논문 21-46-12-29 The Journal of Korean Institute of Communications and Information Sciences '21-12 Vol.46 No.12
https://doi.org/10.7840/kics.2021.46.12.2340

젯슨 나노 기반의 얼굴 인식 시스템 구현

장 일 식•, 박 구 만°

Implementation of Jetson Nano Based Face Recognition System

Il-Sik Chang•, Goo-Man Park°

요 약

얼굴 인식은 최근 딥러닝의 발전으로 많은 성능이 개선이 이루어졌다. 얼굴인식은 얼굴 검출, 얼굴 정렬, 얼굴 인식의 순서로 처리한다. 본 논문에서 학습데이터로 한국인 얼굴로 구성된 K-Face 데이터 셋을 이용하여 얼굴 인식을 학습하였으며, 얼굴 검출에는 MTCNN을 사용하였다. 얼굴 정렬은 OpenCV를 사용하였으며, 얼굴 인식을 위해서 백본은 Resnet-50을 사용하였고, 손실함수는 ArcFace 방법으로 학습하였다. 또한 KISA의 얼굴인증을 받기 위한 알고리즘 구현하였고, 임베디스 시스템인 젯슨 나노에 얼굴 인식 시스템을 구현하였다. KISA의 얼굴 인식 인증을 위해선 DLL로 구현해야 하기 때문에 OpenCV를 사용하여 작업을 하였다. 딥러닝 학습을 위해선 Python과 텐서플로우를 이용하여 진행하였고, KISA인증을 위한 학습된 망을 OpenCV를 이용하여 추론하는 방법을 사용하였다. 임베디스 시스템인 젯슨 나노에서 실시간성을 위해 TensorRT를 사용하여 속도를 향상 시켰다. 동작 테스트를 위해 Qt5를 이용한 GUI 프로그램을 개발 하였으며, 테스트를 위한 영상은 비디오 파일, RTSP, CSI 카메라 등을 선택적으로 사용가능하도록 구현하였다. 실험 결과 구현된 시스템이 10fps의 속도로 얼굴 검출 및 인식이 가능함을 확인하였다.

Key Words : Face recognition, MTCNN, ArcFace, Jetson Nano, TensorRT

ABSTRACT

Face recognition has improved a lot with recent advances in deep learning. Face recognition is processed in order of face detection, face alignment, and face recognition. In this paper, We used the K-Face dataset consisting of Korean faces for training the proposed face recognition model, and MTCNN is used for face detection. OpenCV was used for face alignment, Resnet-50 was used for the backbone for face recognition, and the loss function was learned by ArcFace method. Furthermore, In addition, an algorithm to receive KISA's face authentication was implemented, and a face recognition system was implemented in Jetson Nano, an embedded system. For KISA's face recognition authentication, I had to implement it as a DLL, so I used OpenCV. For deep learning learning, Python and TensorFlow were used to infer the learned network for KISA authentication using OpenCV. We improved the speed by using TensorRT for real-time performance in Jetson Nano, an embedded system. A GUI program using Qt5 was developed for behavior testing, and images for testing were implemented to selectively use video files, RTSP, and CSI cameras. Experiments confirm that the implemented system is capable of face detection and recognition at a rate of 10 fps.

※ 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2017-0-00217, 투명도와 레이어 가변형 실감 사이니지 기술 연구)

• First Author : Dept. of Information Technology and Media Engineering, The graduate School of Nano IT Design Fusion, Seoul National University of Science and Technology, ischang@hanmail.net, 정회원

° Corresponding Author : Dept. of Electronic IT Media Engineering, Seoul National University of Science and Technology, gmpark@seoultech.ac.kr, 종신회원

논문번호 : 202108-214-D-RN, Received August 23, 2021; Revised September 7, 2021; Accepted September 8, 2021

Ⅰ. 서 론

보안 분야는 지문 인식, 홍채 인식, 음성 인식, 얼굴 인식 등 다양한 기술이 사용가능 하다. 특히 얼굴인식은 카메라의 영상을 통해 인식하는 방법으로 CCTV 혹은 카메라 센서를 사용하여 인식이 가능하다. 얼굴 인식에서 전통적인 방법은 얼굴 이미지에서의 고유값 성분을 통한 방법[1], 하르 특징을 사용한 방법[2], Fisherfaces[3] 등이 사용되었으나, 연산하는데 오랜 시간이 소요되거나 정확도가 높지 않은 문제가 있다. 최근 딥러닝 기술이 발전함으로써 영상인식 기술의 향상이 두드러지고 있다. 딥러닝 얼굴인식에 처음으로 접목된 연구는 DeepFace[4] 연구이다. 이후 VGG-Face[5], DoopID[6] 연구 등 다양한 딥러닝 네트워크 구조가 제안되었다. 본 논문에서는 메트릭 러닝(Metric Learning) 방법으로 손실 함수(Loss function)의 재정의를 통해 분별력 있는 특징을 학습하기 위한 소프트맥스 변형 손실함수인 ArcFace[7] 방법을 사용하여 얼굴 인식을 수행하였다. 얼굴 인식은 화면에서 얼굴의 위치를 찾는 얼굴 검출 부분과 얼굴의 촬영 각도나 회전되었을 경우 얼굴인식의 정확도를 올려주기 위해 정면 얼굴로 동일하게 맞춰주는 얼굴 정렬 부분, 정렬된 얼굴을 일정한 크기로 정규화하여 얼굴을 인식하는 얼굴 인식부분으로 나뉜다. 얼굴 검출은 얼굴 검출을 위한 다양한 객체 검출방법이 존재한다. 실시간에 적합한 YOLOv3[8], YOLOv4[9] 등을 사용할수 있으나 얼굴 정렬을 위한 랜드 마크를 추가로 찾아주는 방법이 필요하다. 본논문에서는 얼굴의 랜드마크 위치 값을 얻을 수 있는 MTCNN(Multi-Task cascade Convolutional Neural Network)[10] 방법을 이용하여 구현하였다. 얼굴 정렬은 OpenCV[11] 기반으로 구현하였고, 얼굴 인식의 백본으로 Resnet-50[12]을 사용하였다. 얼굴 인식을 위한 손실함수는 소프트 맥스 손실함수, 거리기반 손실함수, 앵귤러마진 기반 손실함수로 나눌수 있다. 본 논문에서는 손실 함수로는 앵귤러마진 기반 손실함수인 ArcFace 방법을 사용하였다. 또한 한국정보보호진흥원(KISA)[13]에서 시행하는 얼굴인식 알고리즘 성능시험에 관한 내용 및 수행 방법에 대하여 알아보고, 임베디드 보드인 엔비디아의 젯슨 나노(Jetson Nano)[14]를 사용하여 얼굴인식을 구현하였다. 젯슨 나노 기반의 연구는 CCTV, 스마트 도어등[15][16]등 다양한 분야에 적용이 가능하다. 또한 GPU 상에서 추론 속도를 높이기 위해서 엔비디아 GPU 상에서 추론 속도를 향상시키는 TensorRT[17]를 사용하여 속도를 개선하여 얼굴 인식을 수행하였다. 딥러닝 학습은 PC에서 수행을 진행하고, 학습된 모델을 통하여 추론은 KISA 얼굴인식 알고리즘 성능 시험에 OpenCV, 젯슨 나노에 TensorRT를 사용하여 구현하였다.

