The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 20, No. 6, pp.115-121, Dec. 31, 2020. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2020.20.6.115
JIIBC 2020-6-17

딥러닝 기반의 얼굴과 제스처 인식을 활용한 원격 제어

Remote Control System using Face and Gesture Recognition based on Deep Learning

황기태*, 이재문, 정인환****

Kitae Hwang*, Jae-Moon Lee**, Inhwan Jung**

요 약 IoT 기술과 이 확산됨에 따라 얼굴 인식을 활용하는 다양한 응용들이 등장하고 있다. 본 논문은 딥러닝 기반의 얼굴 인식과 손 제스처 인식을 활용하는 원격 제어 시스템을 설계 구현한 내용을 기술한다. 얼굴 인식을 활용하는 응용 시스템은 카메라로부터 실시간으로 영상을 촬영하는 부분과 영상으로부터 얼굴을 인식하는 부분, 그리고 인식된 결과를 활용하는 부분으로 구성된다. 영상을 실시간으로 촬영하기 위해서 어디서나 장착 가능한 싱글보드 컴퓨터인 라즈베리파이를 이용하고, 서버 컴퓨터에는 FaceNet 모델을 활용하여 얼굴 인식 소프트웨어를 개발하고 OpenCV를 이용한 손 제스처 인식 소프트웨어도 개발하였다. 사용자를 알려진 사용자와 위험한 사용자 그리고 모르는 사용자의 3 그룹으로 구분하고, 얼굴 인식과 손 제스처가 모두 통과된 알려진 사용자에 대해서만 자동 도어락을 오픈하는 응용을 설계 구현하였다.

Abstract With the spread of IoT technology, various IoT applications using facial recognition are emerging. This paper describes the design and implementation of a remote control system using deep learning-based face recognition and hand gesture recognition. In general, an application system using face recognition consists of a part that takes an image in real time from a camera, a part that recognizes a face from the image, and a part that utilizes the recognized result. Raspberry PI, a single board computer that can be mounted anywhere, has been used to shoot images in real time, and face recognition software has been developed using tensorflow's FaceNet model for server computers and hand gesture recognition software using OpenCV. We classified users into three groups: Known users, Danger users, and Unknown users, and designed and implemented an application that opens automatic door locks only for Known users who have passed both face recognition and hand gestures.

Key Words : IoT, MQTT, Remote Control, Face Recognition, Gesture Recognition, Deep Learning

Ⅰ. 서 론

사물인터넷(Internet of Things; IoT)이란 사물 혹은 인간이 네트워크를 통해 긴밀하게 상호 작용할 수 있도록 연결하는 기술로 오늘날 인공지능 기술과 결합하여 다양한 응용 시스템들이 개발되고 있다[1,2].

*정회원, 한성대학교 컴퓨터공학부(교신저자)
**정회원, 한성대학교 컴퓨터공학부
접수일자 2020년 10월 19일, 수정완료 2020년 11월 19일
게재확정일자 2020년 12월 4일

Received: 19 October, 2020 / Revised: 19 November, 2020 / Accepted: 4 December, 2020
*Corresponding Author: calafk@hansung.ac.kr
Dept. of Computer Engineering, Hansung University, Korea

특별히 인공지능 기술을 활용한 얼굴 인식은 데이터베이스 검색을 통해 사람을 찾는 응용 뿐 아니라 CCTV나 소형 카메라를 원격으로 연결하여 실시간으로 얼굴을 인식하는 다양한 응용에 활용되고 있다. 이런 경우 원격지에 설치된 CCTV나 카메라를 장착한 소형 컴퓨터가 인공지능 소프트웨어를 실행시키기에 하드웨어적인 한계가 있기 때문에 별도의 인공지능 서버가 활용된다[3].

