JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 21, No. 3, pp. 471-478, Mar. 2020

Check for updates

딥러닝 기반 가상공간에서의 손 제스쳐 인식

임 영 재[1] · 정 일 홍[2*]
[1]대전대학교 컴퓨터공학과 석사과정
[2]대전대학교 컴퓨터공학과 교수

Hand Gesture Recognition in the Virtual Space based on Deep Learning

Young-Jae Lim[1] · Il-Hong Jung[2*]
[1]Master Course, Department of Computer Engineering, Daejeon University, Daejeon 34520, Korea
[2]Professor, Department of Computer Engineering, Daejeon University, Daejeon 34520, Korea

[요 약]

본 논문에서는 기존의 가상/증강현실 인터페이스 장치의 가격, 인식 속도 등의 문제점들을 개선하고자 가상공간에서 사용자 인터페이스로 사용될 Static Gesture와 Dynamic Gesture를 정의하고 RGB 카메라를 통하여 입력 받은 손 제스처를 딥러닝 모델들을 이용하여 특징을 추출하고 인식하는 방법을 제안한다. 여러 가지 딥러닝 모델을 통하여 다양한 방법으로 데이터를 학습시키고 특징을 추출하여 손 제스처를 인식해 보았다. 사용한 딥러닝 모델은 Faster-RCNN, ResNet, U-Net, 3D-CNN 모델이다. 가상공간에서 손 제스처를 인식 하여 사용자 인터페이스로 사용하므로 특정한 센서나 웨어러블 기기의 도움 없이 높은 인식률과 빠른 인식 속도를 통하여 가상/증강 현실을 사용하는데 이바지 하고자 한다,

[Abstract]

In this paper, we define static gestures and dynamic gestures to be used as a user interface in a virtual space, and propose a method to extract features using deep learning models and to recognize hand gestures input through RGB camera in order to improve the price and recognition speed of the existing virtual / augmented reality interface device. Through various deep learning models, we learned the data in various ways and extracted the features to recognize hand gestures. Deep learning models used are Faster-RCNN, ResNet, U-Net, and 3D-CNN. Since we recognize hand gestures in the virtual space and use them as user interfaces, we want to contribute to using virtual / augmented reality through high recognition rates and fast recognition speeds without the help of specific sensors or wearable devices.

색인어 : 딥러닝, 가상 공간, CNN , 손 제스쳐 인식, 사용자 인터페이스

Key word : Deep Learning, Virtual Space, CNN , Hand Gesture Recognition, User Interface

http://dx.doi.org/10.9728/dcs.2020.21.3.471

Received 17 January 2020; **Revised** 15 March 2020
Accepted 25 March 2020

***Corresponding Author; Il-Hong Jung**

Tel: +82-42-280-2548
E-mail: ijung@dju.kr

Ⅰ. 서 론

웨어러블 기기의 발전으로 함께 주목받기 시작한 증강현실은 마이크로소프트에서 발표한 홀로렌즈의 개발 계획과 함께 본격적인 개발궤도에 오르고 있고, SF장르의 단골 소재로 등장하던 가상현실은 2014년 3월 페이스 북에 2조 5천억 원에 인수된 오큘러스 VR의 오큘러스 리프트를 통해 우리에게 점차 현실의 이야기가 되고 있다[1].

하지만 디스플레이 장치의 발전 속도에 비해 인터페이스 기술의 개발은 상대적으로 느린 게 현실이다. 키보드, 마우스, 터치스크린 등의 기존 인터페이스와 달리 웨어러블 기기 같은 장치를 착용하면 사용자가 자유롭게 움직이지 못하기 때문에 가상 세계에서도 편리하게 이용 가능해야 하는 가상/증강현실 인터페이스의 요구사항을 만족시키지 못하고 있다. 뿐만 아니라 가상/증강현실 인터페이스 장치 같은 경우 비싼 가격과 인식 속도가 빠르지 않다[1].

이로 인하여 본 논문에서는 기존의 가상/증강현실 인터페이스 장치의 가격, 인식속도의 문제점들을 좀 더 개선하고 사용자에게 편리함을 더 높여주기 위하여 딥러닝을 활용하여 가상공간에서 손 제스처를 인식 하여 사용자 인터페이스로 사용하고자 한다. 가상공간에서 특정한 센서가 필요하지 않고 비싼 가격을 주고 웨어러블 기계를 구매할 필요도 없을 것이며 높은 인식률과 빠른 인식 속도를 통하여 가상/증강 현실을 사용하는데 이바지 하고자 한다[2].

