

딥러닝 기반 실시간 손 제스처 인식

김규민[†], 백중환^{**}

Real-Time Hand Gesture Recognition Based on Deep Learning

Gyu-Min Kim[†], Joong-Hwan Baek^{**}

ABSTRACT

In this paper, we propose a real-time hand gesture recognition algorithm to eliminate the inconvenience of using hand controllers in VR applications. The user's 3D hand coordinate information is detected by leap motion sensor and then the coordinates are generated into two dimensional image. We classify hand gestures in real-time by learning the imaged 3D hand coordinate information through SSD(Single Shot multibox Detector) model which is one of CNN(Convolutional Neural Networks) models. We propose to use all 3 channels rather than only one channel. A sliding window technique is also proposed to recognize the gesture in real time when the user actually makes a gesture. An experiment was conducted to measure the recognition rate and learning performance of the proposed model. Our proposed model showed 99.88% recognition accuracy and showed higher usability than the existing algorithm.

Key words: Leap Motion, Deep Learning, VR, Gesture Recognition

1. 서 론

최근 AR(augmented reality)/VR(virtual reality)/MR(mixed reality) 시장 수요의 증대와 HMD(head mounted display) 및 모션 센서의 보급화에 따른 긍정적 산업 형성에 따라 인터랙티브 영상 음향 제어, 웹기반 마켓 플러그인/업그레이드 서비스 제공, 인터랙션 VR 서비스, 웹기반 프로그램 판매, 교육, 서비스지원 등 가상공간에서의 콘텐츠 산업이 발전하고 있다. 그에 따라 가상공간 콘텐츠 개발, 연구 산업이 글로벌 투자 확산에 따른 세계적 연구 산업으로 부각되고 있고 단순 콘텐츠를 넘어선 사용자의 오감을 자극하는 인간과 기계 사이의 상호 교감을 통한 참여와 몰입형 콘텐츠 시장이 확산되고 있다. 기존의

가상공간 콘텐츠로는 VR 게임을 대표적으로 생각할 수 있다. 현재 시장에 나와 있는 VR 게임은 게임 화면을 보여주는 HMD 이외에도 게임 콘텐츠 내부의 캐릭터나 사물을 컨트롤하기 위한 컨트롤러가 별도로 존재한다. 그러나 이러한 컨트롤러의 존재는 게임 플레이어가 게임을 즐기는 것에 위화감을 줄 수 있고, 나아가서는 게임에 몰입하는 것을 방해할 수도 있다. 그리고 게임 컨트롤러라는 사물에 종속되어 컨트롤을 하기 때문에 VR 게임을 즐기는데 제약이 발생한다[1,2,3].

본 논문에서는 가상환경에서 컨트롤러의 불편함을 대신할 보편적인 동적 제스처를 유형별로 분류하고, 이에 따른 실시간 손 제스처 인식 기법을 제안한다. Fig. 1은 전체적인 제스처 인식 시스템을 나타낸

※ Corresponding Author : Joong-Hwan Baek, Address: (10540) 76 Hanggongdaehag-ro Deogyang-gu Goyang-si, Korea, TEL : +82-2-300-0125, FAX : +82-2-3159-9257, E-mail : jhbaek@kau.ac.kr

Receipt date : Feb. 22, 2019, Revision date : Mar. 28, 2019
Approval date : Apr. 1, 2019

[†] School of Electronic & Information Eng., Korea Aerospace University (E-mail : gyumin46@naver.com)

^{**} School of Electronic & Information Eng., Korea Aerospace University

※ This work was supported by the GRR program of Gyeonggi province.[GRR Aviation 2019-B04, Development of Interactive VR Player and Service with Space Media Convergence]

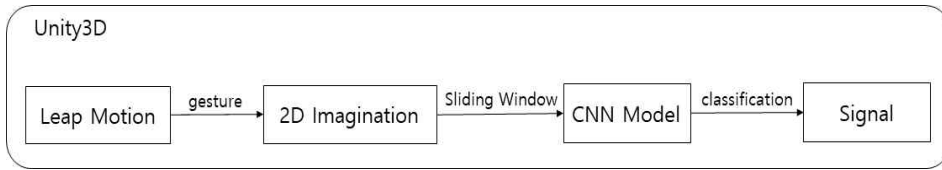


Fig. 1. Flowchart for proposed gesture recognition system.