Ⅱ. 관련 연구

2.1 얼굴 검출

얼굴 검출 기술은 중요 생체 인식 기술의 하나로써, 영상에서 얼굴의 위치를 찾는 기술을 말한다. 얼굴 검출은 얼굴 인식이나 사람의 감정분석 등의 고차원적인 기술을 구현하기 위해서 사용된다. 얼굴 검출을위한 방법으로는 S3FD[18], RetinaFace[19], MTCNN등이 있다. RetinaFace는 FPN[20]을 기본으로 Multi-task 학습을 하고, 얼굴 검출과 얼굴의 랜드마크를 동시에 추론이 가능하다. MTCNN은 CNN을 이용해 얼굴 검출 정확도를 올렸으며 얼굴 검출, 바운딩 박스 회귀(Bounding box regression), 얼굴 정렬(Face alignment)를 동시에 학습시킬수 있는 조인트 학습(Joint learning) 방식을 제안하였다. 빠른 속도와 높은 정확도를 나타내며 세 개의 태스크를 학습한 P-net, R-net, O-net이라는 세 CNN을 차례로 통과하는 Cascade 모델을 사용한다. P-net(Proposal network)은 이미지에서 얼굴을 찾아내는데 중점을 둔 네트워크이다. FCN (Fully connected layer)로 이루어진 것이 특징이고, 바운딩 박스 회귀 벡터와 후보 영역을 얻어 NMS(Non-maximum suppression) 알고리즘으로 높은 정확도의 후보영역만 남도록 추려낸다. 남은 후보 영역의 얼굴 분류, 바운딩 박스 회귀, 얼굴 랜드마크 지역화 값을 다음 네트워크로 전달한다. R-net(Resize network)은 P-net에서 찾아낸 후보 영역을 추려내는데 중점을 두었다. 바운딩 박스 회귀 벡터와 후보 영

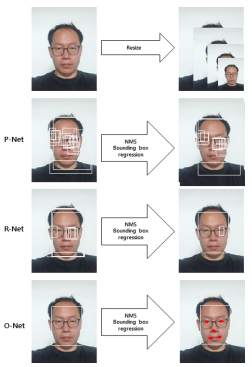

그림 1. MTCNN의 핵심아이디어
Fig. 1. MTCNN Main Idea

역을 구하고, 이들을 NMS 알고리즘으로 높은 정확도의 후보영역만 남도록 추려낸다. 남아있는 얼굴 후보 영역의 얼굴 분류, 바운딩 박스 회귀, 얼굴 랜드마크 지역화 값을 다음 네트워크로 전달한다. O-net(Output network)은 이전 단계에서 찾아낸 후보 영역에서 얼굴 랜드마크를 찾아내는데 중점을 둔 네트워크이다. 최종적인 얼굴 후보 분류, 바운딩 박스 회귀, 얼굴 랜드마크 지역화 값을 얻는다. 그림1[10]은 MTCNN의 핵심 아이디어를 나타낸다.

2.2 얼굴 인식

얼굴 인식 기술은 컴퓨터 비전의 주요 연구 분야 중 하나로서, 본 논문에서는 메트릭 러닝 학습 방법이 적용한다. 메트릭 러닝은 비유사한 특징들을 멀리 떨어지도록 학습하여 학습하지 않은 이미지에 대해서도 인식이 가능하다. 모델은 학습 부분에 CNN을 통해 들어온 이미지가 특정 클래스로 구분될지 학습되도록 뒷부분에 소프트 맥스 층(Softmax layer)이 포함되어 있다. 추론과정에서는 뒷부분의 소프트 맥스 층을 제거하고 CNN에서 통과하는 특징 벡터를 결과물로 낸다. 그리고 검증(Verification)이나 식별(Identification)은 이 모델을 통과해서 추출한 특징 벡터의 값을 유클리디안 거리(Euclidean distance)를 이용해 얼굴 인식의 결과를 판별하게 된다. 그림 2는 얼굴 검증과 얼굴 식별을 보여준다. 얼굴 검증은 시스템에 입력된 두 이미지 속 인물이 동일인인지 검증하는 것이고, 얼굴 식별은 내부 데이터베이스에 미리 저장된 인물 중 가장 유사한 인물을 식별하는 기술이다.

얼굴 인식 모델의 훈련, 추론 단계에서 사진 속의 얼굴 위치가 다르거나 그 촬영 각도 혹은 얼굴의 각도가 회전되었을 경우 얼굴의 형태가 다르면 얼굴 인식 정확도가 낮아질 수 있다. 그리하여 이미지에서 얼굴 영역을 찾아 동일한 형태인 정면 얼굴을 추출하는 얼굴 정렬 과정이 필요하다. 얼굴 정렬 과정은 첫 번째로 시스템에 입력된 이미지에서 얼굴 영역을 찾는 얼굴 검출이다. 두 번째로 눈, 코, 입술 등 얼굴의 특징

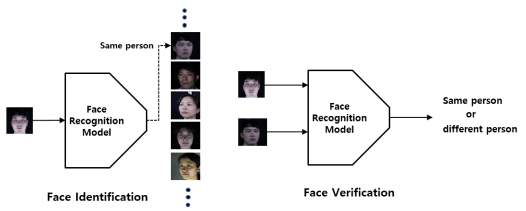

그림 2. 얼굴 검증과 얼굴 식별
Fig. 2. Face Identification and Face Verification

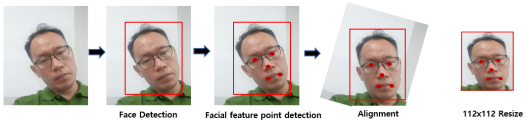

그림 3. 얼굴 정렬 과정
Fig. 3. Process for face alignment.