Ⅱ. CNN 모델

CNN(Convolutional Neural Network)는 모델이 직접 이미지, 비디오, 텍스트 또는 사운드를 분류하는 머신 러닝의 한 유형인 딥러닝에 가장 많이 사용되는 알고리즘이다. CNN은 이미지에서 객체, 얼굴, 장면을 인식하기 위해 패턴을 찾는데 특히 유용하다. CNN은 데이터를 직접 찾고 특징을 분류하는데 직접 학습하기 때문에 수동 작업이 필요하지 않고 높은 수준의 인식 결과를 나타내기 때문에 자율 주행 자동차, 얼굴 인식 애플리케이션과 같이 객체 인식과 컴퓨터 비전이 필요한 분야에서 CNN을 많이 사용한다. 응용 분야에 따라 CNN을 처음부터 만들 수도 있고, 데이터 셋으로 사전 학습된 모델을 사용할 수도 있다[3]. 본 논문에서 사용한 Res Net, U-Net, Faster R-CNN과 3D-CNN에 대해 살펴본다.

2-1 ResNet

ResNet은 마이크로소프트에서 개발한 알고리즘이며, 2015년 ILSVRC에서 오류율 3.6%로 1등을 차지했다. AlexNet이 처음 제안된 이후로 CNN 아키텍처의 층은 점점 더 깊어졌다. AlexNet이 불과 5개 층에 불과한 반면 VGGNet은 19개 층, GoogleNet은 22개 층에 달한다. 하지만 ResNet은 152개 층이다. ResNet 핵심은 Residual Block인데 그레디언트가 잘 흐를 수 있도록 일종의 지름길을 만들어 주는 역할을 한다. ResNet의 성능이 좋은 이유는 Residual Block이 앙상블 모델을 구축한 것과 비슷한 효과를 내기 때문이다. 그림 1은 ResNet50의 구조를 간략히 표현한 것이다[4].

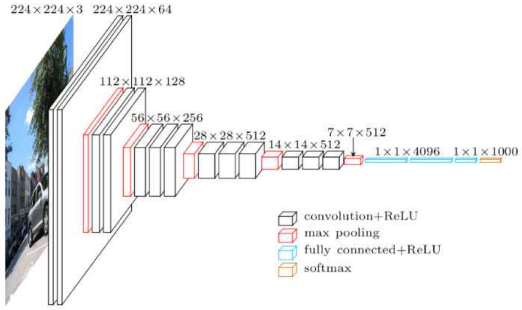


그림 1. ResNet 구조
Fig. 1. ResNet Structure

2-2 U-Net

U-Net은 그림 2처럼 U자 모양처럼 생겼다. U Net은 단순히 이미지를 Classification하는 문제를 넘어서 이미지의 특정 영역을 Label로 표현하는 Image Segmentation하는 것에 주된 목적이 있는 모델이다.

기존의 모델들에 비해서 개선된 점은 크게 두 가지다. 첫째는 Sliding Window보다 빠르게 Patch 방식을 채택하여 이미지를 전부 조금씩 잘라서 훑어보는 것이 아니라, 이미지 전체를 격자 모양으로 잘라서 한 번에 인식하기 때문에 속도가 빠르다. 두 번째는 Patch의 사이즈에 따른 Trade off에 빠지지 않는다. 기존의 방식은 한 번에 넓은 구역을 보면, 전체적인 그림의 인식 확률은 좋아지지만 Localization이 부족해지고, 반대로 좁은 구역을 보면 더 세분화된 Localization이 가능하지만 인식률이 떨어지는 단점이 있다[5].

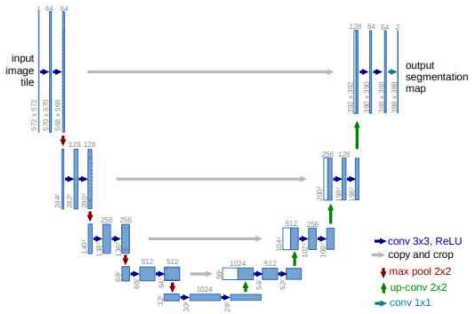


그림 2. U-Net 구조
Fig. 2. U-Net Structure

2-3 Faster R-CNN

Fast R-CNN은 기존의 R-CNN을 개선시킨 모델로 기존의 R-CNN에서는 각 Region Proposal마다 CNN을 거쳐야 했는데 Fast R-CNN에서는 전체 이미지와 Object Proposal들을 한번에 CNN을 거치게 했다는 것이다. Faster R-CNN을 사용하면 이미지 속의 여러 사물을 한꺼번에 분류해 내놓으며, 데이터 학습에 따라서 겹쳐 있는 부분들 까지도 정확하게 사물들을 분류해낼 수 있다[6].

그림 3을 보면 Faster R-CNN 알고리즘은 각 윈도우에 있는 Feature Map을 검색하고, 고정 크기로 조정한 뒤 클래스 확률과 해당 객체에 대한 더욱 정확한 경계박스를 예측하는 방식이다[7].