다. 사용자의 3차원 손 좌표 정보를 립모션 센서를 통해 검출하고 X-Y 평면을 R 채널, Y-Z 평면을 G 채널, Z-X 평면을 B 채널로 만들고 이들을 합쳐 2차원의 RGB 이미지로 생성한다. 만들어진 이미지를 CNN 모델 중 하나인 SSD 모델을 통해 학습시킴으로써 손 제스처를 분류한다. 사용자가 제스처를 취할 시 실시간으로 제스처를 인식하기 위해 슬라이딩 윈도우 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서 2D 제스처를 인식하는 기존 연구 사례를 설명하고, 3장에서는 CNN 모델이 학습할 손 제스처 데이터 구성 및 전처리 방법에 대해서 설명한다. 그리고 4장에서는 CNN 모델의 핵심 구조를 설명하고, 5장에서는 실시간 제스처 인식을 위한 슬라이딩 윈도우 기법을 설명한다. 6장에서는 제안한 CNN 모델의 제스처 학습 인식 실험을 진행한다. 마지막 7장에서는 결론을 맺으며 향후 연구 방향을 제시한다.

2. 기존 연구

최근 손동작 인식이 가능한 장치로 센서 기반의 립모션, 리얼센스, MYO 등이 출시되어 이를 활용한 연구가 활발히 진행되고 있다[4,5]. 이에 따라, 가상 현실 응용에 사용되기 위한 자연스러운 사용자의 인터페이스(natural user interface, NUI)에 대한 연구가 활발히 진행되고 있다. 그 중 많이 사용되고 있는 사용자 인터페이스는 제스처이다. 제스처는 사용자가 자신의 의도를 전달하기 위해 수행하는 의도된 동작뿐만 아니라 무의식 중에 의미 없이 수행하는 동작을 포함하고 있다.

제스처를 인식하는 방법에 대한 연구는 다양한 곳에서 지속적으로 수행되어 왔다. 이에 제스처 인식을 통해 응용 프로그램을 제어하는 방법에 대한 연구가 있었지만 인식에 있어 한계가 있고 이점을 보완하고 제스처 인식의 정확도를 높이기 위해 머신러닝을 활용한 연구가 진행되고 있다[6,7,8].

제스처 데이터는 연속성을 가지고 입력되기 때문에, HMM(hidden markov model), K-NN(k-nearest neighbors) 등의 알고리즘이 사용되어 왔으며, 기존의 기계학습 알고리즘으로 SVM(support vector machine), Random Forest 등의 알고리즘을 활용하여 제스처의 인식률을 높이는 방향으로 발전되어 왔다[9,[10,11]. 기존 연구로 최 근접 이웃 알고리즘을 활용한 제스처 인식에 대한 연구, 키넥트와 립모션을 같이 사용하여 SVM 알고리즘으로 제스처를 인식하는 연구, HMM을 이용한 제스처 인식에 관한 연구, CNN을 이용한 제스처 인식에 관한 연구 등이 진행되었다[12,13,14,15]. 그러나 앞선 연구들은 유사 제스처에 대한 분류와 실사용 측면에 있어 인식률이 떨어지는 단점이 있었으며 3차원 정보를 2차원으로 투사할 때 발생하는 정보의 손실이 있다. 본 연구에서는 기존 연구에서 언급된 제한점을 통해 효율적이고 보완된 제스처 인식 알고리즘을 제안하고 구축한다.









3. 손 제스처 데이터 구성 및 전처리 방법

3.1 손 제스처 데이터 구성 및 수집

손 제스처 기술은 사람이 손을 이용하여 미리 정해진 동작을 했을 때, 그것이 어떤 동작인지를 인식하는 기술을 말한다. SSD 모델 학습을 위한 손 제스처들은 구별 가능한 동적 제스처 8개를 정의하였으며 패턴은 Table 1과 같다. 학습 데이터 960개와 테스트 데이터 240개로 구성하였으며 데이터를 수집하기 위해 3명의 학습 데이터 수집 실험 군과 2명의 테스트 데이터 수집 실험 군이 직접 손으로 제스처를 취하여 제스처의 궤적을 이미지로 변환하여 저장하였다.

데이터 수집 환경은 Unity3D 게임 엔진을 활용하여 GUI(graphical user interface)를 구성하여 손 제스처 데이터를 수집했다. Fig. 2는 립 모션을 활용하여 손 제스처 데이터를 수집하고 있는 화면을 보여준다. 좌측 상단에는 Unity3D에서 실행되고 있는 FPS(frame per second)와 한 개의 프레임이 생성되는데

Table 1. 8 hand gesture patterns

Symbol	Hand gesture	Symbol	Hand gesture
1. V		5. Left->Right	
2. Alpha		6. Half-Circle	
3. Circle		7. Stab	
4. Lightening		8. Triangle	

걸린 시간을 표시했다.

