특집논문 (Special Paper)
방송공학회논문지 제25권 제4호, 2020년 7월 (JBE Vol. 25, No. 4, July 2020)
https://doi.org/10.5909/JBE.2020.25.4.545
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

제스처 인식 기반의 인터랙티브 미디어 콘텐츠 제작 프레임워크 구현

고 유 진[a], 김 태 원[a], 김 용 구[a], 최 유 주[a]‡

Implementation of Interactive Media Content Production Framework based on Gesture Recognition

You-jin Koh[a], Tae-Won Kim[a], Yong-Goo Kim[a], and Yoo-Joo Choi[a]‡

요 약

본 논문에서는 사용자의 제스처에 따라 반응하는 인터랙티브 미디어 콘텐츠를 프로그래밍 경험이 없는 사용자가 쉽게 제작할 수 있도록 하는 콘텐츠 제작 프레임워크를 제안한다. 제안 프레임워크에서 사용자는 사용하는 제스처와 이에 반응하는 미디어의 효과를 번호로 정의하고, 텍스트 기반의 구성 파일에서 이를 연결한다. 제안 프레임워크에서는 사용자의 제스처에 따라 반응하는 인터랙티브 미디어 콘텐츠를 사용자의 위치를 추적하여 프로젝션 시키기 위하여 동적 프로젝션 맵핑 모듈과 연결하였다. 또한, 제스처 인식을 위한 처리 속도와 메모리 부담을 줄이기 위하여 사용자의 움직임을 그레이 스케일(gray scale)의 모션 히스토리 이미지(Motion history image)로 표현하고, 이를 입력 데이터로 사용하는 제스처 인식을 위한 합성곱 신경망(Convolutional Neural Network) 모델을 설계하였다. 5가지 제스처를 인식하는 실험을 통하여 합성곱 신경망 모델의 계층수와 하이퍼파라미터를 결정하고 이를 제안 프레임워크에 적용하였다. 제스처 인식 실험에서 97.96%의 인식률과 12.04 FPS의 처리속도를 획득하였고, 3가지 파티클 효과와 연결한 실험에서 사용자의 움직임에 따라 의도하는 적절한 미디어 효과가 실시간으로 보임을 확인하였다.

Abstract

In this paper, we propose a content creation framework that enables users without programming experience to easily create interactive media content that responds to user gestures. In the proposed framework, users define the gestures they use and the media effects that respond to them by numbers, and link them in a text-based configuration file. In the proposed framework, the interactive media content that responds to the user's gesture is linked with the dynamic projection mapping module to track the user's location and project the media effects onto the user. To reduce the processing speed and memory burden of the gesture recognition, the user's movement is expressed as a gray scale motion history image. We designed a convolutional neural network model for gesture recognition using motion history images as input data. The number of network layers and hyperparameters of the convolutional neural network model were determined through experiments that recognize five gestures, and applied to the proposed framework. In the gesture recognition experiment, we obtained a recognition accuracy of 97.96% and a processing speed of 12.04 FPS. In the experiment connected with the three media effects, we confirmed that the intended media effect was appropriately displayed in real-time according to the user's gesture.

Keyword : gesture recognition, interactive media, dynamic projection mapping, deep learning, convolutional neural network

Ⅰ. 서 론

프로젝션 맵핑은 투사하는 대상의 표면에 가상의 무대 공간을 창조할 수 있기 때문에 빠른 무대 장면 전환을 통한 이야기 전개가 가능하다. 전통적인 극장 공연 및 다양한 대형 행사의 무대 연출에서 현대의 첨단 기술은 해당 콘텐츠 생산에 핵심적인 요소로 작동하고 있다[1]. 기존의 무대 제작 방식에 비해 창작자의 의도대로 무대 연출이 자유롭고 물리적, 경제적 한계가 적으며 스토리에 따라 쉬는 시간 없이 빠르게 전환이 가능한 무대 연출은 관객의 흥미와 몰입을 강화시키는 요인 중 하나이다. 이에 따라 디지털 사이니지(Digital Signage)와 같은 마케팅 분야를 포함하여 공공장소 디스플레이, 전시 공간, 공연 무대, 패션쇼 등 다양한 분야에서 맵핑을 활용한 융복합 콘텐츠가 퍼져나가고 있다[2]. 특히 실시간으로 사물이나 사람의 위치와 동작 등을 추적하여 그에 어울리는 효과를 발생시키는 '제스처 기반의 동적 프로젝션 맵핑 콘텐츠'는 융복합 인터랙티브 미디어 콘텐츠들 중 하나로, 공연 예술계에서 지속적으로 시도되고 있다. 하지만 동적 프로젝션 맵핑 콘텐츠 제작은 사전에 필요한 준비 과정이 많고 구현 난이도가 높은데다 기존에 배포되어 있는 프로젝션 맵핑 유료 상용화 툴은 제스처 기반의 동적 프로젝션 맵핑을 지원하지 않으며 개발자가 직접 툴을 수정하는 것이 불가능하다[3]. 따라서 공연자의 동작에 따른 효율적인 미디어 이펙트에 대한 처리가 복잡한 프로그래밍 과정 없이 용이하게 이루어질 수 있도록 하는 '인터랙티브 콘텐츠 제작 프레임워크'가 요구되고 있다.

이에 본 논문에서는 사용자의 제스처에 따라 반응하는 인터랙티브 미디어 콘텐츠를 프로그래밍 경험이 없는 사용자가 쉽게 제작할 수 있도록 하는 콘텐츠 제작 프레임워크를 제안한다. 제안 프레임워크에서 사용자는 제스처와 파티클 효과를 숫자로 표현하고, 텍스트 기반의 구성화일에서 제스처와 이에 대응하는 파티클 효과를 정의함으로써 별도의 프로그래밍 과정 없이 사용자 제스처에 반응하는 인터랙티브 미디어 콘텐츠를 제작할 수 있도록 한다. 또한, 사용자의 제스처에 따라 반응하는 미디어 콘텐츠를 사용자의 위치를 추적하여 프로젝션 시키기 위하여 동적 프로젝션 맵핑 모듈과 연결한다. 높은 인식률의 효율적인 제스처 인식을 위하여 의미있는 제스처를 한 장의 흑백 이미지로 표현하고, 이를 입력으로 사용하는 모션 히스토리 영상(motion history image) 기반의 제스처 인식 합성곱 신경망(Convolutional Neural Network) 모델을 설계하고, 이를 제안 프레임워크에 적용한다. 마지막으로 제안 프레임워크를 활용한 콘텐츠 제작 실험을 통하여 제안 프레임워크의 성능과 유용성을 입증하고자 한다.

a) 서울미디어대학원대학교 뉴미디어학부 미디어공학&인공지능 응용소프트웨어학과(Department of Newmedia, Seoul Media Institute of Technology, Seoul, South Korea)
‡ Corresponding Author : 최유주(Yoo-Joo Choi)
E-mail: yjchoi@smit.ac.kr
Tel: +82-2-6393-3235
ORCID: https://orcid.org/0000-0001-7520-097X
※ This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF-2017R1D1A1B03035718) funded by the Ministry of Education.
· Manuscript received June 17, 2020; Revised June 23, 2020; Accepted June 23, 2020.