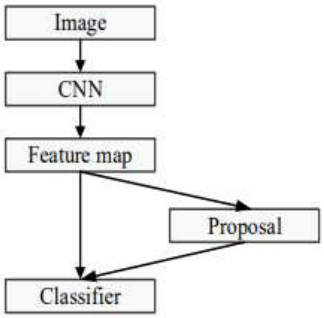


그림 3. Faster R-CNN 구조
Fig. 3. Faster R-CNN Structure

2-4 3D-CNN

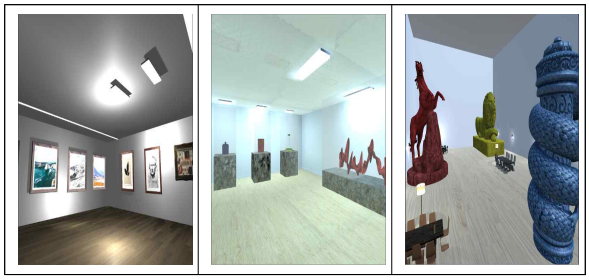
3차원 CNN은 2차원 CNN을 시간 축으로 한 차원 확장시킨 형태의 인공 신경망이다. 2차원 CNN은 일반적으로 이미지를 입력받아서 그 공간적인 특성을 찾아내어 이미지를 분류하거나, 그와 관련된 응용 분야에 뛰어난 성능을 보이지만, 시간 정보는 다룰 수가 없기 때문에 동영상 데이터는 처리 할 수 없다는 한계가 있다. 반면 3차원 CNN은 컨볼루션 연산과 풀링 연산 등을 시간 성분까지 함께 계산하도록 함으로써, 영상 데이터의 특징을 뽑아 낼 수 있다는 장점이 있다[8][9][10].

본 논문에서는 3D-CNN으로 3D-UNet과 3D- ResNet을 사용하였다.

Ⅲ. 딥러닝을 이용한 가상공간에서 손 제스처 인식

3-1 가상 공간

가상공간은 총 3개의 공간으로 구성하였으며 그림 4와 같이 그림 전시관, 유물 전시관, 조형물 전시관으로 구성하였다.



Picture Exhibition Relic Exhibition Sculpture Exhibition

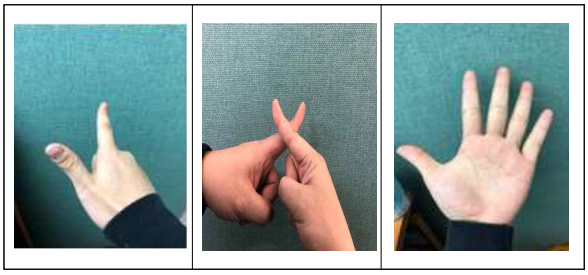
그림 4. 가상공간 – 전시관
Fig. 4. Virtual Space - Exhibition

3-2 손 제스처 정의

가상공간에서 사용자 인터페이스로 사용될 손 제스처는 3가지 Static Gesture와 6가지 Dynamic Gesture로 정의한다.

1) Static Gesture

Static Gesture는 움직이지 않는 손 모양의 제스처로 정의한다. Static Gesture에는 그림 5와 같이 Point Gesture, Stop Gesture, Restore Gesture 총 3가지가 있다.



Point Gesture Stop Gesture Restore Gesture

그림 5. 정적 제스처
Fig. 5. Static Gesture

Point Gesture는 가상환경에서 특정 물체를 움직이지 않고 제자리에서 좀 더 자세히 보고 싶은 경우 사용하는 제스처이다. Point Gesture는 검지손가락이 가리키는 부분에 물체가 있을 경우 그 물체를 현재 보고 있는 시야에서 3m 앞으로 이동해 오게 한다.

두 번째 Stop Gesture는 앞에서 Point Gesture로 가리킨 물체를 더 이상 보고 싶지 않을 경우 사용하는 제스처이다.

마지막으로 Restore Gesture는 뒤에서 나오는 Dynamic Gesture 중 Enlargement Gesture, Reduction Gesture, Rotation Gesture를 통해 물체가 변화하였을 경우 다시 원상복구 시키고 싶을 때 사용하는 제스처 이다.

2) Dynamic Gesture

Dynamic Gesture는 움직이는 손 모양의 제스처로 정의

한다. Dynamic Gesture는 그림 6과 같이 Next Gesture, Previous Gesture, Retreat Gesture, Enlargement Gesture, Reduction Gesture, Rotation Gesture 총 6가지로 구성되어 있다.