손 제스처 데이터 수집 과정은 Unity3D 내부에 있는 일시정지 기능을 활용하여 제스처의 시작과 끝을 구성했다. 그리고 수집된 데이터는 3차원 손 중심 좌표의 궤적으로 이루어져있으며 각 제스처의 라벨링 폴더에 저장된다.

3.2 제스처 데이터 전처리기

본 연구에서는 SSD 모델 학습을 위해 수집된 3차원 손 좌표 데이터를 2차원 이미지로 투영시키는 과정이 필요하다. 이미지의 크기는 300×300이며 제스처 데이터를 취득 후 전처리 과정을 Fig. 2과 같이 수행한다. 립모션을 통해 입력받은 손 중심 좌표는 0~1까지의 범위로 나타난다. 이를 300×300 크기의 이미지로 투영시키기 위해 좌표 정보를 0~300 범위로 정규화 하였다. 이를 위한 수식은 아래 식 (1), (2), (3)과 같다.

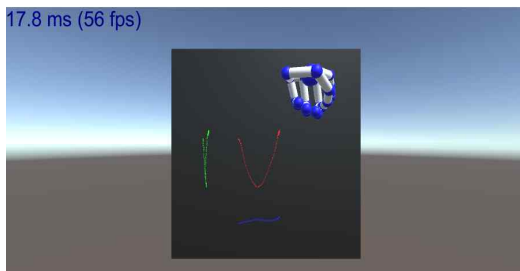


Fig. 2. GUI for data collection.

$$X = 300 \cdot (x_{hand} - x_{min}) / (x_{max} - x_{min}) \quad (1)$$

$$Y = 300 \cdot (y_{hand} - y_{min}) / (y_{max} - y_{min}) \quad (2)$$

$$Z = 300 \cdot (z_{hand} - z_{min}) / (z_{max} - z_{min}) \quad (3)$$

여기에서 $x_{hand}, y_{hand}, z_{hand}$ 는 Unity3D에서 취득되고 있는 손바닥 중심좌표, $x_{min}, y_{min}, z_{min}$ 은 손바닥 중심좌표의 최솟값, $x_{max}, y_{max}, z_{max}$ 은 손바닥 중심좌표의 최댓값, X, Y, Z 는 이미지에 투영될 정규화된 좌표이다.

정규화 과정을 통해 생성된 데이터를 2차원 이미지로 투영시키기 위해 R채널에 X, Y , G채널에 Y, Z , B채널에 Z, X 를 대입한다. 그 후, R, G, B채널을 전부 포함한 RGB 2차원 이미지를 생성한다. 만들어진 이미지는 손 제스처의 이동 궤적을 좌표로 나타내기 때문에 픽셀 각각의 좌표로 표현할 시 인식이 저하될 수 있다. 이러한 현상을 줄이기 위해서 모폴로지 연산을 수행한다. 팽창과 침식 함수는 식 (4), (5)와 같이 정의된다.

$$f \oplus S = \cup_{x \in f} S_x \quad (4)$$

$$f \ominus S = \{x \mid x + s \in f, \forall s \in S\} \quad (5)$$

여기에서 f 는 이미지 화소의 집합, S 는 구조요소이다. 학습 데이터를 만드는 실험 군이 제스처를 입력할 때 사용자가 데이터를 입력하는 시간이 매 업데이트마다 데이터를 처리하는 시간보다 짧을 경우, 데이터를 이미지로 변환할 때 일부 픽셀이 누락되는 현상을 줄이기 위해서 팽창 연산을 수행한다. 또한