Ⅱ. 관련 연구

1. 제스처 인식을 위한 딥러닝 기반의 접근법 분류

제스처 인식은 적외선 혹은 이미지 센서와 같은 다양한 센서를 통해 사용자 신체의 움직임을 인식하여 컴퓨터와 상호작용하는 동작 인식 기술들 중 하나이다. 제스처 인식 기술을 위해서는 사용자가 수행한 제스처를 입력 받을 수 있는 센서가 필요하며 두 가지의 형태로 나뉜다. 센서나 장치를 사용자가 신체로 직접 접촉하여 데이터를 획득하는 접촉식 방식과 원거리 및 근거리 센서를 이용하여 데이터를 획득하는 비접촉식 방식이 있다[4]. 최근에는 큰 규모의 데이터 세트 수집이 용이해지고 GPU를 활용한 고성능 컴퓨팅이 보편화됨에 따라 딥러닝(Deep Learning) 기술을 이용하여 제스처를 인식하고자 하는 연구들이 이어지고 있다[5].

제스처 인식을 위한 딥러닝 접근법은 [표 1]과 같이 크게 세 분류로 나뉜다[6]. 첫 번째 접근 방법은 합성곱 신경망(Convolutional Neural Network)에서 3차원 필터를 사용하는 방법으로서 공간적 및 시간적 차원에서 차별적인 특성을 추출한다. 두 번째 접근 방법은 2차원 옵티컬 플로우 지도(optical flow map)와 같은 모션 특성(motion features)을 미리 계산하고, 이를 네트워크의 입력으로 사용하는 방법

표 1. 제스처 및 동작 인식을 위한 딥러닝 접근법[10]
Table 1. Deep learning approach for gesture recognition[10]

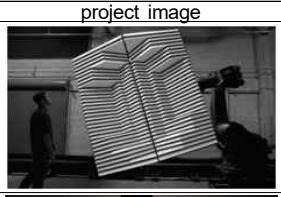
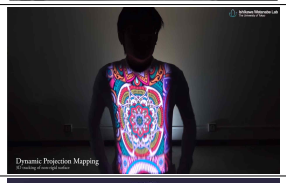
Gesture recognition approaches	3D models (3D convolution a pool)	
	Motion-based input features	
	Temporal methods	2D Models + RNN + LSTM
		2D Models + B-RNN + LSTM
		2D Models + H-RNN + LSTM
		2D Models + D-RNN + LSTM
		2D Models + HMM
		2D/3D Models + Auxiliary outputs
		2D/3D Models + Hand-crafted features

들이다. 세 번째 접근 방법은 RNN(Recurrent Neural Network)과 LSTM(Long Short Term Memory)을 결합하는 등 연속 데이터(Sequence Data)를 활용하는 시간적 방법(Temporal methods)들이다.

2. 동적 프로젝션 맵핑 기반의 인터랙티브 미디어 아트

프로젝션 맵핑(Projection Mapping)은 실제 건물이나 사물의 표면에 알맞은 콘텐츠의 영상을 투영하는 실감형 인터페이스의 미디어아트 기법이다. 미디어아트 분야에서는

표 2. 동적 프로젝션 맵핑의 사례
Table 2. Projects of dynamic projection mapping

project name	description	project image
bot&dolly - Box[9]	It recognizes the movement of the rectangular plane attached to the robot's arm and maps the stereoscopic image to realize the 3D image.	
Ishikawa group Lab - DynaFlash v2 and Post Reality[10]	The new high-speed projector DynaFlash v2 (3-LED + 1-DMD) was developed to naturally express the texture of flexible surfaces.	
connected colors / real-time face tracking and 3d projection mapping[11]	Track a person's face in real time. The points attached to the face are tracked and mapped with a 3D scanner.	
' 緣 ' NMARA Interdisciplinary Art Performance Works[12]	It uses a kinect camera to track the performer's movements in real time and distributes appropriate video effects.	
Virtual Reality Interactive Sandbox (Contour Line)[13]	The color is divided according to the height of the sand to express the contour line.	
Interactive wall and Floor Projection[14]	The color or shape of the object is changed according to the gesture of touching the mapped wall or floor.	

프로젝션 기반의 공간 증강현실기법을 프로젝션 맵핑으로 부르고 있다[7]. 프로젝션 맵핑은 미디어 파사드(facade)와 같이 대형 건물에 화려한 영상을 맵핑하거나 전시, 공연 및 마케팅 분야에서 관객 혹은 고객의 흥미와 몰입도를 높이기 위한 장치로 사용되고 있다. 일반적으로 공연예술에서 볼 수 있는 프로젝션 맵핑 기술은 무대의 세트나 연출에 사용되는 미술도구와 같이 공연장의 실내 환경이나 사용되는 오브젝트의 표면을 맵핑하여 시각적인 증강을 표현한다. 최근에는 3D와 4D를 넘어서 360도 돔에까지 스크린으로 활용이 가능할 정도로 기술이 발전된 동시에 VR/AR과 결합되면서 그 활용 가능성이 더욱 커지고 있는 상황이다[8]. 그중 동적 프로젝션 맵핑은 대표적인 인터랙티브 미디어 콘텐츠의 한 유형이다. 무대 위에서 위치가 이동하거나 형태가 바뀌는 등 동적 객체에 프로젝션 기법을 적용하여 다양한 인터랙션 효과를 제공한다. 사물의 움직임에 따라 맵핑되는 영상이 달라지기 때문에 관객들의 몰입도를 높일 수 있다는 점이 동적 프로젝션 맵핑의 강점이다. [표 2]는 동적 프로젝션 맵핑의 사례들이다.