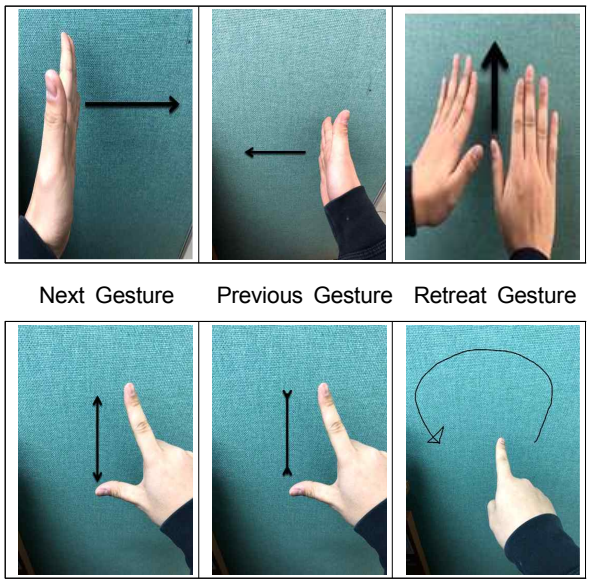


그림 6. 동적 제스처
Fig. 6. Dynamic Gesture

먼저 Next Gesture는 현재 내가 위치하고 있는 가상공간에서 다음 가상공간으로 이동하고 싶을 때 사용하는 제스처이다.

두 번째, Previous Gesture는 현재 내가 위치하고 있는 가상공간에서 이전 가상공간으로 이동하고 싶을 때 사용하는 제스처이다.

세 번째, Retreat Gesture는 Point Gesture로 가리킨 물품을 50cm 정도 뒤로 이동시키고 싶을 때 사용하는 제스처이다.

네 번째, Enlargement Gesture는 Point Gesture로 가리킨 물품을 좀 더 확대 하고 싶을 때 사용하는 제스처이다.

다섯 번째, Reduction Gesture는 Enlargement Gesture를 통해 확대한 물체를 축소하고 싶을 때 사용하는 제스처이다.

마지막으로 Rotation Gesture는 Point Gesture로 가리킨 물품을 회전시키고 싶을 때 사용하는 제스처이다.

3-3 Data Augmentation

20명의 사람들로부터 9개의 손 제스처를 다른 배경에서 40장씩 찍어 수집한 데이터는 7,200장이다. 이 데이터를 Augmentation 기법을 활용하여 45,600장으로 늘렸다. 각 손 제스처 데이터를 랜덤으로 추출하여 Augmentation을 하였다.

본 논문에서 사용한 Augmentation방법은 Rotation, Shifting, Rescaling, Flipping, Shearing, Stretching 방법을 사용하였다. 먼저 Rotation은 데이터를 0~360도 사이로 임의로 회전시키는 방법이다. 두 번째 Shifting은 임의로 데이터를 10픽셀씩 상/하/좌/우 로 움직여주게 된다. 세 번째 Rescaling을 통해 데이터를 1.0~1.6배로 사진을 확대한다. 네 번째 Flipping은 데이터를 상하/좌우 반전을 시킨다. 다섯 번째, Shearing은 강제로 데이터를 찌그러뜨리는 방법이다. 여섯 번째, Stretching은 강제로 데이터를 1.0배~1.3배로 늘어뜨리는 방법이다.

3-4 학습 과정

손 제스처를 인식하기 위해 본 논문에서 활용한 학습과정은 ConvNet 학습과정, Object Detection 학습과정과 Segmentation 학습과정 총 3가지 학습과정이다.

1) ConvNet 학습과정

그림 7과 같이 ConvNet 학습과정은 먼저 수집한 데이터 셋을 Data Augmentation을 통해 45,600장으로 늘린 데이터를 이용하여 CNN 모델로 학습시킨다. 전체 데이터에 대한 한 번의 학습이 끝나면 평가 데이터 세트를 통해 인식률을 측정한다. 본 논문에서 ConvNet 학습과정을 사용한 CNN 모델은 ResNet과 3D-ResNet이 있다.

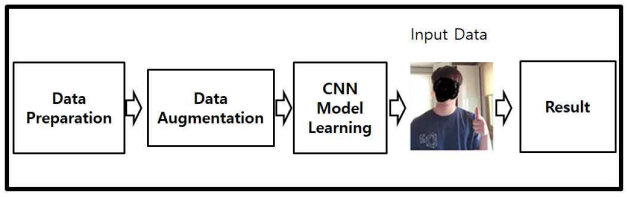


그림 7. ConvNet 학습 과정
Fig. 7. ConvNet Learning Process

2) Object Detection 학습과정

그림 8과 같이 Object Detection 모델의 학습과정은 먼저 수집한 데이터 셋을 Data Augmentation을 통해 45,600장으로 늘리고 Object Detection 모델을 통하여 데이터의 손 부분을 추적한다. Object Detection 시킨 후 손 부분만 추출하여 학습시킨다. 전체 데이터에 대한 한 번의 학습이 끝나면 평가 데이터 셋을 통해 인식률을 측정한다. 본 논문에서 Object Detection 학습과정을 사용한 CNN모델은 Faster R-CNN이 있다.