본 논문에서는 실시간으로 카메라 영상을 생성하는 원격 컴퓨터와 촬영한 영상을 인식하는 인공지능 서버, 그리고 인식된 얼굴 정보를 활용하는 IoT 제어 시스템, Crown을 설계 구현한 사례를 소개한다. 구체적으로 본 연구에서 개발한 얼굴 인식 시스템을 활용하여 알려진 사용자의 얼굴이 인식되면 자동으로 디지털 도어락을 여는 응용과 자동으로 출석을 체크하는 응용 시스템을 구현하였다. 본 논문에서 구현된 Crown 시스템은 새로운 기술을 사용하기보다는 기존 기술들을 통합하여 구현되었다.

라즈베리파이와 인공지능 서버 사이의 영상 전송을 제외하고 시스템 내 구성 요소들 사이의 데이터 통신은 메시지를 주고받는 MQTT 프로토콜[4]을 이용하였다.

스마트 팩토리, 스마트 가드닝 등 현재 사물인터넷 시스템은 전형적으로 인터넷이나 로컬 네트워크에 많은 센서들과 사용자들이 연결되고 이들 사이에는 메시지를 주고받는 방식으로 구축된다. MQTT는 사물인터넷 시스템에서 센서들과 사용자 컴퓨터들 사이에 데이터를 주고받는 표준 프로토콜로 많이 사용된다.

MQTT에서 주고받는 데이터를 메시지라고 부르며, 메시지를 기다리는 구독자(subscriber)와 메시지를 발송하는 발행자(publisher)로 구분 짓는 publish-subscribe 모델로 작동한다. 메시지 구독과 발행은 토픽(topic)을 기준으로 한다. 구독자와 발행자 사이에 메시지 전송을 중계하는 MQTT 브로커를 두고, 구독자는 여러 메시지 중에서 받고자 하는 메시지의 토픽(topic)을 MQTT 브로커에 등록하면, 발행자가 메시지를 토픽과 함께 MQTT 브로커에 전송할 때 MQTT 브로커는 등록된 가입자들 중 토픽을 기다리는 구독자들에게만 메시지를 전송한다. 즉 토픽을 기반으로 구독자와 발행자 사이에 메시지가 전송된다.

2장에서 Crown 시스템의 설계 내용을 설명하고, 3장에서 Crown 시스템의 시연에 설명하고 4장에서 결론을 맺는다.

II. 시스템 설계

1. 시스템 전체 구성

본 논문에서 설계 구현한 Crown 시스템은 그림 1과 같이 구성된다.

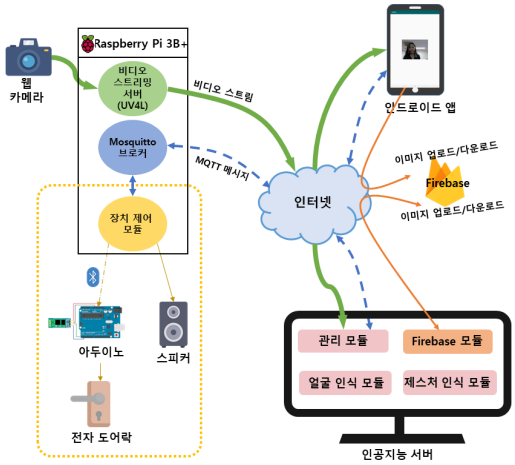

그림 1. Crown 시스템 구성
Fig. 1. System View of Crown

Crown은 총 4개의 부분으로 구성되는데, 웹 카메라를 장착하고 아두이노 등을 제어하는 단말 장치인 라즈베리파이, 인공지능 기반 얼굴 학습과 인식을 담당하는 서버 컴퓨터, 얼굴 사진을 공급하거나 웹 카메라를 모니터링하고 라즈베리파이에 연결된 장치를 제어하는 안드로이드 단말기이다. 본 연구에서는 아두이노가 전자 도어록을 제어하도록 회로를 구성하고 아두이노를 라즈베리파이와 블루투스로 연결하였다. 아두이노는 라즈베리파이로부터 지시를 받아 전자 도어락을 열고 닫는다. 얼굴 학습을 시키기 위해 준비된 사진들은 Firebase에 올려지고, 인공지능 서버는 Firebase에서 사진을 읽어 와서 학습을 진행한다. 또한 인공 지능 서버가 얼굴을 인식한 결과를 Firebase에 올리고, 안드로이드 앱에서는 Firebase로부터 결과를 받는다.