을 나타내는 점인 랜드마크을 이용해 얼굴을 동일한 형태인 정면 얼굴로 변경하고, 동일한 크기로 변경하는 얼굴 정렬의 단계를 거친다. 본 논문에서는 MTCNN을 통하여 얼굴 검출 및 랜드마크를 추출하고, OPENCV를 이용하여 정면 얼굴로 변환후 일정한 크기로 변경한다. 그림 3은 얼굴 정렬에 대한 과정을 나타낸다.

얼굴 인식 아키텍처는 얼굴 검출단계를 거친 후, DCNN(Deep Convolutional Neural Network)[19]을 활용하여 특징 벡터 (Feature vector)를 학습시킨다. 특징 벡터는 얼굴 인식 모델은 수십만명의 인물에 대한 얼굴 이미지로부터 인물을 잘 구분하는 함축된 특징 벡터가 된다. 얼굴 인식의 주요과제는 분별력 있는 특징 벡터를 구하기위해서 손실 함수를 통해 유사도를 비교하여 클래스를 분별하는 것이다. 얼굴 인식 문제의 경우에는 분류 문제와 성격이 달라서 소프트 맥스만으로 충분한 성능을 내기에 어려움이 존재한다. 분류 문제의 경우 학습 과정에서의 클래스의 종류와 테스트 과정에서의 클래스의 종류가 일치하기 때문에 닫힌집합 (Closed-set) 문제로 취급된다. 하지만, 얼굴 인식의 경우는 기존 분류 문제와는 다르게 학습 과정에서의 클래스와 테스트 과정에서의 클래스가 다르다. 이러한 문제를 열림집합(Open-set) 문제라고 부르고, 얼굴 인식 분야에서는 해당 문제를 해결하기 위하여 다양한 손실 함수가 발전하게 되었다.[21] 얼굴 인식에 사용되는 손실함수는 소프트 맥스 손실함수, 거리 기반 손실함수, 앵귤러 마진(Angular Margine) 기반 손실함수, 소프트맥스 변형 손실함수[22]로 나눌수 있다. 거리 기반 손실함수는 대비(contrastive), 트리플렛(triplet) 손실함수를 들 수 있다. 최근 얼굴 인식은 소프트 맥스 기반 손실함수에 앵귤러 마진을 추가한 SphereFace[23], CosFace[24], ArcFace로 서로 다른 얼굴 간의 러리를 넓히는 방향으로 진행 되고 있다. 그림 4[22]는 얼굴인식에서 사용되는 손실함수의 발전 과정을 나타낸다. 붉은색은 소프트맥스 손실함수, 초록색은 거리 기반 손실함수, 파란색은 앵클러 마진 기반 손실함수, 노란색은 소프트 맥스 변형 손실함수를 나타낸다.

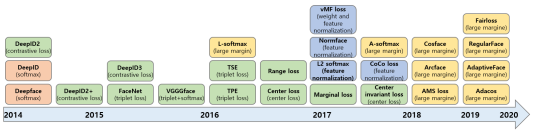

그림 4. 얼굴인식에서 사용되는 손실함수의 발전 과정
Fig. 4. Development flow of loss function used in face recognition

본 논문에서는 소프트맥스 변형 손실함수인 ArcFace를 사용한다. Additive Angular Margin 손실함수는 클래스 내 거리(Intra-class distance)를 줄이고, 클래스 간 특징 거리(Inter-class distance)를 늘린다. DCNN 특징 벡터와 마지막 완전 연결 층 사이의 내적은 정규화 후의 특징 벡터와 가중치(Weight)의 코사인 거리와 동일하다. 호-코사인 (Arc-cosine) 함수를 사용하여 현재 특징 벡터와 목표 가중치 사이의 각도를 계산하며, 목표 각도에 Additive Angular Margin을 추가하고, 코사인 함수를 이용하여 목표 로짓(Logit)을 다시 얻는다. 모든 로짓들을 고정된 특징 Norm에 따라 재 스케일하고, 후속 단계들은 소프트맥스 손실과 동일하게 동작한다. 또한 정규화된 초평면에서 호와 각도 사이의 정확성 덕분에 Geodesic Distance Margin을 최적화하며 이를 통해 좋은 성능을 낸다. 그림 5[7]는 Afcface 손실 함수의 학습을 나타낸다.

본 논문에서는 Resnet-50을 백본으로 사용하였다.

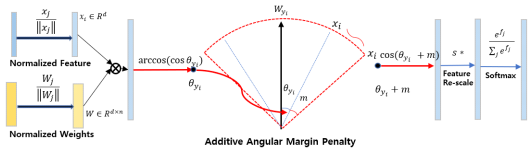

그림 5. ArcFace 손실에 의해 감독되는 얼굴 인식을 위한 DCNN 학습
Fig. 5. Training a DCNN for face recognition supervised by the ArcFace loss

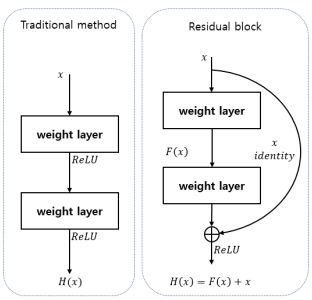

그림 6. Residual 블록
Fig. 6. Residual Block

ResNet은 마이크로소프트에서 개발한 알고리즘으로 핵심은 Residual Block의 출현으로 망이 깊어져도 문제가 되지 않도록 하였다. 전통적인 기존의 망과 차이가 있다면 입력값 x를 출력값에 더해주는 지름길(Shortcut)을 만들어줌으로써 네트워크 출력이 0이 되도록 매핑해서 최종 출력이 x가 되도록 학습한다. 그림 6은 Residual 블록을 나타낸다.