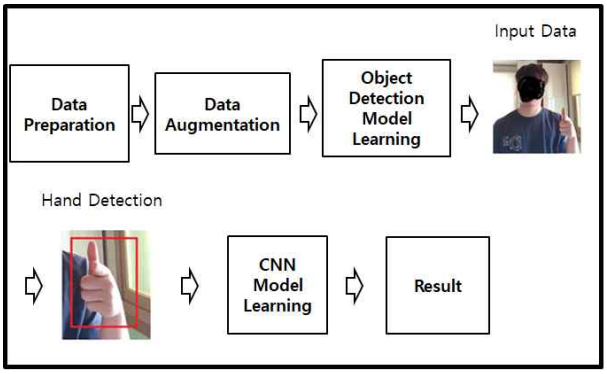


그림 8. Object Detection 학습 과정
Fig. 8. Object Detection Learning process

3) Segmentation 학습과정

그림 9와 같이 Segmentation 모델의 학습과정은 먼저 수집한 데이터 셋을 Segmentation 모델을 통하여 데이터를 Segmentation시키고 Segmentation시킨 데이터를 Augmentation 방법을 사용하여 늘린 데이터를 활용하여 학습시킨다. 전체 데이터에 대한 한 번의 학습이 끝나면 평가 데이터 셋을 통해 인식률을 측정한다. 본 논문에서 Segmentation 학습과정을 사용한 CNN모델로는 U-Net과 3D-UNet이 있다.

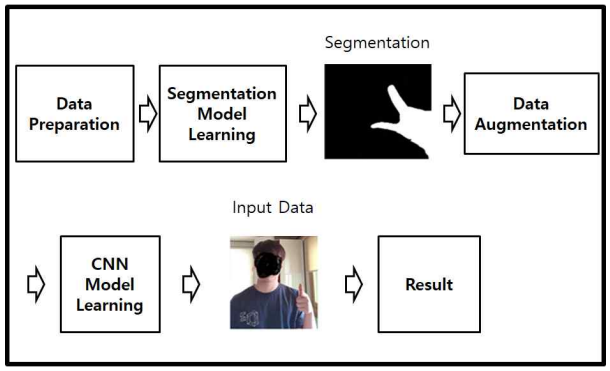


그림 9. Segmentation 학습 과정
Fig. 9. Segmentation Learning process

3-5 손 제스처 특징 추출 및 인식

ConvNet을 활용하여 그림 10과 같이 4가지의 Feature Map을 생성해 보았다. 먼저 ConvNet에서 손 제스처 입력 데이터를 받고 입력 받은 손 제스처 데이터를 컨볼루션 연산을 통해 컨볼루션 필터를 생성한다. 생성된 컨볼루션 필터와 활성화 함수를 연산하여 입력 데이터에 대한 Feature Map을 생성하였다. 생성된 Feature Map은 Sharpen, Edge Detection, Gaussian blur, Box blur 총 4개이다. 생성된 Feature Map을 통하여 특징을 추출하게 된다. 추출된 특징들을 학습을 통해 각각의 손 제스처가 어떤 제스처인지 인식한다[11].

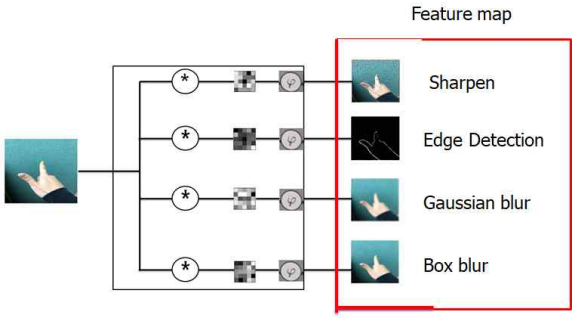


그림 10. Feature map 추출 방법
Fig. 10. Feature map extraction method

Object Detection 알고리즘을 통한 특징 추출 과정은 먼저 각각의 제스처에 대하여 학습시킨 후 입력 영상에서 손 부분만 따로 인식 한다. 인식 된 손 부분에 Bounding Box를 그리고 Convolution Layer에서 특징들을 추출하여 학습한다. 학습 후에는 Bounding Box로 그려진 손에 대하여 어떤 제스처인지 Classification하여 인식한다[12].

그림 11은 Stop Gesture의 특징을 추출하고 인식하는 예시 그림이다. Bounding Box를 통해 손의 경계선을 그려서 손 부분에 집중할 수 있게 된다. 그 다음 Convolution Layer에서 특징들을 추출하여 Classification을 통해 어떤 손 제스처인지 인식한다.