라즈베리파이는 무선 인터넷을 이용하여 인공지능 서버와 안드로이드 앱과 통신한다. 라즈베리파이에는 라즈비안 리눅스가 설치되고 웹 카메라의 비디오를 스트리밍하는 기능과 아두이노를 제어하는 기능이 구현되었으며, 시스템의 나머지 부분들과 메시지를 주고받기 위해 MQTT 브로커인 Mosquitto[5]가 설치되었다.

인공지능 서버에 실행되는 프로그램은 총 4개의 모듈로 구성된다. 안드로이드 앱이나 라즈베리파이로부터

MQTT 메시지를 수신하여 해석하고 다른 모듈을 작동시키는 관리 모듈, 얼굴 인식 프로그램 모듈, 손 제스처를 인식하는 모듈, 그리고 사진 이미지를 저장하는 구글의 Firebase 저장소에 사진 이미지를 저장하고 읽어오는 프로그램 모듈이다. 관리 모듈에서는 얼굴 학습 기능도 수행하면 얼굴 인식 모듈은 인공지능 플랫폼인 tensorflow[6]를 활용하여 구현되었다.

안드로이드 단말기에 구현된 앱은 웹 카메라로부터 실시간 스트리밍을 보여주는 기능과 사용자의 지시에 따라 라즈베리파이에 연결된 전자 도어락을 열고 닫도록 하는 제어 기능이 구현되었다.

2. MQTT 메시지 통신

Crown 시스템에서는 비디오 스트리밍 데이터를 제외한 나머지 데이터 통신은 모두 토픽에 기반하여 MQTT 프로토콜로 텍스트 메시지를 주고받는다. 메시지를 중계하는 MQTT 브로커는 라즈베리파이에 설치되며 본 연구에서 사용되는 토픽은 표 1과 같다.

표 1. MQTT 토픽들
Table 1. MQTT topics

토픽	의미	송신자	수신자
token	FCM 알림 용 안드로이드 기기 토큰	안드로이드 앱	관리 모듈
addgroup	사용자 이름과 그룹	안드로이드 앱	관리 모듈
download/start	사진 다운로드 시작	관리 모듈	Firebase 모듈
download/finished	사진 다운로드 완료	Firebase 모듈	관리 모듈
gesture/key	제스처 설정	안드로이드 앱	관리 모듈
gesture/success	제스처 성공	얼굴 인식 모듈	안드로이드 앱
gesture/fail	제스처 실패	얼굴 인식 모듈	안드로이드 앱
doorlock	도어락 제어	얼굴 인식 모듈	라즈베리파이의 장치 제어
state/doorlock	도어락 상태	라즈베리파이의 장치 제어	안드로이드 앱
speaker	스피커 제어	얼굴 인식 모듈	라즈베리파이의 장치 제어
state/speaker	스피커 상태	라즈베리파이의 장치 제어	안드로이드 앱
warning/danger	Danger 인물 인식	얼굴 인식 모듈	안드로이드 앱
warning/unknown	Unknown 인물 인식	얼굴 인식 모듈	안드로이드 앱

3. 스트리밍

비디오 스트리밍을 위해서는 UV4L 소프트웨어[7]를 이용하였다. 라즈베리파이에 UV4L 서버를 설치하여 웹 카메라로부터 실시간 영상을 스트리밍하도록 하였다. UV4L 서버는 HTTP 프로토콜로 실시간 영상을 전송한다. 인공 지능 서버에서는 OpenCV 라이브러리를 활용하여 '라즈베리파이의 IP주소:8090/stream' 으로 접속하여 라즈베리파이로부터 실시간 카메라 영상을 받는다. 그리고 이 실시간 영상으로부터 얼굴을 인식한다. 안드로이드 앱의 경우 웹 브라우저를 활용하여 동일한 URL을 활용하여 실시간 카메라 영상을 출력한다.