[9]와 같이 대상 오브젝트의 형태는 고정되어 있으나 움직이는 위치나 각도에 따라 모양이 변하는 콘텐츠가 있고, [10]처럼 대상 오브젝트들의 위치 변화나 형태 변화에 따라 프로젝션되는 미디어가 변형되는 콘텐츠가 있다. 기존의 프로젝션 맵핑 퍼포먼스 공연들은 기술적 한계로 인해 이미 만들어진 동영상 콘텐츠에 사람이 합을 맞추어 진행하는 방식으로 공연을 제작하였다. 이러한 제작 방식은 하나의 프로젝션 맵핑 퍼포먼스 공연이 올라갈 때마다 매번 필요한 동영상 작업 기간을 포함하여 공연자의 오랜 연습기간이 필요하다는 한계를 가진다. [12]는 적외선 카메라를 통해 공연자의 몸을 추적하고 실시간으로 공연자의 제스처를 파악해 비디오를 맵핑한다. 깊이 카메라나 센서 등을 사용하여 실시간으로 사람의 위치와 실루엣을 추적하고 그 위에 영상을 맵핑하여 실감도를 높인다. 이렇듯 기술의 발전과 동적 프로젝션 맵핑이 가진 높은 흥미도와 몰입감으로 인해 최근 창작자들이 다양한 방식으로 지속적인 시도를 하고 있지만 구현의 어려움으로 완성하기까지의 장벽은 여전히 높다.

Ⅲ. 제안 인터랙티브 미디어 콘텐츠 제작 프레임워크

1. 제안 프레임워크의 구성

본 논문에서 제안하는 '제스처 인식 기반 인터랙티브 미디어 콘텐츠 제작 프레임워크'는 크게 5가지 모듈로 구성되고, C++과 파이썬(Python) 기반의 텐서플로우(TensorFlow)로 구현된다. 제안 프레임워크를 구성하는 첫 번째 구성 요소는 구성파일(Configuration file)로부터 지정 제스처와 미디어 콘텐츠 효과간의 연결 내용을 읽어 들이고, 실시간으로 입력되는 카메라 영상을 기반으로 모션 히스토리 이미지를 생성하며, 현재 인식된 사용자 제스처 타입에 따른 상태변이(state transition)를 관리하고 이를 미디어 파티클 시스템에 전달하는 메인 제어 클래스(ofApp)이다. 두 번째 구성요소는 ofApp로부터 모션히스토리 이미지를 받아서 텐서플로우의 제스처 추론 모델에 전달하고, 제스처 추론 모델로부터 현재 프레임에 대한 추론된 제스처 타입을 받아서 제스처 큐를 구성하며, 제스처 큐의 구성 내용에 따라 최종적으로 현재의 제스처 타입을 판정하는 클래스(Recognizer)이다. 세 번째 구성 요소는 파티클 시스템 클래스(Particle System)이다. 이 클래스에서는 ofApp로부터 제스처에 따른 현재 상태와 사용자 위치에 동적으로 파티클 효과를 프로젝션 맵핑하기 위한 카메라와 프로젝터간의 공간 맵핑 정보를 전달받고, 적절한 파티클 효과를 렌더링 한다. 네 번째 구성요소는 실시간 제스처 인식 처리를 수행하기 전의 사전 처리 단계로서, 각 제스처별로 수집된 모션 히스토리 이미지 데이터셋을 읽어 들이고 학습을 수행하여, 제스처 인식을 위한 PB(Protobuf) 파일을 생성하는 합성곱 신경망 모델(Gesture Training)이다. 마지막으로 다섯 번째 구성요소는 생성된 PB 파일을 기반으로 하여 실시간으로 모션 히스토리 이미지 입력받아 매칭되는 제스처 타입을 추론하는 텐서플로우 모듈(Gesture Inference)이다. 본 논문에서 제안하는 인터랙티브 미디어콘텐츠 제작 프레임워크의 전체적인 개요도는 [그림 1]과 같다.

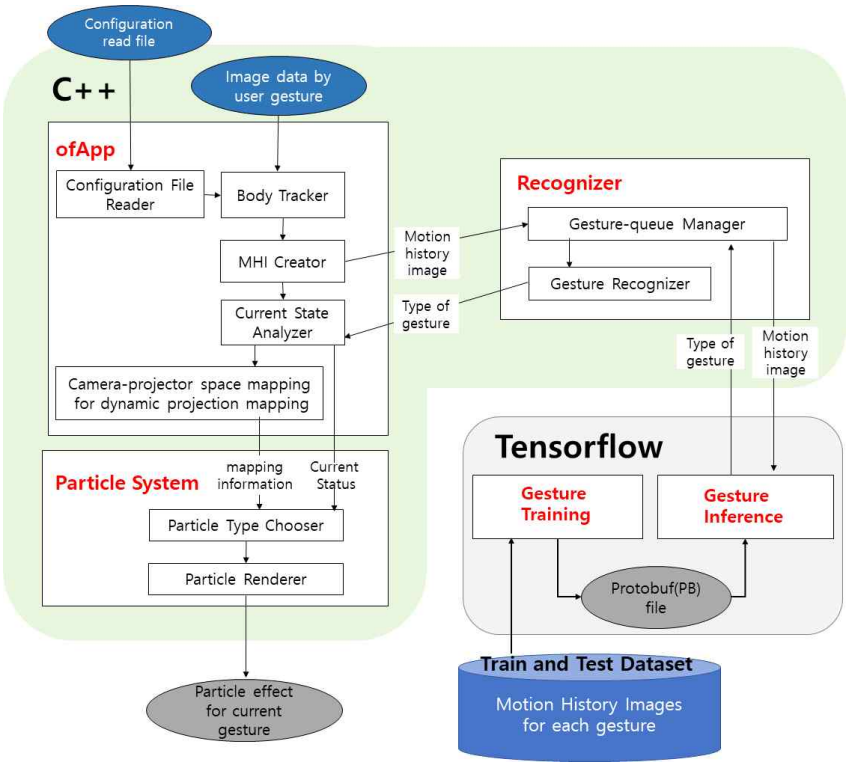


그림 1. 제안하는 인터랙티브 미디어 콘텐츠 제작 프레임워크의 전체 구조도
Fig. 1. Diagram of the proposed interactive media content production framework

현재 프레임 시점을 기준으로 30 프레임 이전부터의 프레임이미지를 처리하여 한 장의 모션 히스토리 이미지를 생성한다. 모션 히스토리 이미지는 각 프레임 영상에서 사용자의 실루엣을 추출하고, 사용자 실루엣 영역은 현재 타임스탬프에 매칭되는 픽셀값으로 채우고, 배경영역은 검은색으로 픽셀값을 채운 결과를 현재 프레임까지 완성된 모션 히스토리 이미지에 누적시켜 나가는 방식으로 생성된다. 한 장으로 표현된 모션 히스토리를 기반으로 합성곱 신경망을 통하여 실행하는 제스처 인식은 100%의 인식률을 보이지 못한다. 실시간 입력되는 영상을 기반으로 안정적 제스처 인식을 수행하기 위하여 텐서플로우 추론 모델을 통해 얻은 결과 값을 바로 현재의 제스처 타입으로 판정하지 않고, 제스처 큐(queue)를 구성하여, 프레임별로 노이즈(에러상황)로 인식된 상황을 걸러낸다. 제스처 큐는 세션에서 결과 값을 리턴 할 때마다 반복 수행한다. 아래 알고리즘 1은 제스처 큐를 이용하여 노이즈를 제거하고 최종적으로 제스처 타입을 결정하는 알고리즘이다.