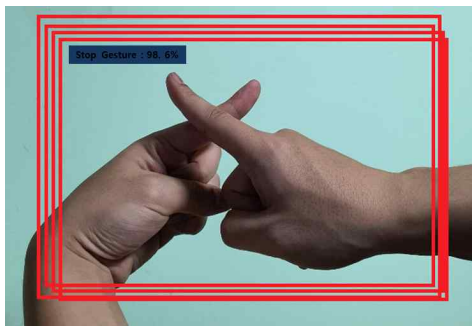


그림 11. Object Detection 방법에 의한 Stop Gesture 인식
Fig. 11. Stop Gesture Recognition using Object Detection

Segmentation 알고리즘을 통한 특징 추출 과정은 Segmentation을 통하여 손을 제외한 이미지의 모든 것을 제거한다. 이미지에서 배경을 빼면 이진 임계값을 사용하여 대상 제스처를 완전히 흰색으로 만들고 배경을 완전히 검정색으로 만드는 Segmentation 기법을 활용하였다. Segmentation을 통해 새로운 이미지를 생성하고 새롭게 생성 된 이미지에서 특징들을 추출하여 어떤 제스처인지 인식하게 된다[13].

그림 12는 손 제스처를 Segmentation 한 예시 그림이다.

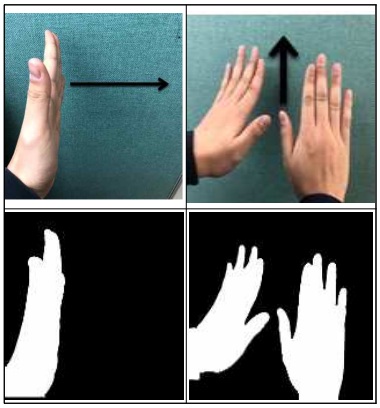


그림 12. 손 제스처 Segmentation 예시
Fig. 12. Examples of Hand Gesture Segmentation

3-6 가상공간에서 손 제스처 인식 구현

그림 13은 가상공간에서 Point Gesture를 인식한 실행화면이다. Point Gesture를 인식하면 검지손가락의 좌표를 받아와 벡터 값으로 계산한다. 벡터 값 계산 방법으로는 손 끝 검출방법을 사용하였다. 이렇게 계산된 벡터 값에 어떠한 물품이 부딪히게 되면 그 물품이 현재 보고 있는 시야에서 3m 앞으로 다가오게 된다. 만약 어떠한 물품에도 부딪히지 않으면 이 제스처는 작동하지 않는다.

그림 13. 가상공간에서 Point Gesture를 인식한 실행 화면
Fig. 13. Execution Screen that Recognizes Point Gesture in Virtual Space

그림 14는 가상공간에서 Enlargement Gesture를 인식한 실행화면이다. Enlargement Gesture를 인식을 한 경우 Point Gesture로 가리킨 그림이나 물품, 조형물을 10% 확대시켜준다. 좀 더 세밀하게 관찰하기 위해 만든 제스처이다.

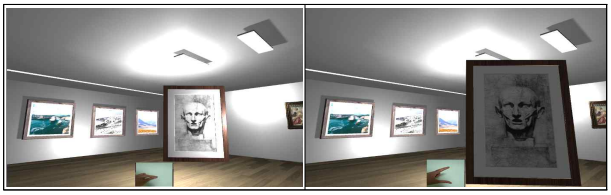


그림 14. 가상공간에서 Enlargement Gesture를 인식한 실행 화면
Fig. 14. Execution Screen that Recognizes Enlargement Gesture in Virtual Space

Ⅳ. 성능평가 및 분석

4-1 데이터 학습의 정확도에 대한 성능평가 및 분석

Static Gesture는 공간적인 특성만 고려하는 제스처이다. 공간적인 특성만 고려하다보니 2D CNN모델과 3D CNN모델 두 가지 모두 사용 할 수 있었다.

그림 15는 Static Gesture에 대한 모델 별 데이터 학습의 정확도를 나타내고 있다. 3D-UNet은 99%, 3D-ResNet은 97%, Faster R-CNN 92%, U-Net 94%, ResNet 88%의 결과가 도출되었다. Static Gesture의 모델 별 결과를 보았을 때 Segmentation한 모델인 3D-UNet과 U-Net이 다양한 조명 조건, 복잡한 배경 및 피부색에 대한 제약 조건을 제거되었기 때문에 성능이 좋았다.

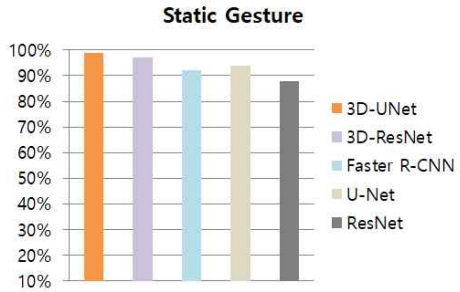


그림 15 정적 제스처에 대한 모델 별 데이터 학습의 정확도
Fig. 15. Model-specific Learning Accuracy for Static Gesture

Dynamic Gesture는 공간적인 특성뿐만 아니라 시간적인 특성도 고려해야 하는 제스처이다. 시간적인 특성과 공간적인 특성을 고려하다보니 2D CNN모델은 사용이 불가능하며 3D CNN모델만 사용 할 수 있었다.