4. 인공지능 서버

가. 사용자 그룹

사용자는 Known, Danger, Unknown의 3 그룹으로 분류하고 얼굴 학습 시에 Known이나 Danger 중 하나로 인식시킨다. Known 그룹은 사용자의 가족, 친구, 지인 등 주변 인물들을 등록해 놓은 그룹이고, Danger 그룹은 주변에 사는 범죄자와 같은 위험한 인물이라고 생각되는 사람들을 등록해 놓은 그룹이다. 인식된 사람이 이 두 그룹에도 속하지 않는다면 Unknown 그룹으로 판단한다.

나. 서버 소프트웨어

서버 소프트웨어는 4개의 모듈로 구성된다. 첫째 관리 및 얼굴 학습 모듈로서 Crown 시스템의 다른 부분으로부터 MQTT 메시지를 받아 해석하고 처리하며 얼굴 학습 기능도 수행한다. 둘째, Firebase 입출력 모듈로서 Firebase로부터 얼굴 이미지를 읽어오거나 얼굴 인식의 결과 Danger나 Unknown 그룹의 경우 실시간 영상의 프레임 이미지를 Firebase에 올리고 MQTT 메시지를 안드로이드 앱에 보내 안드로이드 앱의 사용자가 볼 수 있게 한다. 셋째, 얼굴 인식 모듈로서 스트림 영상 속에 담긴 얼굴을 인식하여 Known, Danger, Unknown으로 구분한다. 넷째, 제스처 인식 모듈로서 얼굴 인식 후 정해진 손가락 제스처를 인식하여 한 번 더 사람을 확인한다.

다. 얼굴 학습

얼굴 인식이 이루어지기 위해서는 사용자들의 얼굴학습이 먼저 이루어져야 한다. 본 연구에서는 얼굴 학습을 위해 한 사용자에 대해 약 40장의 얼굴 이미지를 인공지능 서버로 보내 학습시킨다.

40장의 얼굴 사진을 준비하는 과정은 안드로이드 앱에서 이루어진다. 안드로이드 카메라를 이용하여 사람 얼굴을 왼쪽 오른쪽 올려가면서 다양한 형태의 얼굴 모습을 40장 촬영하고 이들을 구글 클라우드의 Firebase 데이터베이스에 저장한다. 그리고 인공 지능 서버에게 addgroup 메시지를 보내 새로운 얼굴이 준비되었으니 학습하도록 지시한다.

인공지능 서버는 addgroup 메시지를 받게 되면 Firebase에서 40장의 사진을 다운받고 얼굴 부분만 이미지 파일에 저장하고 이 파일들을 이용하여 학습시킨다. 그림 2는 인공지능서버가 자동으로 얼굴부분만 끊어낸 40개의 이미지사례이다.

그림 2 얼굴 학습에 사용된 40개의 이미지
Fig. 2. 40 face images for face learning

얼굴 학습 모듈은 이미 얼굴 인식이 가능하도록 학습된 FaceNet 모델[8]에 40장의 사진 속 얼굴 데이터를 학습시켜 새로운 FaceNet 모델을 생성한다. 그림 3은 40장의 사진 샘플로 학습시키는 과정이다. 그림 3에서 왼쪽 사진은 초기 사진이며, 중간 사진은 인공지능서버가 얼굴 부분만 끊어내어 만든 사진이다.

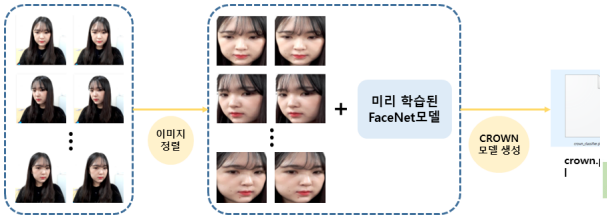

그림 3. 얼굴 학습 과정
Fig. 3. 3 Process of Face Recognition

라. 얼굴 인식

얼굴 인식은 라즈베리파이의 스트리밍 서버로부터 실시간으로 영상을 받아와 프레임 별로 인물을 인식하여 이름을 결정하고, Known, Danger, Unknown 중 어디에 속하는지 결정하는 과정이다. 얼굴 인식 모델은 tensorflow 기반의 FaceNet 모델을 사용하였다.