Algorithm 1 : Gesture Type Decision
Input : returned value from tensorflow module
Output : gesture type
Begin
```
1 : queue.push(returned value from tensorflow module);
2 : if queue.size() == 4
3 :     int arr[the total number of gesture type] = {};
4 :     for (int i = 0; i <queue.size(); i++)
5 :         arr[queue[i]]++;
6 :     for (i∈T, where T = Target gesture's index)
7 :         if arr[i]>=queue.size() − 1
8 :             queue.pop();
9 :             return i;       // return Gesture Type i
10:     queue.pop();
11:     return 999;
12: else return 999;
```
End

먼저 1행 에서는 텐서플로우 모듈을 통해 리턴 받은 제스

처 타입을 계속 제스처 큐에 푸쉬한다. 2행과 11행을 보면 queue의 사이즈에 따라 리턴 값이 달라지는데, 제스처 큐의 사이즈가 4보다 작으면 제스처와 상관없는 값을 리턴하여 제스처 타입을 제공하지 않고 제스처 큐의 사이즈가 4가 될 때부터 그중 가장 많이 리턴된 제스처 타입을 찾는 과정을 수행한다. 타겟 제스처 타입의 개수와 크기가 같은 int 배열 arr을 선언하고, 값을 모두 0으로 초기화 한다. 4, 5행에서는 제스처 큐에 들어가 있는 제스처 타입별 개수를 카운트한다. 6~9행의 for문에서 사용되는 변수 i는 타겟 제스처의 인덱스 값이다. 7행은 arr[i]의 값이 제스처 큐의 과반수를 차지하게 되는 경우를 의미한다. 8 ~ 9행에서는 제스처 큐의 원소 하나를 팝(pop)하고 제스처 큐에서 과반이 넘은 제스처 타입 i를 리턴한다. 9행에서는 제스처 큐의 원소를 하나 팝처리 한다. 10행까지 진행이 되었다는 것은 타겟으로 설정된 제스처 타입이 제스처 큐 내에서 과반수를 차지하지 못했다는 의미로, 제스처 큐의 원소 하나를 팝처리하고 '현재 프레임에서 제스처를 결정하지 못했다'라는 의미의 값을 리턴한다.

2. 구성파일(Configuration File)을 통한 동적 효과 정의

각 동작과 매칭되는 동적 파티클 효과는 openFramework를 기반으로 제작되었다. openFrameworks은 C++ 툴킷으로 직관적인 프레임워크를 통해 콘텐츠 제작을 돕는 오픈소스이다. 텍스트로 구성되는 구성파일(Configuration File)에 제스처와 파티클 효과를 연결하는 인터랙티브 미디어 콘텐츠를 정의한다. [표 3]은 구성파일에서 총 제스처의 개수와 파티클 효과의 개수, 판별된 제스처의 결과값을 저장하는 디렉토리 등 각각의 제스처와 그에 맞는 파티클 효과 매칭을 정의하는 규칙을 보여준다. 제스처와 파티클 효과 매칭 구성파일은 [그림 1]의 전체구조도에서 보이는 바와 같이 C++메인 모듈의 configuration file reader를 통하여 읽어 들이고, 구성파일의 내용대로 제스처와 파티클 효과 자료구조를 구성한다.

3. 제스처 인식 모듈 구현 및 실험

제안 도구는 3개의 세부 프로그램 즉, 제스처 데이터 생성 및 저장 프로그램(C++), 제스처 학습 프로그램(Tensor- Flow), 실시간 카메라 영상 기반 제스처 추론 및 동적 프로젝션 통합 프로그램(C++/TensorFlow)으로 구성된다. 제스처 인식을 위한 제스처 데이터 수집 및 학습의 절차는 [그림 2]와 같다. 제스처 데이터 생성 및 저장 프로그램에서는 카메라를 통해 수집한 제스처 영상 이미지를 메인 클래스 모듈에서 모션 히스토리 이미지로 변환하여

표 3. 제스처와 파티클 효과 매칭을 위한 구성파일 규칙 예
Table 3. An example of the configuration file for five gestures and four media effects

Each gesture and media effect name	Description
Num_Gesture: 5	Number of target gestures
GestureDir:"gesture directory name"	The name of the directory which stores the tensorflow modules for gesture recognition
Num_Effect: 4	Number of Particle effects
Effect1:"ParticleEffect1"	Particle effect ID
Effect2:"ParticleEffect2"	Particle effect ID
Effect3:"ParticleEffect3"	Particle effect ID
Effect4:"ParticleEffect4"	Particle effect ID
PerformerComeOut: 1	If it is determined that there is a performer (= user), effect 1 is executed.
Gesture1: 2	Gesture1 and 2nd effect matching
Gesture2: 3	Gesture2 and 3rd effect matching
Gesture3: 3	Gesture3 and 3rd effect matching
Gesture4: 4	Gesture4 and 4th effect matching
Gesture5: 4	Gesture5 and 4th effect matching

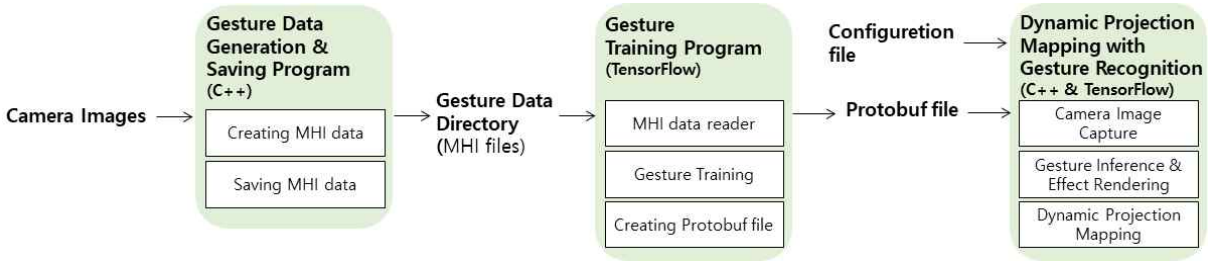


그림 2. 제스처 데이터 생성 및 학습 절차에 관한 다이어그램
Fig. 2. Diagram of gesture image data training step

저장한다. 제스처 데이터 생성 및 저장 프로그램의 인터페이스는 [표 4]와 [그림 4]를 통해 구체적으로 설명한다. 제스처 학습 프로그램에서는 저장된 모션 히스토리 이미지를 읽고 합성곱 생성망을 통하여 이를 학습하고, 학습 결과를 담은 Protobuf 파일을 생성한다. Protobuf 에 저장된 제스처 학습결과에 따른 추론과 이에 따른 파티클 효과의 표현은 동적 프로젝션 매핑 통합 프로그램에서 수행된다.