그림 16은 Dynamic Gesture에 대한 모델 별 학습의 정확도를 나타내고 있다. 3D-UNet은 97%, 3D-ResNet은 94%의 결과가 도출 되었다. Dynamic Gesture의 모델 별 결과를 보았을 때 Segmentation한 3D-UNet 모델이 다양한 조명 조건, 복잡한 배경 및 피부색에 대한 제약 조건을 제거가 되었기 때문에 성능이 좋았다.

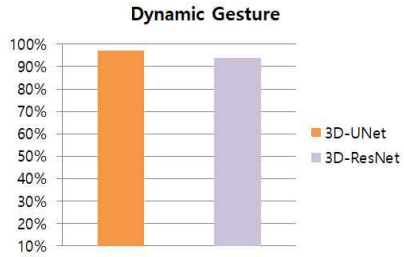


그림 19. 동적 제스처에 대한 모델 별 학습의 정확도
Fig. 19. Model-specific Learning Accuracy for Dynamic Gesture

4-2 각 제스처 인식률에 대한 성능평가 및 분석

표 1은 2D CNN 모델별 손 제스처 인식률과 인식속도를 나타내고 있다. 인식률을 봤을 때 ConvNet을 이용한 ResNet모델이 특징 추출의 정확도가 가장 낮았다. 낮은 이유로는 사람의 손의 경계선을 찾기 힘든 문제로 다른 모델에 비해 정확도가 낮았으며, Faster R-CNN의 경우 Object Detection 방법을 통하여 손 부분만 추출하면서 ResNet 모델의 문제점을 해결하였다. 그래서 Faster R-CNN모델은 손 쪽에만 집중할 수 있어 좀 더 높은 정확도가 나오게 되었다. 마지막 U-Net은 ResNet 모델의 문제도 해결하고 Faster R-CNN에 문제인 다양한 조명 조건, 복잡한 배경에 대한 조건도 해결 할 수 있어서 3개의 모델 중 정확도가 가장 높았다. 연산 속도는 모델의 연산량과 파라미터 수에 의해 파라미터 수에 의해 영향을 받아 표 1의 결과가 나왔다.

표 1. 2D CNN 모델별 인식률 및 인식속도
Table 1. 2D CNN Model-specific Recognition rate and Recognition speed

	ResNet		Faster R-CNN		U-Net	
	Rate	Speed	Rate	Speed	Rate	Speed
Point Gesture	92.9%	4.7ms	94.7%	5.1ms	96.9%	4.9ms
Stop Gesture	93.3%	4.4ms	95.3%	4.7ms	97.5%	4.4ms
Restore Gesture	91.6%	5.0ms	93.2%	5.3ms	96.6%	5.1ms
Average	92.6%	4.7ms	94.4%	5.0ms	97.0%	4.8ms

표 2는 3D CNN 모델별 손 제스처 인식률과 인식속도를 나타내고 있다. 3D-ResNet모델이 특징 추출의 정확도가 낮은데 낮은 이유로는 사람의 손의 경계선을 찾기 힘든 문제로 정확도가 낮았으며, 3D-UNet은 3D-ResNet 모델의 문제도 해결하고 다양한 조명 조건, 복잡한 배경에 대한 조건도 해결 할 수 있어서 2개의 모델 중 정확도가 높았다. 공간적인 특성만 고려하면 되는 Static 제스처가 시간적과 공간적인 특성을 모두 고려해야하는 Dynamic Gesture보다 성능이 전체적으로 좋다는 것도 알 수 있다. 연산 속도는 모델의 연산량과 파라미터 수에 의해 영향을 받아 표 2의 결과가 나왔다.

표 2. 3D CNN 모델별 인식률 및 인식속도
Table 2. 3D CNN Model-specific Recognition rate and Recognition speed

	3D-UNet		3D-ResNet	
	Rate	Speed	Rate	Speed
Point Gesture	99.2%	4.7ms	97.2%	4.6ms
Stop Gesture	99.7%	4.4ms	96.9%	4.4ms
Next Gesture	98.2%	7.2ms	97.2%	7.1ms
Previous Gesture	97.1%	7.2ms	95.1%	7.1ms
Enlargement Gesture	97.2%	7.6ms	97.2%	7.8ms
Reduction Gesture	98.8%	7.6ms	97.8%	7.7ms
Rotation Gesture	98.3%	8.1ms	97.3%	8.2ms
Average	98.3%	6.6ms	96.9%	6.7ms

Ⅴ. 결 론

본 논문에서는 가상공간에서 사용자 인터페이스로 사용될 Static Gesture와 Dynamic Gesture를 정의하고 RGB 카메라를 통하여 입력받은 손 제스처를 딥러닝 모델들을 이용하여 특징을 추출하고 인식하는 방법을 제안하였다.