FaceNet 모델은 탐지와 선처리, 특징 추출, 얼굴 매칭의 4단계로 얼굴 인식을 진행하는데, 탐지의 과정은 이미지에서 얼굴이 들어 있는 박스를 결정하는 과정으로 MTCNN(Multi-Cascade Neural Network)를 활용하여 얼굴이 들어있는 부분을 탐지한다. 두 번째 선처리 단계에서는 특징 데이터를 기반으로 이미지에서 얼굴이라고 판단되는 부분만 잘라낸다. 세 번째 특징 추출의 단계에서는 128차원의 얼굴 특징을 추출하여 벡터화하고 유클리드 거리를 이용하여 가장 가까운 거리의 데이터를 찾는 triplet loss 기법을 사용한다. 현재 이미지의 얼굴 벡터와 거리를 계산하는데 얼굴이 닮지 않을수록 1에 가깝게 나오고 닮을수록 0에 가까운 값이 나온다.

본 논문에서는 얼굴이 비슷한 사용자들이 다소 있을 수 있기 때문에 count 변수를 사용하여 한 프레임마다 판별된 인물에 대해 count를 1씩 증가시키고 count가 15가 되었을 때 그 인물이라고 판단하는 방법을 사용하였다. 전 프레임에는 있었지만 현 프레임에는 없는 인물에 대해서는 count를 1 줄이는 방법을 사용하였다.

인물이 감지되면 이 인물이 Known에 속하는지 Danger에 속하는지 판별하여 이름을 결정하고, 두 그룹에 속하지 않는 경우 Unknown 그룹으로 판단한다.

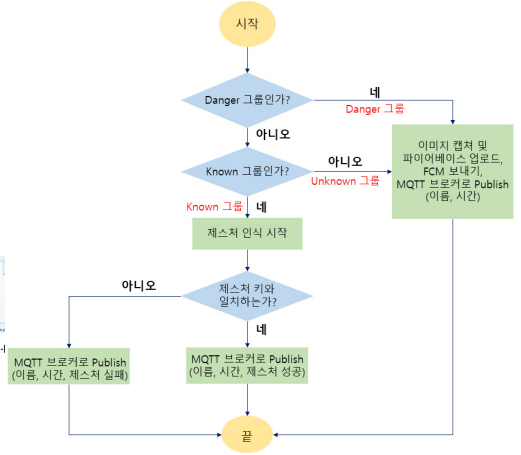

그림 4. 얼굴과 제스처 인식 알고리즘
Fig. 4. Algorithm for recognition of face and hand gesture

Danger나 Unknown으로 인식된 경우, 곧바로 해당 프레임을 Firebase에 업로드한 뒤 '인물 이름'과 '인식된 시간' 등의 정보를 안드로이드 앱에게 보내 Danger나 Unknown의 출현을 알린다.

안드로이드 앱은 이 메시지를 받으면 시간 정보를 비교하여 Firebase에서 이미지를 가져 온다. 그리고 사용자가 원할 때 스크린에 출력한다.

Known으로 판단된 경우, 제스처 인식을 통해 다시 한 번 인물을 검증한다. 제스처 인식이 성공한 경우 안드로이드 앱으로 인물 정보와 시간 정보, 그리고 제스쳐 인식 성공 메시지를 전송한다. 제스처 인식이 실패한 경우 안드로이드 앱으로 인물 정보와 시간 정보, 그리고 제스쳐 인식 실패 메시지를 전송한다. 전체 과정은 그림 4와 같다.