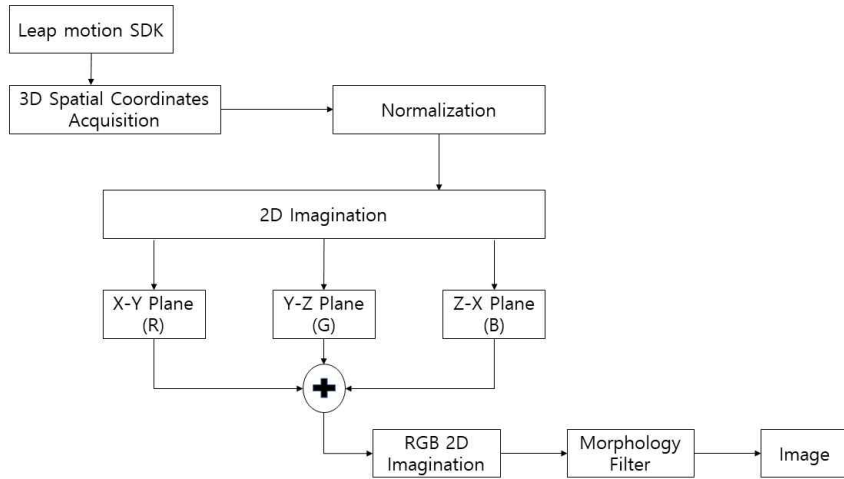


Fig. 3. Flowchart for organizing gesture learning data.

실험 군이 데이터를 입력 할 때 잘못 들어가는 일부 픽셀을 없애기 위해 침식 연산을 수행한다. Fig. 4와 Fig. 5는 모폴로지 연산을 수행하기 전과 수행한 후 그림이다. Fig. 4 이미지를 보면 픽셀이 각각의 좌표로 표현되어 흐릿한 현상을 볼 수 있다. 이는 SSD 모델의 학습 이미지로 입력 될 시 인식이 저하될 수 있다. 최종적으로 모폴로지 연산을 통해 데이터의 손실을 최소화시켜 정상적인 이미지를 취득 할 수 있다. Fig. 5 이미지를 보면 300×300 크기의 이미지에 맞춰 만들어진 학습 제스처 이미지를 확인 할 수 있다.

방식이나 스타일, 왼손 혹은 오른손 사용 여부 등에 따라 다양한 모양을 나타낸다. 이는 제스처를 인식하는 데 있어 복잡도를 심화시킨다. 그러나 CNN은 입력 데이터의 이동, 왜곡, 크기, 기울어짐, 시점 등에도 불구하고 패턴 인식에 있어 효과적이다. CNN은 기존의 신경망과 다르게 입력 원본 데이터를 바로 연산하여 출력하지 않고, 특징(feature) 추출 단계와 분류화(classification) 단계를 거쳐 결과값을 내는 것이 특징이다. CNN의 구조는 컨벌루션층(convolution layer), 풀링층(pooling layer), 완전 연결층(fully-connected layer)으로 구성되어 있다.

4. 제스처 인식을 위한 CNN 모델

4.1 CNN(Convolution Neural Network)

일반적으로 제스처 패턴의 경우 사용자가 취하는

4.2 SSD(Single Shot multibox Detector)

본 연구에서는 신경망을 토대로 깊은 심층 구조를 형성하여 3차원 제스처 인식을 수행했다. 사용된 모델은 Fig. 7과 같다. SSD는 사물 인식을 위한 CNN

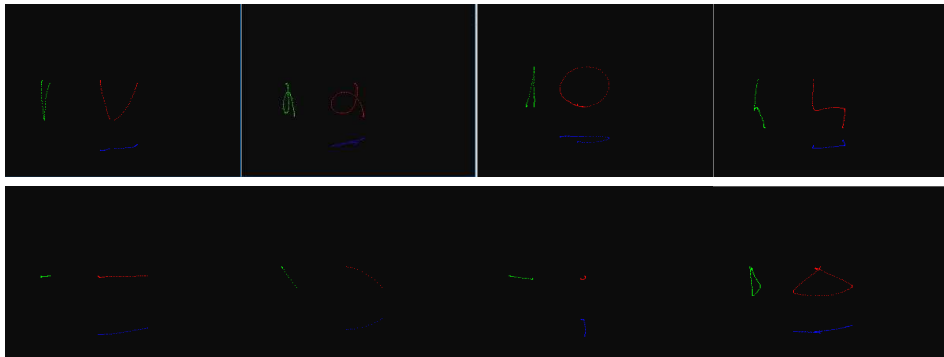


Fig. 4. Result of gesture images before morphology processing.