3.1 모션 히스토리 이미지 생성 및 저장

본 논문에서는 입력데이터로 모션 히스토리 이미지를 사용했다. 모션 히스토리 이미지는 움직임이나 제스처의 진행 경로를 실루엣 이미지로 한눈에 확인할 수 있는 정적 이미지 템플릿이다. 일반적으로 제스처 인식을 위해서는 연속 프레임 이미지들을 사용하기 때문에, 2D 이미지를 이용한 객체판별(object classification)의 경우보다 사용되는 메모리 크기가 커지고 계산량도 증가된다. [그림 3]은 본 논문에서 학습 실험을 진행한 '발차기' 동작의 모션 히스토리 이미지를 순차적으로 나열한 것이다. 각 단계에서는 프레임별로 사용자의 실루엣을 추출하고 사용자 실루엣 이미지는 30개 프레임의 사용자 실루엣 정보를 저장하는 원형큐(circular queue)에 저장한 뒤, 실루엣 원형큐의 과거 시점부터 현재 시점까지 실루엣 이미지를 순차적으로 받아 들여 실루엣 영역의 픽셀 값을 읽고, 이를 모션 히스토리 이미지 공간에서 매칭되는 픽셀의 픽셀 값으로 대체한다. 사용자 실루엣을 이용하여 표현된 모션 시퀀스는 동작정보를 저용량의 그레이 스케일 이미지로 압축하여 표현하게 한다.

제스처 데이터 생성 즉, 모션 히스토리 이미지 생성 프로그램은 [표 4]와 같이 두 단계의 절차를 거쳐 실행된다. 사용자는 키보드를 클릭만을 활용하여 쉽게 제스처 이미지 데이터를 수집 저장한다. 수집된 이미지 파일들은 [그림 4]와 같이 지정된 폴더에 저장되고 추후 제스처 학습 모듈로 전달되어 데이터로 사용된다.

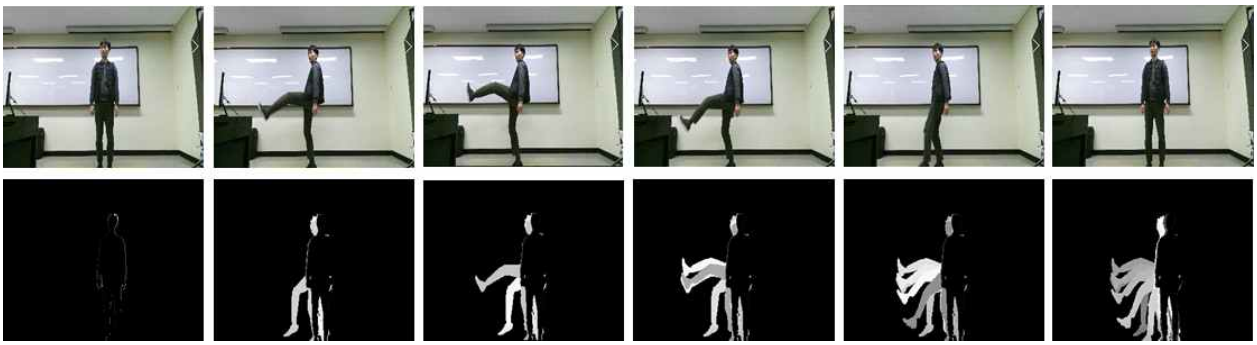
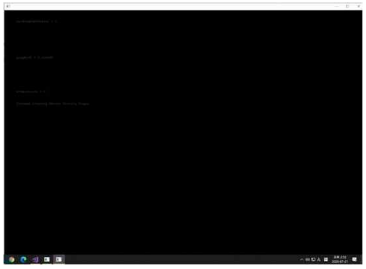
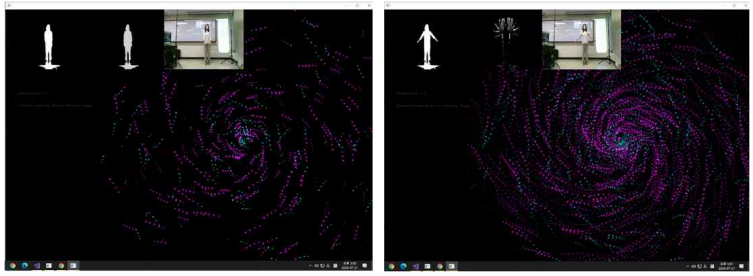


그림 3. '발차기-왼쪽' 동작에 대한 모션 히스토리 이미지의 나열
Fig. 3. a series of motion history images for 'kicking-left'

표 4. 모션 히스토리 이미지의 저장 프로그램
Table 4. Motion history image generation program

Creating and saving of motion history images	
1	2
	
Start the gesture data generation / saving program and press 'p(play mode)' on keyboard	Press the key 'c(capture mode)' on keyboard for saving a series of color images and motion history images

