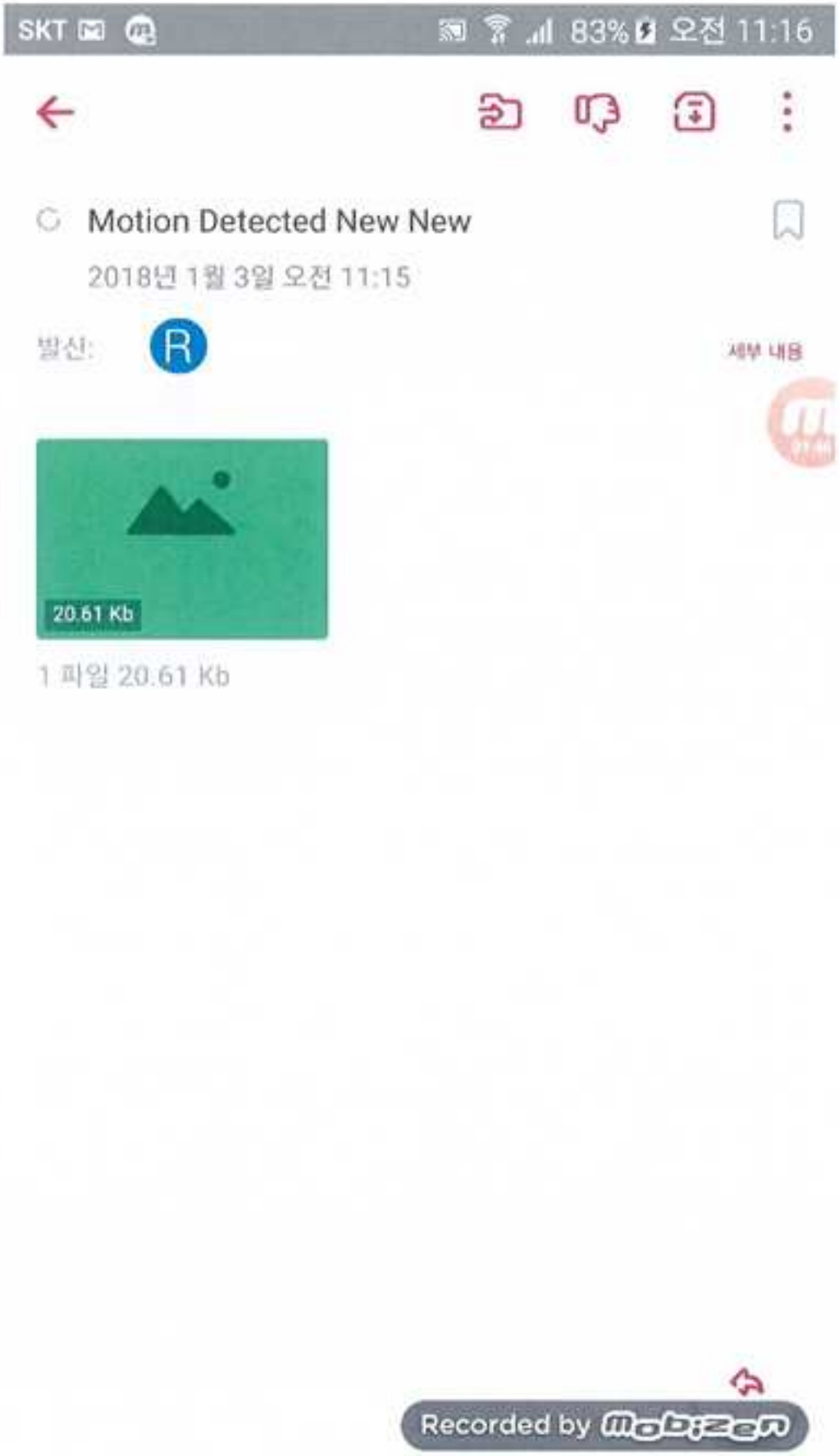
```
157             }
158         }
159
160         return 0;
161     }
```