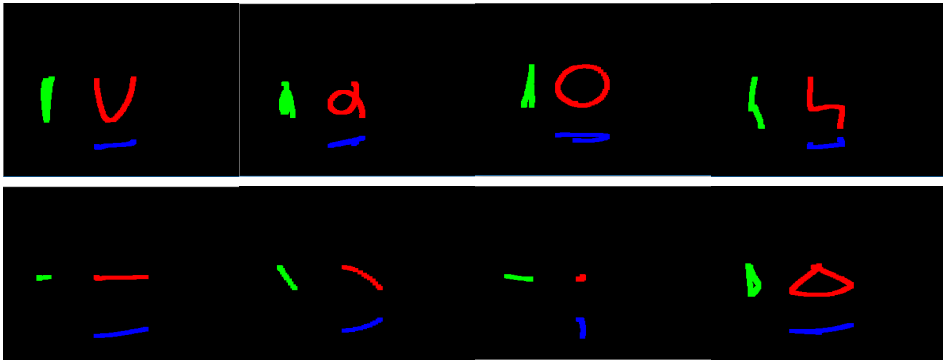


Fig. 5. Result of gesture images after morphology processing.

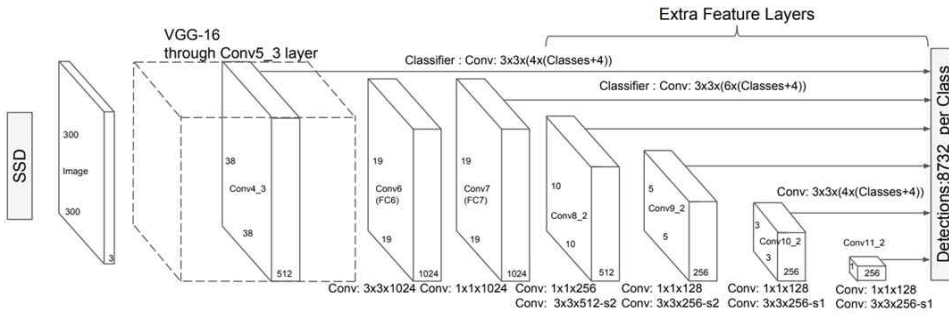


Fig. 6. SSD model.

모델 기반 물체 인식 및 검출 알고리즘이다. 특징으로는 VGG-16을 기본으로 사용하며 하나의 심층 신경망을 사용하여 이미지의 객체를 검출한다. SSD는 여러 히든 레이어에 정보가 분산되어 있다. Conv4_3, conv7, conv8_2, conv9_2, conv10_2, conv11_2을 입력으로 컨벌루션 하여 생성된 6개의 특징 맵 안에는 경계박스과 클래스 정보가 담겨있다. 이 특징 맵의 크기는 모두 다르며 가로세로 크기가 38*38, 19*19, 10*10, 5*5, 3*3, 1*1 로 점점 작아진다. 예측 경계박스의 총 개수를 보면 각 클래스 당 8,732개의 경계박

스를 예측하며 예측 중에서 신뢰도가 가장 큰 것만 남기고 나머지는 모두 지우는 NMS(Non-Maximum Supression) 알고리즘을 사용한다. 위 구조를 통해 위치 추정 및 입력 영상의 resampling 과정 없이도 정확도 높은 결과를 도출할 수 있다.

학습 과정에서는 Fig. 8과 같은 과정을 거친다. SSD에 학습하기 위해서는 Fig. 8(a)와 같이 입력영상과 학습하고자 하는 객체의 ground truth 상자가

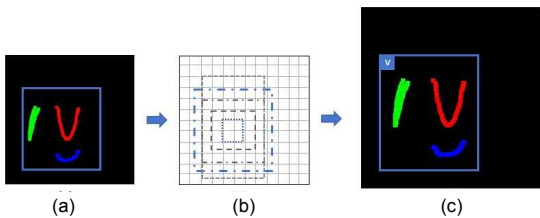


Fig. 7. Process of object detection and classification (a) Image with ground truth boxes (b) 12*12 feature map (c) Classification.

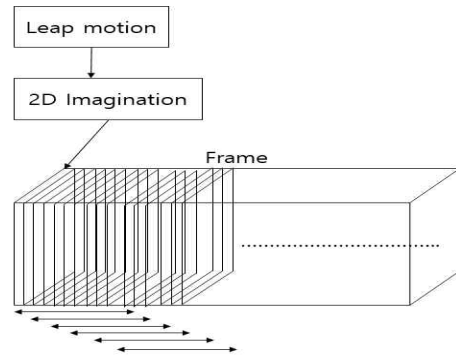


Fig. 8. Sliding window method.