2.3 데이터 셋

얼굴 인식을 위한 데이터셋는 MegaFace, LFW[25], YTF, CelebFaces등 다양한 데이터 셋이 공개되고 있다. KISA인증을 위해선 한국인의 데이터 셋이 필요하다. 위의 데이터 셋은 서양인이 주를 이루는 데이터 셋이기 때문에 본 논문에서는 얼굴 인식 학습데이터 셋으로 한국인 안면이미지 데이터인 K-face[26]를 사용하였다. K-face는 성별, 연령대 분포를 고려하여 한국인을 대상으로 촬영한 안면 영상 데이터셋이다. 한국인 안면 이미지 데이터는 한국정보화진흥원(NIA) 지능정보산업 인프라 조성 사업의 세부 과제인 영상정보지식베이스 구축 사업의 일환으로 구축되었다. 한국인 안면이미지 데이터는 성별, 연령대 분포를 고려하여 한국인을 대상으로 촬영한 안면 영상 데이터이고, 2017년 200명, 2018년 200명, 2019년 600명의 인원에 대해 얼굴 포즈 20종, 액세서리 6종, 조명 30종, 표정 3종에 대해 촬영하였다.[24] 데이터는 고해상도(864x576), 중간해상도(346x230), 저해상도(173x115)

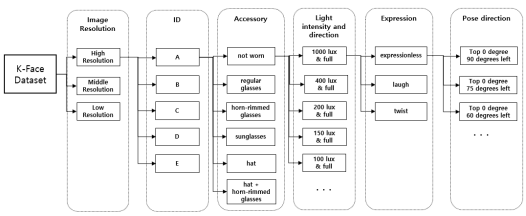

그림 7. K-Face 데이터셋의 구성도
Fig. 7. Configuration diagram of the K-Face dataset

그림 8. K-Face 얼굴 이미지
Fig. 8. K-Face face Images

로 구성된다. 파일은 "해상도/ID/액세서리 속성/조명 속성/표정 속성/포즈 속성.jpg"로 존재한다. 데이터베이스의 구조는 해상도, ID, 액세서리, 표정, 조명 위치 및 세기, 촬영 각도에 따라 다음 그림 7과 같이 구성되어 있다.

그림8은 K-Face의 데이터셋 얼굴을 나타낸다.

2.4 KISA 얼굴인식 인증

KISA 바이오인식정보시험센터는 국제 표준에 적합한 바이오인식시스템을 개발토록 유도하고 시스템의 정확성과 상호환성을 향상시켜 국산 바이오인식시스템의 수출 및 국내 바이오인식 산업진흥 촉진을 유도하고자[13]한다. 시험 인증 절차는 총 3단계로 나뉘며, 시험준비 단계, 시험 단계, 인증/종료 단계 순으로 이루어 진다. 본 논문에서는 얼굴 분야에 관한 인증 과정을 목표로 한다. 표 1은 KISA 바이오 얼굴인식 인증기준을 나타낸다. 표 2는 KISA 성능 시험 항목을 나타낸다. 표 3은 얼굴인식 성능시험용 DB 구성정보를 나타낸다.

실제 인증을 위해 다양한 데이터 수정 및 네트워크 확인 등을 통한 튜닝 작업이 필요하다. KISA 방문 후 DLL을 통한 알고리즘 성능 테스트 진행 후 얼굴 인식 인증 획득과정을 거치게 된다.

2.5 엔비디아 젯슨 임베디드 시스템

엔비디아의 임베디드 시스템인 젯슨 나노 모듈은 젯팩(JetPack)이 설치 되며, 젝팩은 최신 버전의 쿠다, cuDNN, TensorRT, 데스크톱 리눅스 운영체제가 함께 제공되어 AI 애플리케이션 구축을 빠르고 신속하게 시작할 수 있는 레퍼런스 플랫폼이다. micro SD 카드는 컴퓨터 운영 체제 및 저장 공간을 갖는다. 모니터 디스플레이를 위한 HDMI 포트, 유선 이더넷 연결을 위한 기가빗 이더넷 포트, 카메라 연결을 위한 CSI-2(Camera Serial interface) 카메라 커넥터, 외부 인터페이스 확장을 위한 40핀 확장 커넥터, 4개의 USB 3.0 포트, 5V 전원 DC 배럴 잭 등의 인터페이스를 갖는다. 표 4는 젯슨 나노의 기술 사양을 나타낸다.

표 1. 바이오 얼굴인식 인증기준
Table 1. Bio Face Recognition Certification Criteria

Category	EER			
	Light	Expression	Pose	Accessory
Face Recognition	1% or less	2% or less	2% or less	1.5% or less

표 2. KISA 성능 시험 항목
Table 2. KISA performance test items

Category	Details
FMR	others acceptance rate The ratio of when the biometric algorithm compares two images and determines that they are the same image, but is actually a different image
FNMR	self-rejection rate The ratio of when the biometric algorithm compares two images and identifies them as different images, but is actually the same image
EER	Error Rate when FMR and FNMR are the same

표 3. 얼굴인식 성능시험용 DB 구성정보
Table 3. DB configuration information for face recognition performance test

		1 set		2 set	
Number of people		500 people		500 people	
DB for registration		500 sheets		500 sheets	
DB for recogniti on		Category	DB number	Category	DB number
	1	lighting direction(8)	4,000 sheets (500x8)	lighting direction(8)	4,000 sheets (500x8)
	2	facial expression change(4)	2,000 sheets (500x8)	facial expression change(4)	2,000 sheets (500x8)
	3	pose (3)	1,500 sheets (500x8)	pose (3)	1,500 sheets (500x8)
	4	accessory (2)	1,000 sheets (500x8)	accessory (2)	1,000 sheets (500x8)
Total Images		9,000 sheets		9,000 sheets	
		18,000 sheets			
Shooting conditions		DB for registration : Frontal/Main white/Expressionless/Frontal lighting/Front pose lighting direction(8) : 0°, 45°, 90°, 135°, 180°, 225°, 270°, 315° expression(4) : Smiley, Angry, Eye closed, Surprised pose(3) : 22.5°(Left, Right), 45°(Left) accessory(2) : Hair band, Glasses			
Format		Image size(WxH) : 640x480 File Format : JPG file			