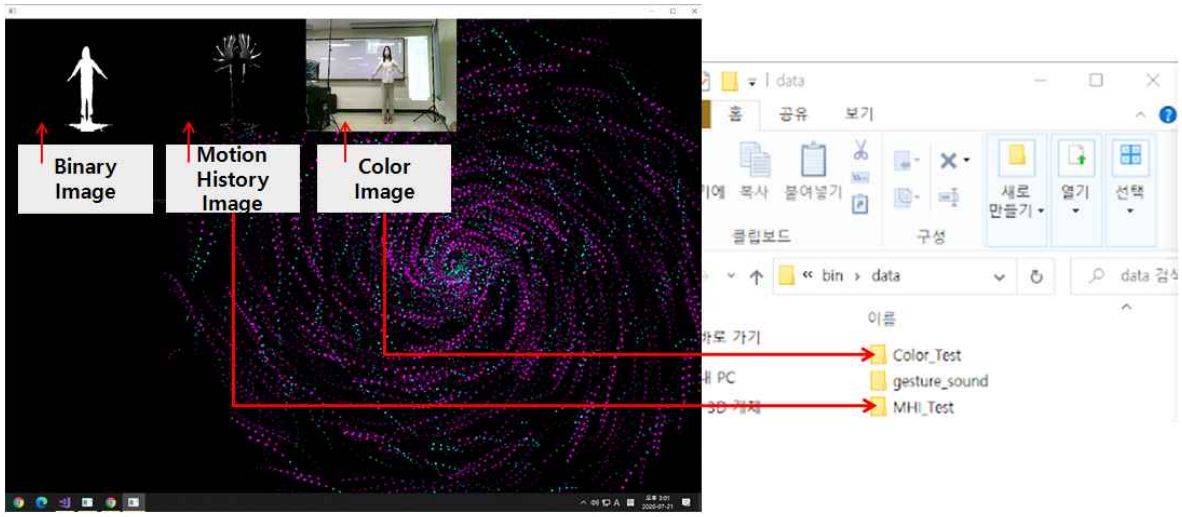


그림 4. 제스처 데이터 수집 화면
Fig. 4. Interface for collecting gesture data

3.2 제스처 인식을 위한 합성곱 신경망 설계

본 논문에서는 파이썬 기반의 텐서플로우로 구현한 제스처 학습 모듈을 통해 제스처 인식 학습을 진행하였다. 실험에 쓰인 합성곱 신경망은 [그림 5]의 3계층의 합성곱 레이어와 2계층의 전연결 (Fully-connected) 레이어로 구성되는 모델과 [그림 6]의 5계층의 합성곱 레이어와 1계층의 전연결 (Fully-connected) 레이어로 구성되는 모델을 구성하여 실험하였다. 3계층의 합성공 신경망에서는 합성곱 레이어의 필터수를 'X'라 표기하고, 각 레이어에서 모두 32, 16, 8 개의 필터를 적용하는 3가지 버전으로 학습을 진행하였다. 그리고 모든 레이어에서 5x5 필터를 적용하였다.

5계층 합성곱 신경망의 경우, 모든 합성곱 레이어에 스트라이드(stride)의 크기를 2x2로 적용하고 각 계층의 필터의 수를 차례로 16, 32, 64, 128, 256으로 설정하여 학습하였다. [그림 6]은 제안 플랫폼에 적용한 5계층 합성곱 신경망 구조를 보여주고 있다.

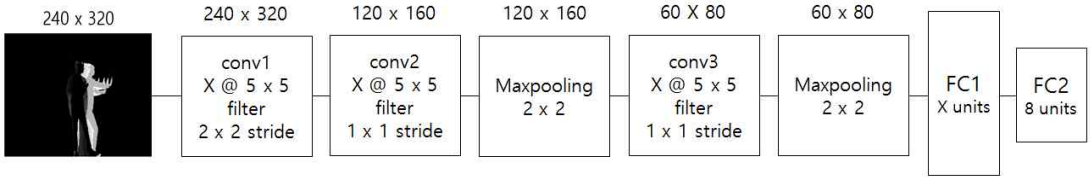


그림 5. 2계층의 전연결 레이어를 포함한 3계층 합성곱 신경망
Fig. 5. Three-layered CNN with 2 fully connected layers

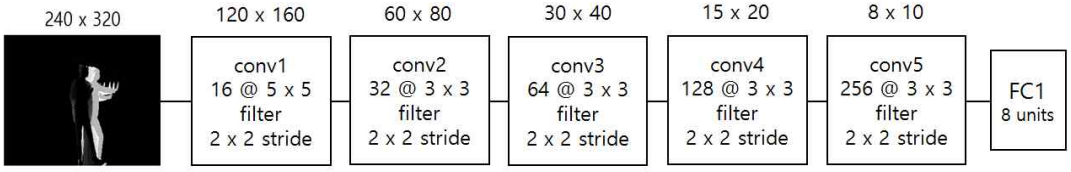


그림 6. 1계층의 전연결 레이어를 포함한 5계층 합성곱 신경망
Fig. 6. Five-layered CNN with 1 fully connected layer

3.3 C++ 모듈과 텐서플로우의 연결

본 논문에서 제안하는 “제스처 인식 기반 인터랙티브 미디어 콘텐츠 제작 프레임워크”는 카메라로부터 영상을 입력 받아 모션 히스토리 이미지를 생성하는 모듈과 미디어 이펙트를 표현하여 렌더링하는 인터페이스 전반적인 부분은 C++ 구현되었고, 모션 히스토리 이미지 한 장에 따른 적합한 제스처 타입 인식 모듈은 텐서플로우로 구현되었다. C++모듈과 텐서플로우 모듈을 연결하기 위하여 ‘boost 파이썬’ 라이브러리를 사용하였다. ‘boost 파이썬’ 라이브러리를 통해 C++에서 pb파일로 된 제스처 학습 결과를 노드 그래프(node graph)로 가져와 연결하는 방식으로 진행하였다. Boost는 1.69ver, Conda 4.6.15ver, Python 3.7.3ver 그리고 openFramework는 0.9.8ver으로 설정하여 진행하였다.

Ⅳ. 제안 프레임워크 적용 실험

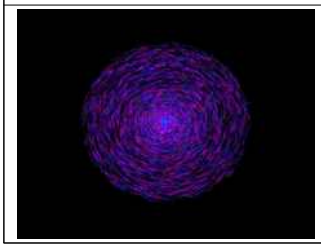
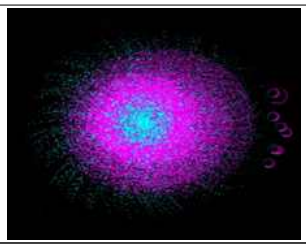
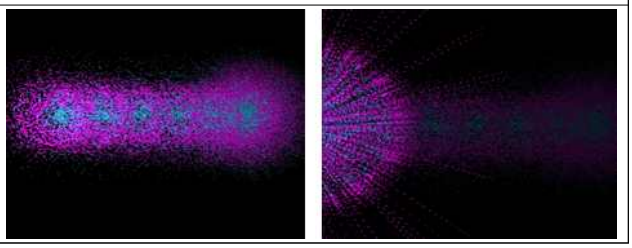
1. 목표 제스처 및 제스처 별 타겟 효과

목표 제스처들은 본 논문에서 구상하는 콘텐츠의 시나리오에 요구되는 제스처들로 설정하였다. 첫 번째는 기를 모으는 제스처, 두 번째는 모아진 기를 손으로 발사하기 위한 장풍쏘기 제스처, 마지막으로 세 번째는 모아진 기를 발을 이용해 발사하는 발차기 제스처이다. 장풍 쏘기와 발차기

표 5. 목표 제스처 별 모션 히스토리 이미지 예
Table 5. An example of motion history image for each target gesture

gesture name 동작 명	Power Charging 기 모으기	Shooting left 장풍 쏘기 - 왼쪽	Shooting right 장풍 쏘기 - 오른쪽	Kicking left 발차기 - 왼쪽	Kicking right 발차기 - 오른쪽
motion history image of gestures 제스처의 모션 히스토리 이미지					
Description	Put your arms up and slowly lower them in a circular motion.	Stretch your arms to the left as if pushing something out of your palm.	Stretch your arms to the right as if pushing something out of your palm.	Stretch your feet to the left.	Stretch your feet to the right.