마. 손 제스처 인식

손 제스처 인식 모듈은 OpenCV 라이브러리를 활용하여 작성하였다. 제스처 인식 모듈은 동영상의 각 프레임에서 배경을 삭제한 후 모폴로지 연산을 통해 남아 있는 객체를 뚜렷하게 만든다. 모폴로지 연산은 그림 5와 같이 opening 과정을 통해 밝은 노이즈를 제거하고 closing 연산을 통해 어두운 노이즈를 제거하여 전체 노이즈가 제거된 객체를 생성한다. 손 제스처는 미리 정해 둔다.

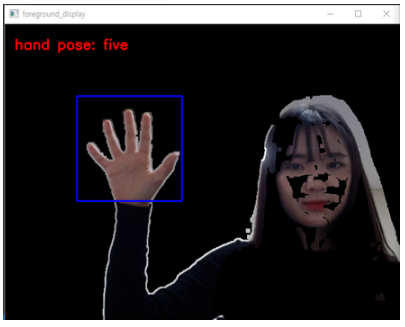

그림 5. 모폴로지 연산 결과
Fig. 5. Result of Morphology

5. IoT 응용

본 논문에서는 얼굴 인식의 응용 사례로 Known 그룹의 사용자가 손 제스처를 통과하였을 때 전자 도어락을 자동 오픈하는 사례를 구현하였다. 시중에서 판매하는 보통의 전자 도어락을 아두이노 보드에 연결하여 아두이노에서 전자 도어락을 열 수 있도록 구현하였다.

아두이노 보드에는 블루투스 실드를 장착하여 라즈베리파이와 통신하도록 하였다. 라즈베리파이의 오디오 연결단자에 스피커를 연결하여 도어락이 열리거나 Danger나 Unknown 그룹에 속한 사용자이거나 손 제스처가 틀렸을 경우 실패 효과음을 내도록 하였다.

6. 안드로이드 앱

안드로이드 앱에는 카메라 모니터링, 그룹 관리, 로그 확인, IoT 장치 제어의 4가지 기능을 구현하였다. 안드로이드 앱은 이미지 데이터를 다루기 위해 C++ OpenCV 라이브러리를 활용하였다. 안드로이드 앱은 자바로 작성되었으며 JNI 기법으로 OpenCV의 C++ 라이브러리를 호출하도록 구현하였다.

가. 카메라 모니터링

안드로이드 앱을 통해 언제 어디서든 웹 카메라 앞의 상황을 모니터링할 수 있도록 웹 뷰를 개발하였다. 이 웹 뷰는 라즈베리파이의 스트리밍 서버에 직접 접속하여 스트리밍 화면을 실시간으로 보여준다.

나. 그룹 관리

안드로이드 앱의 그룹 관리 기능은 스마트폰 카메라를 이용하여 40여장의 사진을 직접 촬영하고 얼굴 부분이 있는지 검사하여 얼굴 부분이 있을 경우에만 이미지 파일로 저장하고 얼굴이 없는 사진은 다시 촬영하도록 개발하였다.

다. 로그 관리

로그는 인공지능 서버가 얼굴을 인식하는 경우, 이름, 시간, 그룹의 정보를 담아 Firebase에 올려놓은 정보이다. 안드로이드 앱은 로그를 다운받아 사용자가 원할 때 볼 수 있도록 한다. 로그에서 시간은 출입, 등장 등 이벤트가 일어난 시각을 의미하고, 그룹은 Known의 경우 초록색, Danger는 빨간색, Unknown은 파란색으로 색을 표시해 사용자가 보다 쉽게 로그를 조회할 수 있도록 하였다. 또한 Known 그룹의 경우 제스처 인식 성공 여부도 함께 표시하였다. Danger나 Unknown 인물의 로그는 해당 로그를 클릭했을 때 등장한 시점의 이미지를 보여주어 사용자에게 어떤 인물이 등장했는지 시각적인 정보를 제공하도록 구현했다.