표 4. 젯슨 나노의 기술 사양
Table 4. Jetson Nano Technical Specifications

GPU	128-core NVIDIA MaxwellTM GPU
CPU	Quad-core ARM® A57 CPU
Memory	4 GB 64-bit LPDDR4
Storage	microSD (not included)
Video Encode	4x 1080p@30, 4K@30, 9x 720p@30 (H.264/H.265)
Video Decode	2x 4K@30, 4K@60, 8x 1080p@30, 18x 720p@30 (H.264/H.265)

Camera	2x MIPI CSI-2 DPHY lanes
Connectivity	Gigabit Ethernet, M.2 Key E
Display	HDMI 2.0 and eDP 1.4
USB	4x USB 3.0, USB 2.0 Micro-B
Others	GPIO, I2S, I2C, SPI, UART
Mechanical	100 mm x 80 mm x 29 mm

2.6 Tensor RT

TensorRT는 학습된 딥러닝 모델을 최적화하여 GPU 상에서의 추론 속도를 수배에서 수십배까지 향상시켜 딥러닝 성능을 개선하는데 도움을 줄 수 있는 모델 최적화 엔진이다. Caffe, Pytorch, TensorFlow 등의 딥러닝 프레임워크를 통해 학습된 딥러닝 모델을 TensorRT를 통해 학습된 모델을 최적화하여 JETSON TX2, TESLA T4, TESLA V100, JETSON NANO 등의 엔비디아 GPU 플랫폼에 싣는 것이다. 또한, TensorRT는 엔비디아 GPU 연산에 적합한 최적화 기법들을 이용하여 모델을 최적화해주는 Optimizer와 다양한 GPU에서 모델 연산을 수행하는 런타임 (Runtime) 엔진을 포함한다. TensorRT는 딥러닝 프레임워크에서 학습된 모델을 지원하고, 엔비디아 플랫폼에서 최적의 추론 성능을 낼 수 있도록, 네크워크 압축, 네트워크 최적화, GPU 최적화 기술들을 딥러닝 모델에 자동 적용해준다. 또한 최적의 딥러닝 모델 가속화를 지원한다. 딥러닝 가속화 방법으로는 양자화 및 정밀도 캘리브레이션, 그래프 최적화, 커널 자동 튜닝, 동적 텐서 메모리 및 멀티 스트림 실행등이 있다.[17] 그림 9는 엔비디아 TensorRT 구성을 나타낸다.

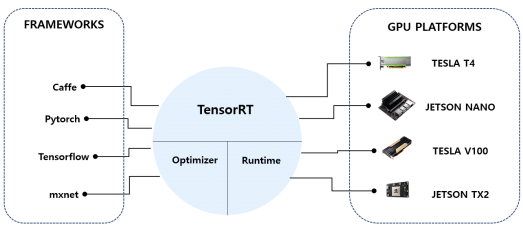

그림 9. NVIDIA TensorRT 구성
Fig. 9. NVIDIA TensorRT configuration

Ⅲ. 제안된 얼굴인식 시스템

3.1 시스템 구성

그림 10은 전체 시스템 구성도를 나타낸다. 그림1의 각 항목은 얼굴인식 학습 모델, KISA DLL, 젯슨 나노에서의 흐름도를 나타낸다.

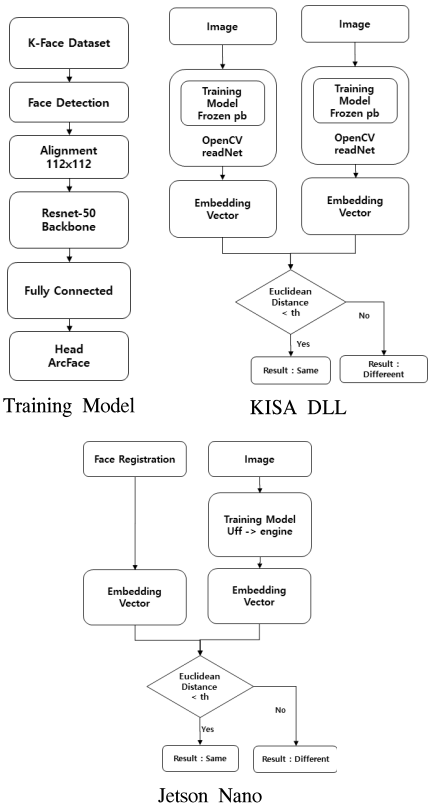

그림 10. 시스템 구성도
Fig. 10. Block diagram of system.

딥러닝의 학습은 파이썬을 이용하고, 딥러닝 프레임워크인 텐서플로우를 사용하였다. 실제 추론을 위해선 본 논문에서는 KISA 인증을 위한 방법과 젯슨 나노에서 동작하는 방법을 사용한다. KISA 인증을 위해선 Visual Studio C++를 사용하여 DLL로 파일을 만들어야 한다. 본 논문에서는 딥러닝으로 학습된 망의 사용이 가능하고, Visual Studio C++로 적용이 가능한 OpenCV를 적용하였다. 젯슨 나노 보드에서 수행하기 위해선 TensorRT로 동작 가능하도록 UFF(Universal Framework Format) 파일로 변경 후 엔진 파일을 만든후 사용하였다. 본 논문에서는 MTCNN 얼굴 검출 방법과 K-Face 데이터 셋을 사용하여 백본으로 Resnet-50과 손실 함수로 ArcFace를 사용하여 학습하였다. 학습된 방법을 KISA 인증 및 젯슨 나노 보드에서 TensorRT를 사용하여 실시간 동작하는 얼굴 인식 시스템을 구현하였다.

3.2 얼굴 검출

본 논문에서는 얼굴 검출 방법은 새롭게 학습하지

않고, 미리 학습된 망을 사용하였다. K-Face를 이용하여 얼굴 검출은 RetinaFace와 MTCNN 방법을 사용하여 테스트한 결과 MTCNN이 좀더 좋은 검출 성능을 나타내어 논 논문에서는 MTCNN 방법을 사용하였다.

3.3 얼굴 인식

본 논문에서는 구축된 학습데이터를 통한 얼굴 인식 백본으로 MobileNetV2[27]와 Resnet-50을 사용하여 학습을 진행하였다. 성능 결과가 더 좋은 Resnet-50을 백본으로 사용하고, 손실 함수로는 ArcFace, CosFace, 소프트맥스를 사용하여 학습을 진행 하였다. 학습 결과 가장 좋은 성능을 나타내는 ArcFace 방법을 사용하였다. 표 5는 얼굴 인식에 사용한 백본 및 손실함수 비교 결과를 타나 내고, %항목은 얼굴 인식의 성공률을 의미한다.