표 6. 목표 제스처에 따른 파티클 효과
Table 6. Particle effect of target gestures

User detection 사용자 인식	Effect of 'Power Charging' 기 모으기 효과	Two-step effects of 'Shooting' and 'Kicking' 장풍 쏘기 및 발차기에 따른 2단계 효과
		

제스처의 경우, 사용자에 따라 움직이는 방향이 다르다는 점을 고려하여 왼쪽과 오른쪽을 구분하여 총 5가지 제스처의 이미지 데이터를 수집하였다. 아래의 [표 5]는 본 논문에서 인식하고자 하는 목표 제스처의 동작 흐름을 모션 히스토리 이미지로 표현하고 각 제스처에 대한 설명을 보여준다. [표 6]은 각 목표 제스처에 따른 인터랙션 파티클 효과이다. 파티클 효과(particle effect)는 openFrameworks를 기반으로 제작되었다.

사용자가 지정된 위치가 서면 카메라가 사람을 인식하여 사용자의 주변에 [표 6]의 사용자 인식에 해당하는 파티클 효과를 뿌려준다. '기 모으기' 제스처를 시작하면 주변에 발산하던 빛들이 사용자의 몸통 중심으로 모여 뭉쳐지는 효과가 보여진다. '장풍 쏘기'는 공연자의 두 손 중심에서 손을 뻗는 방향으로 파티클 덩어리가 발사되어 실제로 사용자의 손에서 빔이 발사되는 것과 같은 효과를 준다. '발차기' 역시 동작 마지막 시점에 발끝이 향하는 방향으로 불꽃 형태의 파티클이 발사되어 터지는 효과이다. 실제로 발끝에서 파티클이 날아가서 영상의 끝에서 파티클들이 빠르게 분산되는 직관적 효과를 주어 관람객들의 빠른 이해를 돕는다.

2. 학습 및 테스트 데이터셋 구축

우리는 5계층 합성곱 신경망에 자체적으로 제작한 데이터셋을 활용하여 모션 히스토리 이미지 기반의 제스처 인식 실험을 진행하였다. 수집된 모션 히스토리 이미지는 Kinect의 컬러 카메라로부터 연속 영상을 받아, 3.1절에서 설명한 방식으로 모션 히스토리 이미지를 생성하여 320x240 픽셀 크기의 그레이 스케일 이미지로 저장하였다. 총 7명의 참가자를 대상으로 각 제스처에 대한 이미지 데이터를 확보하였다. 5가지의 목표 제스처와 3가지의 오류 동작들로 나누어 총 8가지의 제스처 이미지 데이터를 수집하였다. 아래의 [표 7]은 본 논문의 실험에 사용된 데이터셋의 구성이다. 학습 데이터 세트와 테스트 데이터 세트의 비율은 약 7:3으로 구성하였다.

표 7. 학습 및 테스트 데이터셋을 구성하는 모션 히스토리 이미지 수
Table 7. Number of motion history images that make up the training and test dataset

	Training	Test
Power Charging	611	261
Shooting-Left	613	333
Shooting-Right	613	333
Kicking-Left	555	253
Kicking-Right	554	246
Error 1(black)	241	102
Error 2(etc)	318	136
Error 3(ready)	139	59
총 개수	3644	1723

5가지의 제스처 이미지 데이터 외의 오류 제스처들은 애플리케이션 체험 시 다양하게 나타나는 사용자들의 타겟 제스처간 중간동작(오류동작)들에 관한 데이터이다. 아래의 [표 8]은 3종류로 분류하여 학습시킨 오류 동작들의 모션 히스토리 이미지 예이다.

표 8. 오류로 분류하여 학습시킨 모션 히스토리 이미지 데이터
Table 8. Motion history images of 'Error' gestures

Error 1 - Black			
Error 2 - Etc			
Error 3 - ready			

첫 번째 오류 제스처는 'Black' 군으로 분류된 이미지들로서 모션 히스토리가 거의 잡히지 않은 검정 화면들을 수집하였다. 이는 아직 사용자가 등장하지 않았음에도 반응할 수 있는 가능성을 배제하기 위해 추가하였다. 두 번째 오류 제스처는 'etc' 군으로 분류된 이미지들로서 목표 제스처와 관계없이 사용자가 카메라 앞으로 이동 중에 대부분의 이미지 영역이 사용자 영역으로 인식된 경우의 이미지들을 담았다. 가장 범위가 큰 분류로써 사용자가 본 논문에서 제안하는 플랫폼을 사용할 때 자연스럽게 발생하는 대부분의 무의미한 제스처들이 포함된다. 마지막으로 세 번째 오류 제스처는 'ready' 군으로 분류된 이미지로서, 목표 제스처와 관계없는 작은 동작들을 모은 것으로 주로 목표하는 제스처를 수행하기 이전에 나오는 미세한 제스처들에 해당된다.

3. 실험 환경

실험에서는 우선, 3장에서 설계한 합성곱 신경망에 대한 제스처 인식률과 실행시간을 측정하였다. 그리고, 타겟 제스처와 파티클 효과 연결에 따른 가시화 결과를 관찰하였다. 실험환경으로 실험자와 카메라간의 거리는 2.25m이고 카메라와 프로젝터의 거리는 1.35m로 고정하여 실험을 진행하였다. 카메라의 높이는 0.8m로 고정하였다. 실험에 참여하는 사용자는 카메라를 정면으로 마주보는 위치에서 실험을 진행한다. 제안 플랫폼을 통한 제스처 인식 및 효과 확인 실험에는 깊이 카메라인 Kinect V2와 해상도 5000안시의 고광량 빔프로젝터를 사용하였다. 제스처 학습과 추론, 파티클 효과 가시화를 위한 제안 프레임워크 구현 및 실행은 모두 Windows 10 운영체재, CPU RAM 64.0GB 그리고 NVIDIA 11GByte 메모리를 가지는 GeForce GTX 1080 Ti의 GPU 환경에서 진행되었다. 파이썬은 3.7 버전, 텐서플로우는 1.14.0 버전을 사용하였다.