필요하다. Fig. 8 (b)의 12x12 크기의 특징 맵으로 가정하면 12*12 특징맵에서는 V 제스처 이미지의 모든 경계박스 후보를 찾는다. Confidence loss가 높은 순으로 디폴트 박스를 정렬하여 positive와 negative의 비율이 1:3이 되도록 confidence loss가 높은 것만 선별하여 학습시킨다. 오브젝트에 대한 confidence loss가 큰 것만을 줄이는 방향으로 학습시키면 결국 모든 학습 데이터에 대한 confidence loss가 점차 낮아진다. 최종적으로 학습 후, 목표 대상을 인식 후 Fig. 8(d)와 같이 사용자가 설정한 class 별로 분류한다.

5. 슬라이딩 윈도우 기법

본 연구에서는 실시간으로 제스처를 인식하기 위해 Fig. 9와 같이 슬라이딩 윈도우 기법을 제안한다. 사용자가 립모션을 통해 제스처를 취할 시 이미지가 Unity3D에서 생성된다. 실험을 통해 본 연구에서 정의된 제스처를 사용자가 완료하는 데는 평균적으로 50 프레임이 소요되는 것을 확인하였다. 따라서 슬라이딩 윈도우 구간 크기를 50 프레임으로 설정 후, 0~50 프레임까지 취한 제스처를 누적하여 SSD 모델 구조에 입력하여 제스처를 분류한다. 그 후, 사용자가 프로그램을 종료하기 전까지 5 프레임씩 이동시키며 계속 누적된 매 50 프레임씩의 제스처 이미지를

Table 2. Result of gesture recognition accuracy

Batch size	Learning rate	Iteration	Accuracy
24	0.01	50,000	99.8%

Table 3. Comparison of classification rates between conventional and proposed methods

Class	Conventional method (R) Accuracy	Proposed method (RGB) Accuracy
1(V)	90%	100%
2(Alpha)	100%	100%
3(Circle)	100%	100%
4(Lightening)	100%	100%
5(Left->Right)	100%	100%
6(Half-Circle)	90%	100%
7(Stab)	86%	100%
8(Triangle)	90%	99%
Average	97.50%	100.00%

순차적으로 SSD 모델에 입력하여 분류한다. 사용자가 설정한 임계값에 따라 임계값을 초과하는 인식을만 제스처로 정의된다. 본 연구에서 제안하는 슬라이딩 윈도우 방법을 통해 실시간으로 높은 인식의 정확도를 보이는 것을 확인할 수 있었다.

6. 실험 결과

본 연구는 프로세서 Intel Core I7 7700, 그래픽 카드 GTX 1080 Ti, RAM 16GB, Visual Studio 2017 환경에서 개발되었다. 딥러닝 학습을 위한 프레임워크로 Caffe를 이용하였으며 손 제스처 인터페이스를 적용한 시뮬레이션 개발을 위하여 Opencv 2.4을 기반한 OpencvforUnity 라이브러리와 Unity3D 2017.3.1.f1 엔진을 사용하였다.

6.1 제스처 이미지 분류 실험

본 연구에서는 SSD 모델을 활용하여 8 종류의 제스처 분류 실험을 진행하였다. 데이터 셋의 크기는 학습 데이터와 테스트 데이터의 합이며, 학습 데이터는 3명의 실험자에 대한 데이터이고, 테스트 데이터는 2명의 피실험자에 대한 데이터이다. 학습한 데이터의 개수는 총 960개이며 정확도 측정을 위한 테스트 데이터는 제스처별 30개씩 총 240개이다. 아래 Table 2는 본 연구에서 제시한 SSD 모델 학습 환경 변수와 분류 결과로 99% 이상의 인식의 정확도를 나타낸다.

기존 연구에서 사용했던 1개 채널을 이용한 CNN 기반 제스처 인식과 본 논문에서 제시한 모델의 인식의 정확도를 비교하는 실험을 진행했다[15]. 실험은 Table 2와 같이 동일한 환경 변수의 SSD 모델을 사