비교 데이터 셋은 LFW, AgeDB30(Age Database)[28], CFP-FP(Celebrities in Frontal Profile)[29], K-Face를 사용하였다. K-Face를 제외한 나머지 데이터 셋은 모두 서양인 얼굴이고, 화질 차이도 많이 나서 실제 성능에 많은 차이가 나는 것을 볼 수가 있다.

표 5. 백본 및 손실함수 비교 결과
Table 5. Comparison result of backbone and loss function

	Dataset	ArcFace	CosFace	Softmax
Resnet-50	LFW	60.38%	58.83%	61.33%
	AgeDB30	50,30%	50.58%	49.15%
	CFP-FP	53.67%	53.99%	54.50%
	K-Face	99.38%	99.10%	98.71%
MobileNetV2	LFW	53.40%	58.53%	66.10%
	AgeDB30	55.52%	50.00%	53.73%
	CFP-FP	51.00%	54.19%	60.36%
	K-Face	96.57%	96.21%	97.95%

3.4 KISA 인증

본 논문에서 언급하는 KISA 인증은 기업에서 얼굴 인식 시스템 개발할 경우 제품으로 개발시 객관적 성능 지표로 사용이 가능하다. KISA 인증을 위해 KISA 인증 문서에 따른 DLL 개발이 필요하다. DLL 개발을 위해 프로그램은 Visual Studio C++를 사용하였다. 실제 데이터의 학습은 PC상에서 Python을 사용하였고, 딥러닝 프레임워크는 텐서플로우를 사용하였다. 본 논문에서는 학습된 모델을 DLL 파일에서 사용하기 위해 C++에서 동작가능한 OpenCV를 사용하였다. OpenCV에서 학습된 텐서프로우 모델을 사용하기 위해선 pb 파일로 변환이 필요하다. pb 파일은 모델과 가중치가 모두 저장된 파일이다. 고정된 그래프(Frozen graph)란 그래프를 고정하여 pb 파일로 변경이되는 것이고, 재학습이 불가능 하다. 변환된 pb파일은 OpenCV의 readNet 함수를 이용하여 추론을 하였다. OpenCV의 readNet() 함수는 훈련된 가중치가 저장된 모델 파일과 네트워크 구조를 표현하는 설정 파일을 이용하여 Net 객체를 생성한다. readNet() 함수 원형은 "Net readNet(const String& model, const String& config = "", const String& framework = "");"이다. 추론을 할때는 forward 함수를 사용한다.

3.5 Tensor-RT 포팅

젯슨 나노 환경에서의 얼굴 인식으로 OpenCV, Tensorflow를 통한 실험을 하였으나 속도 저하로 인한 최적화가 필요하였다. 본 논문에서는 TensorRT를 사용한 성능 고도화 및 최적화를 수행하였다. TensorRT에서 모델을 사용하기 위해선 uff 또는 ONNX(Open Neural Network Exchange)[30]로 변환해야 한다. 본 논문에서는 convert-to-uff를 통해서 frozen pb 파일을 uff 파일로 변경해준다. 변환된 uff 파일을 TensorRT 엔진 파일로 변환 후 실행시 불러오면 된다. 엔진 파일로 변경을 위해선 tensorrt의 builder.build_cuda_engine(network) 함수를 사용한다. 엔진 파일을 로드하기 위해선 deserialize_cuda_engine() 함수를 사용한다.

Ⅳ. 얼굴인식시스템 구현

4.1 Face Recognition Dataset 구축

본 논문에서는 한국인으로만 구성된 K-Face의 데이터셋만을 사용하여 구현하였다. 실제 KISA 인증을 위해선 토픽이미지스 사에서 공급한 데이터셋과 자체 보유한 추가 데이터 셋을 추가하여 사용하였다. K-Face 데이터셋의 고화질 영상을 이용하여 학습 및 테스트를 진행하였다. 얼굴 검출은 MTCNN으로 진행 후 랜드마크를 이용한 정렬 과정을 거쳐 112x112 크기의 이미지로 구축하였다. 학습 이미지는 총 620,202장으로 하였고, 테스트를 위한 같은 이미지 쌍 5%, 다른 이미지 쌍 5%로 진행하였다. 얼굴의 클래스는 총 400개를 사용하였다. +-45°, +-30°, +-15°, 0 으로 총 7가지 방향 (빛의 방향 및 안경, 모자 착용 등 데이터 포함. 썬그라스, 모자 + 안경, 그밖에 몇가지 조명 조건 미포함)으로 데이터 셋을 구축하였다. 그림11은

K-Face Images　　Face Detection & Aligned Image

그림 11. K-Face 이미지와 얼굴 검출 & 정렬된 이미지
Fig. 11. K-Face Images and Face Detection & Aliened Image

K-Face 이미지 및 얼굴 검출 및 정렬이 수행된 이미지를 나타낸다.

4.2 KISA 인증

KISA 얼굴 인식 인증을 위해선 KISA에 시험신청서를 작성한다. 인증을 위한 최종 파일은 DLL 파일로 제공되어야하기 때문에 Visual Studio C++ 2015를 이용하여 구현하게 된다. Python으로 구현된 학습망을 DLL로 구현하는 방법으로 OpenCV를 이용하여 학습된 망을 읽어 올 수 있다. 얼굴 검출 및 얼굴 인식을 위한 학습은 PC에서 Python을 이용하고, 딥러닝 프레임워크인 텐서플로우를 사용한다. 학습된 모델은 pb 파일로 변환 후 OpenCV ReadNet() 함수를 이용하여 학습된 모델을 불러오고, forward() 함수를 사용하여 추론을 한다. KISA에서 인증 통과를 위해선 사전 시험에 통과하여야 한다. 사전 시험은 Visual Studio C++를 이용하여 DLL 파일을 만들어 KISA에 방문하여 테스트가 가능하다. 사전 성능 시험용 DB는 총 900장이며, 사전 성능 시험에 통과하면 본시험을 통해 인증 시험을 보게 된다. 사전 시험 통과후 KISA에 인증용 DLL을 제출하면 KISA에서 보유한 테스트 데이터 셋을 통하여 시험후 인증 기준을 통과하면 KISA 인증서를 받게된다.