III. 테스트 및 실행

1. 시스템 구현

그림 6은 Crown 시스템이 실제 구현된 모습을 보여준다. Full HD 웹카메라, Raspberry3B+, Arduino Uno와 전자 도어락이 사용되었고 인공지능 서버로는 4.2GHz의 Intel Core i7-7700K CPU와 32GB RAM과 NVIDIA GeForce GTX 1060 3GB 그래픽 카드를 갖춘 HP EliteDesk 800 G3 TWR PC를 사용하였다. 인공지능 서버에는 Windows10을 탑재하였다.

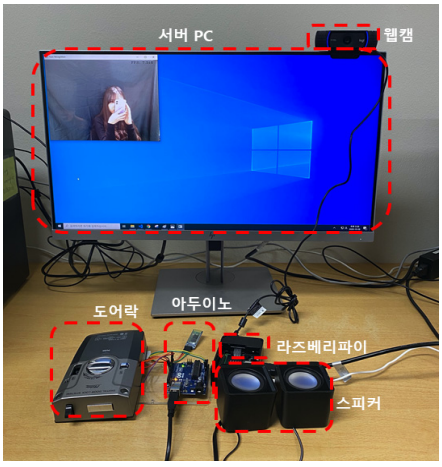

그림 6. Crown 시스템 구현
Fig. 6. Crown system implemented

2. 테스트

Crown 시스템은 먼저 그림 7와 같이 안드로이드 앱을 이용하여 Known 그룹의 사용자 사진을 정면, 왼쪽, 오른쪽 등 연속하여 촬영하여 Firebase에 업로드한다. Danger 그룹은 실험을 위해 임의로 사진을 이용하였다. 안드로이드 앱에는 사진 촬영 시 얼굴이 위치할 영역을 점선으로 알려주어 얼굴인식이 잘되도록 하였다.

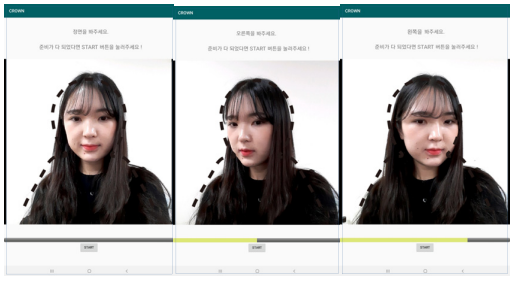

그림 7 안드로이드 앱으로 얼굴 촬영
Fig. 7. Face shoot using Android app

안드로이드 앱은 addgroup 메시지를 인공지능 서버의 관리 모듈로 보내 얼굴 학습을 요청한다. 인공지능 서버의 관리 모듈이 이 메시지를 받으면 Firebase로부터 얼굴 사진을 다운받고 학습을 진행하여 새로운 인식 모델을 만든다.

그 후부터 인공지능서버가 라즈베리파이로부터 얼굴이 포함된 동영상 스트림을 받게 되면 Known, Danger, Unknown 중 하나로 인식하고 Known으로 판단되면 손 제스처를 즉각적인 인식한다. 이 두 과정이 모두 통과하면 Known 그룹의 얼굴 인식 성공을 라즈베리파이에게 알리고 라즈베리파이는 아두이노를 제어하여 전자 도어록을 연다. 그림 8은 인공지능서버가 얼굴과 제스처를 인식하는 과정을 보여준다.

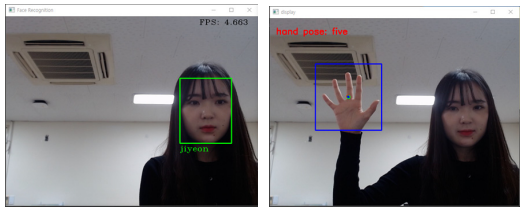

그림 8. 인공지능서버의 얼굴과 제스처 인식
Fig. 8. Recognition of face and hand gesture by AI server

그림 9는 안드로이드 앱의 실행 샷으로, 왼쪽은 인공지능서버가 Danger 그룹의 사람을 인식하여 해당 프레임을 Firebase에 올린 로그를 다운받아 보여주는 상황이며, 오른쪽은 라즈베리파이 카메라의 영상을 실시간 모니터링하는 상황이다.