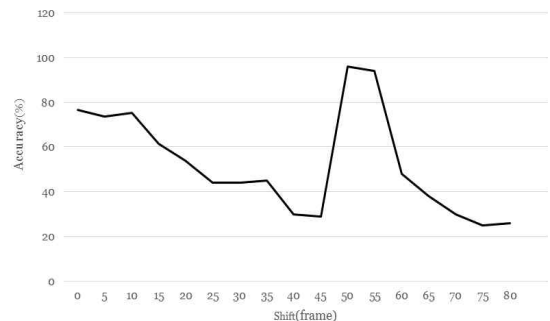


Fig. 9. Result of Class 1(V) gesture recognition using sliding window method

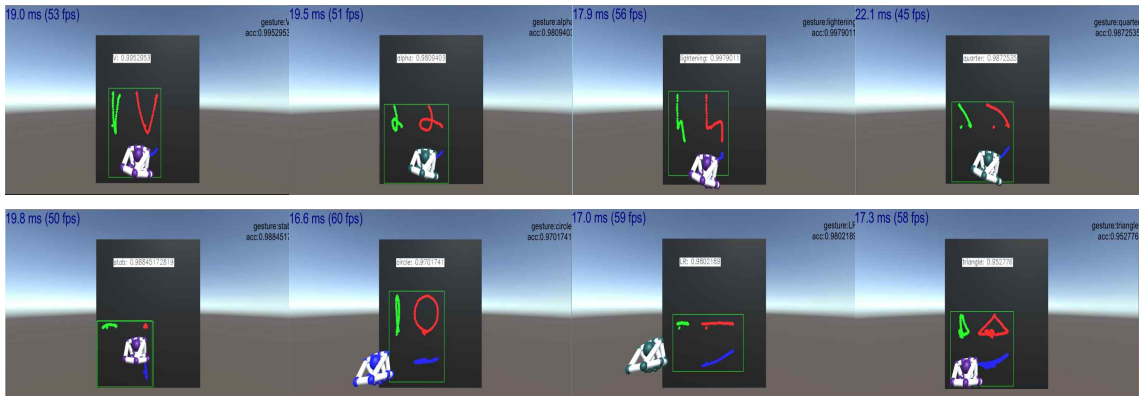


Fig. 10. Result of gesture classification.

용하였다. 각 제스처마다 30개의 입력영상을 테스트 데이터로 입력하였으며 분류기에서 나온 인식의 정확도를 Table 3에 보인다.

Table 3을 보면 본 연구에서 제안한 모델은 평균적으로 99.88%의 높은 인식의 정확도를 나타냈고, 기존 방법인 1개의 채널(R)만 사용할 때는 평균 94.5%의 정확도를 보였다. 각 제스처의 패턴 특징을 살펴보았을 때, 1개의 채널만 사용했을 경우, 3차원 제스처 이미지를 정면으로 보기 때문에 구분하기 힘든 6번, 7번, 8번 패턴은 인식이 저하되는 것을 볼 수 있다. 정면으로 보는 패턴의 특징이 단순할수록 다른 패턴들과 유사성이 크기 때문에 본 연구에서 제안한 알고리즘의 인식의 정확도가 높다.

6.2 Unity3D에서의 제스처 인식 실험

Fig. 9는 Table 3의 class 1(V)을 본 연구에서 제안한 슬라이딩 윈도우 기법을 사용할 때의 정확도 결과이다. 제스처로 인식되기 위한 임계값은 0.8로 설정하였다. 즉 80%이상의 정확도를 가질 때만 제스처로 인식되었다. Fig. 10은 학습된 모델을 이용하여 손 제스처에 대해 생성된 이미지와 그에 따른 인식 결과 화면을 보인다. 제스처 학습 데이터 영상과 동일하게 립모션에서 취득되는 손 중심좌표 3개를 컬러채널로 변환하였으며, 본 연구에서 제안한 슬라이딩 윈도우 기법을 이용하여 실시간으로 사용자의 제스처를 인식하였다.

7. 결 론

본 논문에서는 실시간으로 제스처를 분류하기 위

해 CNN 모델 중 SSD를 활용하고 슬라이딩 윈도우 기법을 제안하였다. 총 8개의 제스처 종류를 구성하고 Unity3D와 립모션을 이용하여 제스처 데이터를 수집했다. 3차원 제스처 공간좌표를 제안모델에 학습하기 위해서 3개의 채널을 이용하여 2차원 이미지로 투영시켰으며, 제안 모델은 신경망을 토대로 심층 구조를 형성한 SSD를 활용했다. 제안한 모델의 인식률과 학습 성능을 측정한 실험에서 제안한 모델은 99.88%로 상대적으로 적은 학습 데이터로도 높은 인식의 정확도를 확인할 수 있었다. 기존의 1개 채널만을 사용하는 알고리즘에 비해 3개의 채널을 사용함으로써 3차원적인 모션이 더 잘 검출되어 높은 인식률을 보였다. 또한 슬라이딩 윈도우 기법을 통해 실시간으로 제스처를 인식할 수 있었다. 향후 연구로 제스처 종류를 늘리고 다양한 모델에 대한 실험을 수행할 필요가 있다.