4.3 젯슨 나노에서 실행

젯슨 나노에서 얼굴 인식을 위해 영상은 CSI, RTSP(Real Time Streaming Protocol), 비디오 파일 등 총 3가지 방식으로 받을 수 있도록 하였다. 각각의 영상은 메뉴 설정을 통하여 선택 가능하다. 그림12는 실행 환경과 젯슨 나노 모듈 및 카메라 모듈을 나타낸다. 얼굴 등록과 동작 결과를 확인하기 위해 자체로 Qt5를 이용하여 GUI 프로그램을 개발하였다.

그림 13은 GUI의 실행 화면을 나타낸다. 그림 13의 실행 화면에서 얼굴의 등록은 화면상에서 검출된

Execution environment　Jetson Nano Board　Camera Module

그림 12. 실행 환경, 젯슨 나노 보드, 카메라 모듈
Fig. 12. Execution environment, Jetson Nano Board, Camera Module

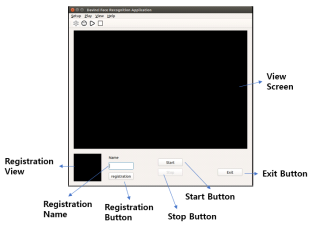

그림 13. GUI 실행 화면
Fig. 13. GUI Execution Screen

얼굴을 클릭하면 아래에 등록 화면(Registration View)에 표시가 되고, 등록 이름(Registration Name)을 수정한 후 등록 버튼(Registration Button)을 통하여 등록이 가능하다.

그림 14는 영상 입력 및 다양한 설정이 가능한 스트림 설정을 나타낸다.

표 6은 Stream 설정의 각 항목별 내용을 나타낸다.

그림 15는 얼굴 등록을 위한 Face 설정을 나타낸다. Face 설정은 파일을 통한 등록 및 등록된 얼굴의 이름 수정, 등록된 얼굴 삭제등의 기능이 있다.

표 7은 얼굴 설정의 각 항목별 내용을 나타낸다.

젯슨 나노 환경에서의 실시간 처리를 위해선 TensorRT를 사용하여야 한다. 표 8은 MTCNN 방법을 사용할 경우 TensorRT 사용에 따른 성능 비교를 나타낸다.

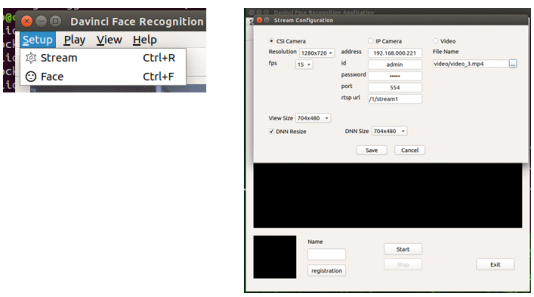

그림 14. Stream 설정
Fig. 14. Stream Setting

표 6. Stream 설정의 각 항목별 내용
Table 6. Contents of each item of Stream setting

CSI Camera	CSI(Camera Serial Interface) Camera
IP Camera	RTSP Camera
Video	Video File
Resolution	CSI Camera Resolution
fps	1~30
address	Network Camera IP address
id	Network Camera Id
password	Network Camera Password
port	RTSP Port Number
rtsp url	rtsp:://address(rtsp url)
File Name	Video File Path
View Size	320x240, 640x360, 704x480, 960x540, 1280x720, 1600x900
DNN Resize	Check Deep Neural Network Resize
DNN Size	Deep Neural Network Resize

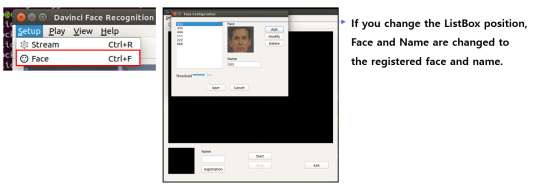

그림 15. 얼굴 설정
Fig. 15. Face Setting.

표 7. Face 설정의 각 항목별 내용
Table 7. Contents of each item of Face setting

Add	Detecting a face in the file and then registering the detected face
Modify	Modified the name of a registered face
Delete	Delete registered face
Threshold	Setting the threshold of distance when discriminating the same face

표 8 TensorRT 사용에 따른 MTCNN 성능 비교
Table 8. Performance MTCNN comparison using TensorRT

Algorithm	Resolution	OpenCv 4.1.0 Time (msec)	TensorRT Time (msec)
MTCNN	640x360	340 ~ 360	60 ~ 70
	1280x720	1,30 ~ 1,400	117 ~ 120

얼굴인식 방법인 ArcFace의 경우 TensorRT로 적용하여 실행할 경우 추론 시간이 30 msec 정도로 빠르게 실행되었다. 젯슨 나노 보드에서 TensorFlow를 실행하면 나도보드의 메모리 부족으로 메모리 오류가 나는 문제로 젯슨 자비어 (Jetson Xavier)를 사용하여 테스트하였다. 표 9는 TensorRT 사용에 따른 ArcFace 성능 비교를 나타낸다.

표 9. TensorRT 사용에 따른 ArcFace 성능 비교
Table 9. Performance ArcFace comparison using TensorRT

Algorithm	TensorFlow Cuda 10.2 Time (msec)	TensorRT Time (msec)
ArcFace	90 ~ 100	9 ~ 20

TensorFlow Cuda 10.2는 90 ~ 100 msec, TensorRT는 9 ~ 20 msec로 TensorRT를 사용함으로써 속도가 빨라지는 것을 확인 하였다.