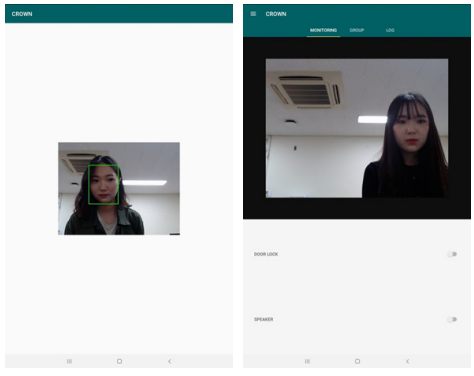

그림 9 안드로이드 앱
Fig. 9. Android app running

IV. 결 론

IoT 시스템은 네트워크를 통해 사물들이 메시지를 주고받는 단순한 응용에서 인공지능이 결합되는 형태로 진화하고 있다. 본 논문에서는 IoT 시스템에 인공지능이 결합되는 응용 시스템 Crown을 만든 사례를 소개한다. Crown 시스템은 새로운 기술을 고안하기 보다는 기존의 얼굴 인식 딥러닝 기술과 MQTT 기반의 메시징 기법 등을 결합하여 구현되었다. Crown은 라즈베리파이에 FHD 급의 카메라를 연결하여 실시간으로 영상을 스트리밍하여 인공지능서버가 얼굴과 손 제스처를 인식하도록 구현하였다. 알려진 사용자를 인식한 경우 라즈베리파이에 연결된 아두이노를 통해 전자 도어락을 열도록 하였고 로그를 통해 안드로이드 스마트폰 사용자에게 이미지가 전송되도록 하였다.

References

[1] SeoHyung Kim, "IoT : Internet of Things Technology", IEIE Magazine, Vol. 43, No.3, pp. 64-71, 2016

[2] Kitae Hwang. "Design and Implementation of Dashboard Author and Viewer for IoT Systems based on MQTT", JIIBC [Internet]. 2018 Oct 31;18(5):31-7. Available from: https://doi.org/10.7236/JIIBC.2018.18.5.31

[3] S. B. Calo, M. Touna, D. C. Verma and A. Cullen, "Edge computing architecture for applying AI to IoT", 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, 2017, pp. 3012-3016, doi: 10.1109/BigData.2017.8258272.

[4] https:/mqtt.org

[5] https://mosquitto.org/

[6] L. Yuan, Z. Qu, Y. Zhao, H. Zhang and Q. Nian, "A convolutional neural network based on TensorFlow for face recognition", 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, 2017, pp. 525-529, doi: 10.1109/IAEAC.2017.8054070.

[7] http://www.linux-projects.org/uv4l/

[8] Florian Schroff, Dmitry Kalenichenko, James Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815-823

※ 본 연구는 한성대학교 교내 학술 연구비를 지원받았음.

저 자 소 개

황 기 태(종신회원)*

- 서울대학교 컴퓨터공학과 학사
  서울대학교 컴퓨터공학과 석사
  서울대학교 컴퓨터공학과 박사
- 1994년 ~ 현재 한성대학교 컴퓨터공학과 교수
- 경력
  University of Florida 방문 교수
- 관심분야 : 모바일 시스템, IoT, 인공지능

이 재 문(정회원)

- 한양대학교 전자공학과(학사)
  한국과학기술원 전기및전자공학과 (석사)
  한국과학기술원 전기및전자공학과 (박사)
- 경력
  한국통신 연구개발단
- 관심분야 : 기계학습, 게임프로그래밍, 감성컴퓨팅

정 인 환(정회원)

- KAIST 정보및통신공학과 박사
- 삼성전자 수석연구원
- 한성대학교 컴퓨터공학부 교수
- 관심분야 : IoT, 분산시스템, 모바일시스템