REFERENCE

- [1] J.J. Chae, J.H. Lim, H.S. Kim, and J.J. Lee “Study on Real-time Gesture Recognition Based on Convolutional Neural Network for Game Applications,” *Journal of Korea Multi-media Society*, Vol. 20, No. 5, pp. 835-843, 2017.
- [2] M.J. Kim, J. Heo, J.H. Kim, S.Y. Park, and J. Chang, “Development and Evaluation of Leapmotion-based Game Interface Considering Intuitive Hand Gestures,” *Journal of the Korean Society for Computer Game*, Vol. 27,

No. 4, pp. 69-75, 2014.

[3] J. Gu and D. Shin, "An Empirical Evaluation of the Representative Hand Gesture of Task from User Perspective," *Archives of Design Research*, Vol. 28, No. 1, pp. 133-145, 2015.

[4] I. Sin, D. Cheon, and H. Park, "Implementing Leap-motion-based Interface for Enhancing the Realism of Shooter Games," *Journal of The Human Computer Interaction Society of Korea*, Vol. 22, No. 1, pp. 67-74, 2017.

[5] H. Ju, M. Jo, S. In, G. Jo, and J. Min, "Development of Baseball Game Using Leap Motion Controllers," *Korean Institute of Information Scientists and Engineers Transactions on Computing Practices*, Vol. 21, No. 5, pp. 343-350, 2015.

[6] M. Go, G. Lee, C. Kim, J. An, and I. Kim, "An Implementation of User Interface Using Vision-based Gesture Recognition," *Korean Institute of Information Scientists and Engineers*, Vol. 35, No.1, pp. 507-511, 2008.

[7] J. Jeong, J. Kim, D. Kim, S. Gwon, S. Ju, W. O et. al., "A Hand Gesture Interface for Controlling 3D e-Books," *Journal of The Korean Society for Computer Game*, Vol. 25, No. 4, pp. 119-127, 2012.

[8] J. Suarez and R.R. Murphy, "Hand Gesture Recognition with Depth Images: A Review," *Proceeding of 2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, pp. 411-417, 2012.

[9] H.K. Lee and J.H. Kim, "An HMM-based Threshold Model Approach for Gesture Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 10, pp. 961-973, 1999.

[10] C. Cortes and V. Vapnik, "Support-vector Networks," *Machine Learning*, Vol. 20, No. 3, pp. 273-297, 1995.

[11] L. Breiman, "Random Forests," *Machine Learning*, Vol. 45, pp. 5-32, 2001.

[12] J. Chae, J. Im, and J. Lee, "Gesture Classification Based on k-Nearest Neighbors Algorithm for Game Interface," *Journal of Korea Multimedia Society*, Vol. 19, No. 5, pp. 874-880, 2016.

[13] G. Marin, F. Dominio, and P. Zanuttigh, "Hand Gesture Recognition with Leap Motion and Kinect Devices," *Proceeding of 2014 IEEE International Conference on Image Processing*, pp. 1565-1569, 2014.

[14] D. Song, D. Kim, and C. Lee, "The Chinese Characters Learning Contents Based on Gesture Recognition Using HMM Algorithm," *Journal of Korea Multimedia Society*, Vol. 15, No. 8, pp. 1067-1074, 2012.

[15] R. McCartney, J. Yuan, and H.P. Bischof, "Gesture Recognition with the Leap Motion Controller," *Proceeding of International Conference on Image Processing, Computer Vision, and Pattern Recognition*, pp. 3-9, 2015.



김 규 민

2013년 3월 ~ 현재 한국항공대학교 전자 및 항공전자공학 학사과정
 관심분야: 영상처리, 패턴인식, 가상현실, 딥 러닝



백 중 환

1981년 한국항공대학교 항공통신공학 졸업(공학사)
 1987년 오클라호마주립대학원 전기 및 컴퓨터공학 졸업(공학석사)
 1991년 오클라호마주립대학원 전기 및 컴퓨터공학 졸업(공학박사)

1992년~현재 한국항공대학교 항공전자정보공학부 교수
 관심분야 : 영상처리, 패턴인식, 멀티미디어, 가상현실, 딥 러닝