V. 결 론

본 논문에서는 한국인 얼굴로 구성된 K-Face 데이터 셋을 이용하여 학습하였다. 학습을 위한 백본으로 Resnet-50을 사용하였고, 소프트맥스 변형 손실함수인 ArcFace를 사용하여 얼굴 인식을 학습하였다. 얼굴 인식을 위해선 MTCNN 방법으로 얼굴 검출 과정을 거쳐 얼굴을 검출하고, MTCNN을 통해 검출된 랜드마크를 이용하여 얼굴 정렬 후 얼굴 인식 과정을 거친다. KISA 얼굴 인식 인증을 위해 딥러닝 프레임 워크로 학습된 망을 OpenCV를 사용하여 추론하였으며, 인증을 위해서 KISA에서 요구하는 DLL로 변경하여 인증을 진행하였다. 임베디스 시스템인 젯슨 나노를 통해 실시간 얼굴 인식을 동작하기 위해선 TensorRT를 사용하여 얼굴 인식을 구현하였다. TensorRT의 사용여부에 따라 속도를 비교하였으며, 동작 테스트를 위해 자체 GUI 프로그램을 개발하였다. 이슈 항목으로는 한국인 얼굴만을 사용하였기 때문에 오히려 서양인 얼굴 인식 성능이 떨어지는 문제가 있었다. 향후 다양한 얼굴 표정이나 악세사리 적용 및 밝기 변화, 인종에 따른 얼굴인식 정확도를 높이기 위해 한국인 데이터셋뿐 아니라 다양한 데이터셋의 적용과 개선된 방법에 대한 연구를 지속 할 예정이다.

References

[1] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. 1991 IEEE Comput. Soc. Conf. CVPR*, pp. 586-591, 1991.

[2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. 2001 IEEE Comput. Soc. Conf. CVPR*, 2001.

[3] P. N. Belhumeur, J. P. Hespanha, and D. J.

Kriegman, “Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection,” in *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 19, no. 7, pp. 711-720, Jul. 1997.

[4] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “DeepFace: Closing the gap to human-level performance in face verification,” *2014 IEEE Conf. CVPR*, pp. 1701-1708, 2014.

[5] M. Omkar, et al., “Deep face recognition,” in *Proc. BMVC*, pp. 41.1-41.12. BMVA Press, Sep. 2015.

[6] Y. Sun, X. Wang, and X. Tang, “Deep learning face representation from predicting 10,000 classes,” *2014 IEEE Conf. Comput. Vision and Pattern Recog.*, pp. 1891-1898, 2014.

[7] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” in *Proc. IEEE Conf. CVPR*, pp. 4690-4699, 2019.

[8] J. Redmon and A. Farhadi, “YOLOv3: An incremental improvement,” arXiv:1804.02767, Apr. 2018.

[9] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal speed and accuracy of object detection,” arXiv:2004.10934, Apr. 2020.

[10] H. Gan and C. Qing, “Research on face recognition algorithm based on MT-CNN,” *Ind. Contr. Comput. J.*, vol. 31, no. 11, pp. 119-122, May 2018.

[11] *OpenCV Library*, https://opencv.org/

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” arXiv:1512.03385, 2015.

[13] *Korea Internet & Security Agency*, https://www.kisa.or.kr

[14] *Nvidia Corporation*, https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[15] YoungJoon Hwang, Jinha Song, Jongho Nang, “Design for Smart CCTV Platform Architecture on Jetson nano with FFMPEG and TensorRT,“ *Proceedings of the Korean Information Science Society Conference,* pp. 1664-1666, 2020.

[16] Sang-hun Chun, Ji-hoon Choi, Ye-jin Kim, Sung-kwan Kang, “Smart Door Implementation Using Jetson Nano-Based OpenCV and Deep Learning,” *The Journal of Korean Institute of Communications and Information Sciences,* 46(2), pp. 380-387, Feb 2021.

[17] *Nvidia Corporation*, https://developer.nvidia.com/tensorrt

[18] S. Zhang, et al., “S3FD: Single shot scale-invariant face detector,” in *2017 IEEE ICCV*, pp. 192-201, Venice, Italy, 2017.

[19] J. Deng, et al., “RetinaFace: Single-shot multi-level face localisation in the wild,” in *Proc. IEEE/CVF Conf. CVPR*, pp. 5203-5212, 2020.

[20] T.-Y. Lin, et al., “Feature pyramid networks for object detection,” in *Proc. IEEE Conf. CVPR*, pp. 2117-2125, 2017.

[21] A. Krizhevsky, et al., “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, pp. 84-90, Jun. 2017.

[22] M. Wang and W. Deng, “Deep face recognition: A survey,” arXiv:1804.06655v9, 2020.

[23] W. Liu, et al., “SphereFace: Deep hypersphere embedding for face recognition,” in *Proc. IEEE Conf. CVPR*, pp. 212-220, 2017.

[24] H. Wang, et al., “CosFace: Large margin cosine loss for deep face recognition,” in *Proc. IEEE Conf. CVPR*, pp. 5265-5274, 2018.

[25] G. B. Huang, et al., “Labeled faces in the wild: A database forstudying face recognition in unconstrained environments,” *Wkshp. faces in ‘Real-Life’ Images: Detection, alignment, and recognition*, 2008.

[26] *AI Hub*, kface.aihub.or.kr

[27] M. Sandler, et al., “MobileNetV2: Inverted residuals and linear bottlenecks,” in *Proc. IEEE Conf. CVPR*, pp. 4510-4520, 2018.

[28] S. Moschoglou, et al., “Agedb: The first manually collected, in-the-wild age database,” in *Proc. IEEE Conf. CVPRW*, pp. 51-59, 2017.

[39] S. Sengupta, et al., “Frontal to profile face verification in the wild,” *IEEE WACV*, pp. 1-9, 2016.

[30] https://onnx.ai/

장 일 식 (Il-Sik Chang)

2011년 2월 : 서울과학기술대학교 NID융합기술대학원 석사
2020년 3월~현재 : 서울과학기술대학교 지능형미디어연구센터 책임연구원
2020년 9월~현재 : 서울과학기술대학교 나노IT디자인융합대학원 정보통신미디어공학전공 박사과정
<관심분야> 전자공학, 통신공학, 광통신 공학
[ORCID:0000-0003-0822-9857]

박 구 만 (Goo-Man Park)

1984년 2월 : 한국항공대학교 전자공학과 공학사
1986년 2월 : 연세대학교 전자공학과 공학석사
1991년 2월 : 연세대학교 전자공학과 공학박사
1991년 3월~1996년 9월 : 삼성전자 신호처리연구소 선임연구원
2016년 1월~2017년 12월 : 서울과학기술대학교 나노IT디자인융합대학원 원장
1999년 8월~현재 : 서울과학기술대학교 전자IT미디어공학과 교수
2006년 1월~2007년 8월 : Georgia Institute of Technology Dept.of Electrical and Computer Engineering, Visiting Scholar
<관심분야> 컴퓨터비전, 지능형실감미디어
[ORCID:0000-0002-7055-5568]