여러 가지 딥러닝 모델을 통하여 다양한 방법으로 데이터를 학습시키고 특징을 추출하여 손 제스처를 인식해 보았다. 먼저 학습시킨 데이터에서 입력되는 영상 그대로를 추출하여 인식하는 방법, Segmentation을 활용하여 전처리 과정을 통해 복잡한 주변 환경으로부터 영향을 줄였고 손가락 인식 및 추적을 통하여 인식 속도와 정확성을 높이는 방법 그리고 Object Detection을 통한 방법을 이용해 보았다.

2D CNN 모델에서 손 제스처 인식률은 U-Net, Faster R-CNN, ResNet 순으로 좋았으며 인식속도는 Faster R-CNN, U-Net, ResNet 순으로 좋았다. U-Net은 ResNet 모델의 문제점인 사람의 손의 경계선을 찾기 힘든 문제를 해결하고 Faster R-CNN에 문제점인 다양한 조명 조건, 복잡한 배경에 대한 조건도 해결 할 수 있어서 3개의 모델 중 손 제스처 인식률이 가장 높은 것으로 분석된다. 연산 속도는 가각 모델의 연산량과 파라미터 수에 의해 영향을 받았음을 알 수 있었다.

3D CNN모델에서 손 제스처 인식률은 3D-UNet, 3D-ResNet 순으로 좋았으며 인식속도는 3D-ResNet, 3D-UNet 순으로 좋았다. 3D-UNet은 3D-ResNet 모델의 문제점인 사람의 손의 경계선을 찾기 힘들어 특징 추출의 정확도가 낮은 문제를 해결하고 다양한 조명 조건, 복잡한 배경에 대한 조건도 해결 할 수 있어서 2개의 모델 중 정확도가 높았다. 공간적인 특성만 고려하면 되는 Static Gesture가 시간과 공간적인 특성을 모두 고려해야하는 Dynamic

Gesture보다 성능이 전체적으로 좋다는 것도 알 수 있다.

향후 연구에서는 딥러닝을 이용하여 손 제스처가 아닌 몸의 동작을 인식하여 가상공간에서 사용자 인터페이스로 사용한다면 가격적인 면에서도 부담이 적어지게 될 것이고 동작 인식에 필요한 불필요한 센서들을 장착하지 않기 때문에 좀 더 편리하게 VR을 접할 수 있을 것으로 기대된다.

참고문헌

[1] J.K. Cha. “ Wearable sensor technology for virtual and augmented reality gesture recognition” *The Journal of Electronics Society*, VOl. 42 No. 6, pp. 63-70, 2015

[2] S. Lee, I. Jung, “A Design and Implementation of Natural User Interface System Using Kinect”, J. of Digital Contents, Vol.15, No.4, pp. 473-480, 2014

[3] Mathworks.[Internet]. Available : https://kr.mathworks.com/solutions/deep-learning/convolutional-neural-network.html

[4] Raon People[Internet]. Available :https://m.blog.naver.com/PostView.nhn?blogId=laonple&logNo=220800190798&proxyReferer=https%3A%2F%2Fwww.google.com%2F

[5] worb1605 [Internet]. Available : https://m.blog.naver.com/PostView.nhn?blogId=worb1605&logNo=221333597235&proxyReferer=https%3A%2F%2Fwww.google.com%2F

[6] Incredible.AI [Internet]. Available : http://incredible.ai/deep-learning/2018/03/17/Faster-R-CNN/

[7] Hello BLOG! [Internet]. Available : https://curt-park.github.io/2017-03-17/faster-rcnn/

[8] CDM[Internet]. Available : https://cdm98.tistory.com/35

[9] ResearchGate[Internet]. Available : https://www.researchgate.net/figure/Architecture-of-the-adaptive-lung-nodule-classifier-consisting-of-a-3D-ResNet-and-a_fig3_320442155

[10] Simulation/ML[Internet]. Available : https://jay.tech.blog/2017/02/02/3d-convolutional-networks/

[11] TAEWAN.KIM[Internet]. Available : http://taewan.kim/post/cnn/

[12] GitBook[Internet]. Available : https://deepbaksuvision.github.io/Modu_ObjectDetection/posts/01_00_What_is_Object_Detection.html

[13] Reniew’s blog[Internet]. Available : https://reniew.github.io/18/

임영재(Young-Jae Lim)

2018년 : 대전대학교 컴퓨터공학과 공학사

2018년~현재 : 대전대학교 대학원 컴퓨터공학과 석사과정

※관심분야 : 가상현실(Virtual Reality), 딥러닝(Deep Learning), 컴퓨터 그래픽스(Computer Graphics) 등

정일홍(Il-Hong Jung)

1993년: 애리조나 주립대학 컴퓨터공학과 졸업 (공학석사)

1998년: 애리조나 주립대학 컴퓨터공학과 졸업 (공학박사)

1998년 ~ 현 재 : 대전대학교 컴퓨터공학과 교수

※관심분야 : 컴퓨터 그래픽스(Computer Graphics), 멀티미디어(Multimedia), 가상현실(Virtual Reality), 딥러닝(Deep Learning) 등