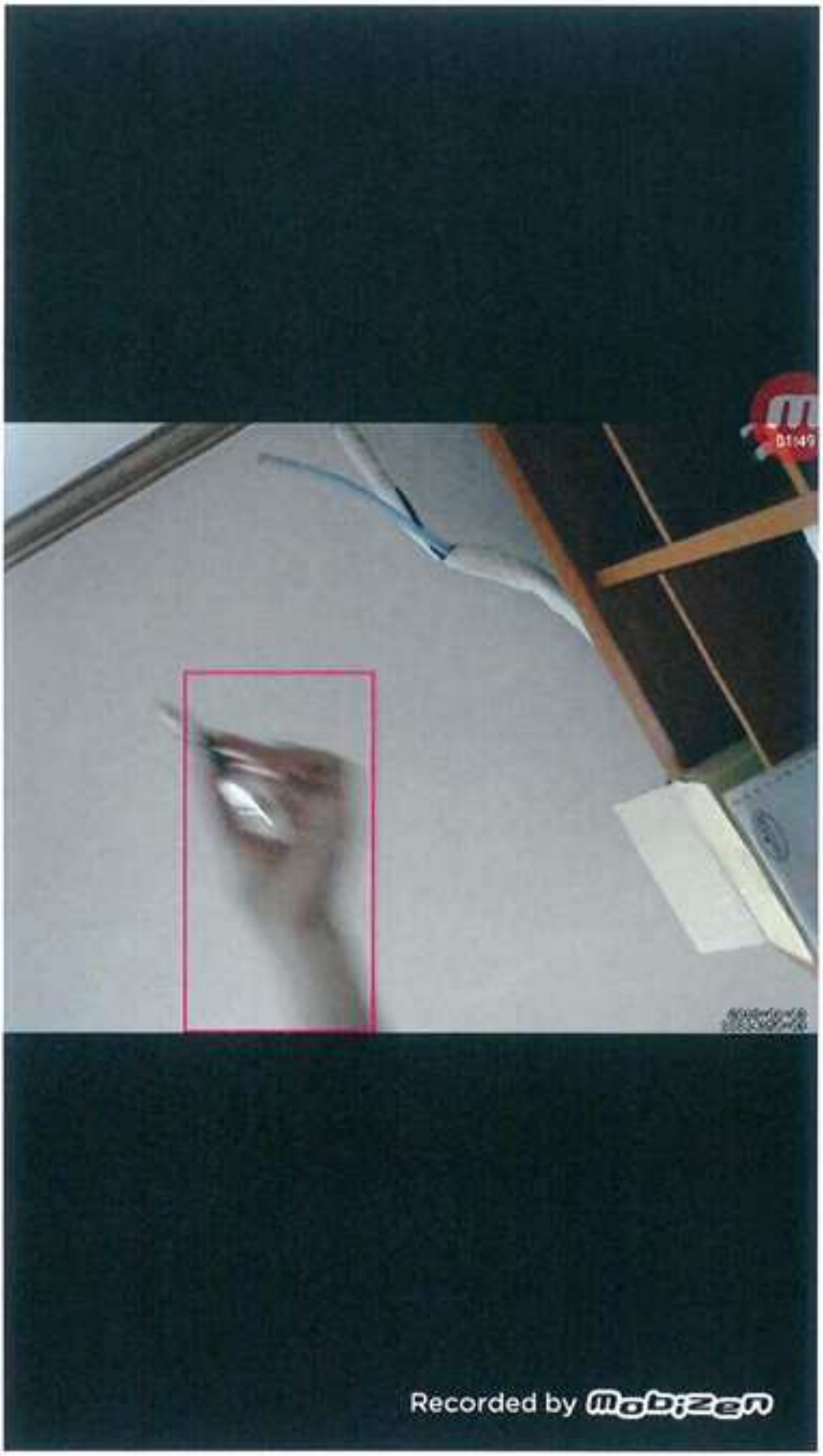

화면 녹화 시작
HomeCCTV
805
TALK
전화
카카오톡
Recorded by Mobizen

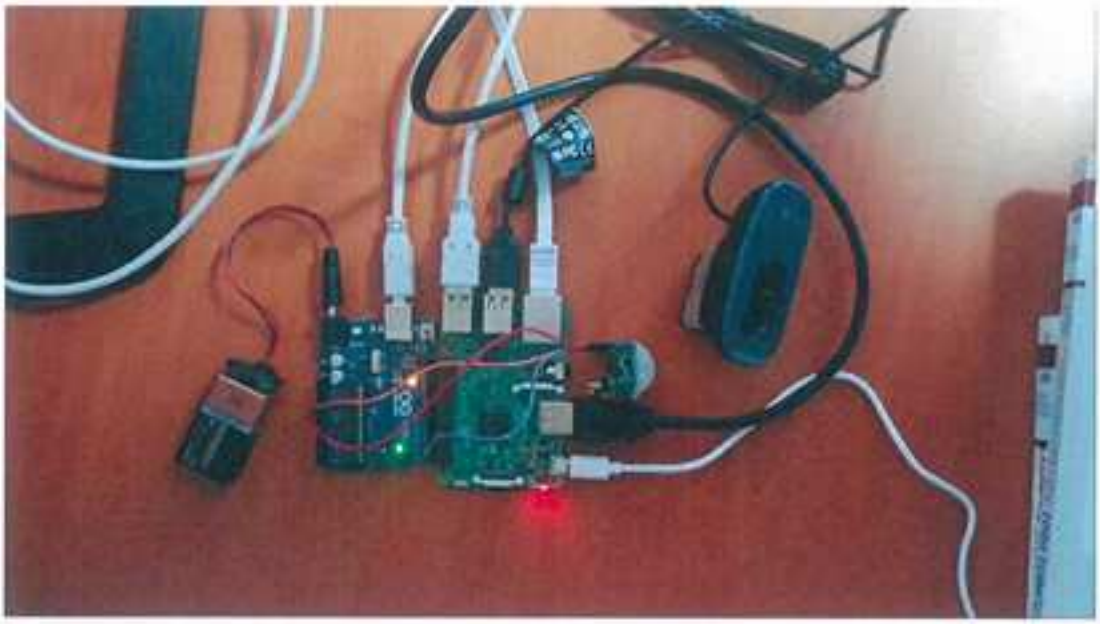

SKT
83% 오전 11:14
Motion 4.0.1 Running [1] Cameras
All
00:04
Recorded by Mobizen

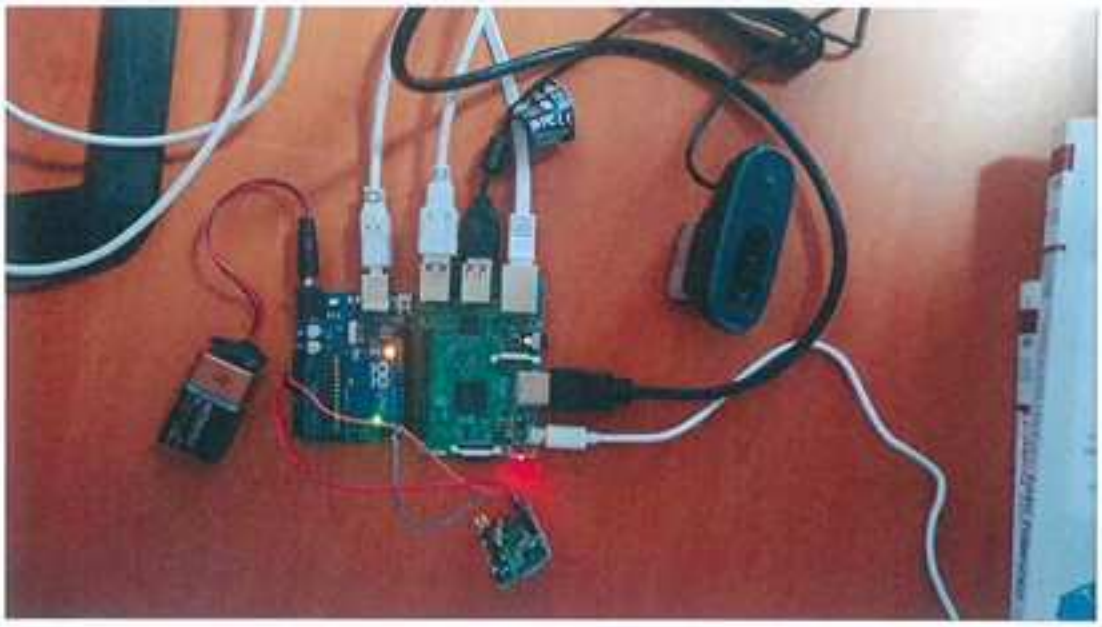



오전 11:16 1월 3일 수요일
편집
Wi-Fi
Home_2.4...
모바일 데이터
위치
세로
소리
자동
root
오전 11:16
Motion Detected New New
회신
보관
718
root
오전 11:15
Motion Detected New New
84
화면 녹화 중
오전 11:14
녹화를 정지하려면 터치하세요.
미디어 디바이스로 연결
기타 USB 옵션을 보려면 누르세요.
Recorded by Mobizen


SKT
83% 오전 11:16
Motion Detected New New
2018년 1월 3일 오전 11:15
발신:
R
세부 내용
20.61 Kb
1 파일 20.61 Kb
Recorded by Mobizen


Recorded by Mobizen





-라즈베리파이와 휴대폰으로 촬영한 동영상 파일 첨부

※ 분량 제한 없음
※ 참고문헌이 있을 시 정확히 명시