4. 합성곱 신경망 구조별 실험결과

[표 9]는 3계층 합성곱 신경망을 사용하여 진행한 실험에서 테스트 데이터셋에 대한 제스처 인식률과 실행시간을 보여주고 있다. 실행시간은 하나의 모션 히스토리 이미지를 입력으로 제스처 타입을 추론하는 데 걸리는 시간을 초 단위로 표현한 것이다. 초기 학습률(learning rate)는 0.01로 하고, 배치 사이즈는 30, 300 에폭을 적용하였고, 250 에폭부터 학습률 디케이(decay)를 적용하여 학습하였다. 실험결과, 각 레이어별 8개의 필터를 사용하였을 때, 테스트 데이터에 대한 높은 제스처 인식률을 얻었지만 학습데이타를

표 9. [그림 5]에서 제시된 3계층 합성곱 신경망 적용 시 제스처 인식률과 실행시간
Table 9. Gesture recognition accuracy and execution time in the 3-layered convolutional neural network of Fig. 5

learning rate : 0.001, batch size : 30, number of epochs : 300			Execution time per image (sec.)
Application of learning rate decay	Number of filters(X)	accuracy	
Applied from epoch 250	32	0.9473	0.2535
	16	0.9649	0.0780
	8	0.9824	0.0219

표 10. [그림 6]에서 제시된 5 계층 합성곱 신경망의 인식률과 런타임 결과
Table 10. Gesture recognition accuracy and execution time in the 5-layered convolutional neural network of Fig. 6

learning rate : 0.0001, number of epochs : 200								
condition		conv_1 16@	conv_2 32@	conv_3 64@	conv_4 128@	conv_5 256@	Accuracy	Execution time per image (sec.)
con.1	filter size	5 x 5	5 x 5	5 x 5	3 x 3	3 x 3	0.9649	0.5450
	Stride size	2 x 2	2 x 2	2 x 2	1 x 1	1 x 1		
	learning rate	0.0001						
	batch size	30						
con.2	filter size	5 x 5	3 x 3	3 x 3	3 x 3	3 x 3	0.9796	0.0838
	Stride size	2 x 2	2 x 2	2 x 2	2 x 2	2 x 2		
	learning rate	0.0002						
	batch size	5						

그림 7. [그림 5]의 3계층 합성곱 신경망에서 모든 합성곱 레이어에 8개의 필터를 적용한 경우, 학습 데이터에 대한 인식률
Fig. 7. Gesture recognition accuracy in the 3-layered CNN of Fig. 4 with 8 filters applied to each layer using the training data

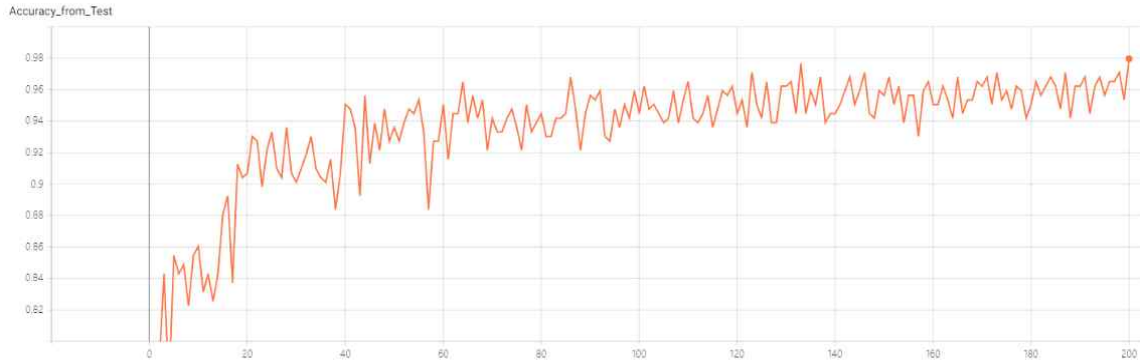


그림 8. [그림 6]의 5계층 합성곱 신경망에서 [표 10]의 조건 2를 적용한 경우, 학습 데이터에 대한 인식률
Fig. 8. Gesture recognition accuracy in the 5-layered CNN of Fig. 6 with condition 2 of Table 10 using the training data

표 11. 각 제스처 별 인식률
Table 11. Accuracy of each gesture recognition

	Gesture					
	meangless motion	gauge	kick-L	kick-R	wind-L	wind-R
false/total images	3/297	7/261	11/253	14/246	15/333	19/333
accuracy	98.98	97.31	95.65	94.30	95.49	94.29

이용한 인식률의 편차가 [그림 7]과 같이 높게 나타났다.

[표 10]은 [그림 6]에서 제시된 5계층 합성곱 신경망에서 스트라이드 크기(stride size)와 적용 필터 크기, 학습률, 배치 사이즈 등의 조건을 변경하여 학습을 진행한 후, 테스트 데이터를 적용하여 측정한 결과이다.

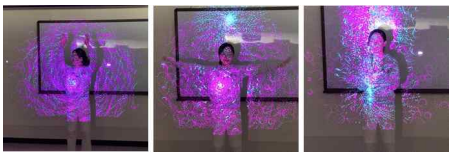
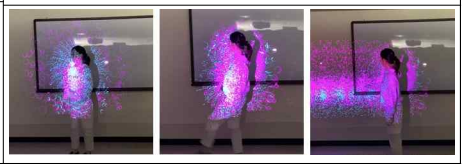
실험 결과, 4번째와 5번째 합성곱 레이어에서 스트라이드 크기를 1x1에서 2x2로 변경했을 시 실행시간이 눈에 띄게 줄어든 것을 확인할 수 있었다. [표 10]의 조건 2과 같이 각 합성곱 레이어에 적용하는 필터의 크기를 변경하여 학습시키고, 테스트 데이터를 적용하였을 때, 제스처 인식률이 향상되고, 실행 속도를 현저히 줄일 수 있었다. 또한, [그림 8]과 같이 학습 데이터 적용 시 학습률의 편차가 높지 않게 안정적으로 학습이 진행되었다.

본 논문에서는 높은 인식률을 보이면서 인식률의 편차가 작고 실행속도가 빠른 [그림 6]의 5계층 합성곱 신경망에 [표 10] 조건 2의 하이퍼파라미터를 적용하여 제스처를 학습시켰다. [표 11]은 학습된 제스처 별 인식률을 나타낸 표이다. 학습된 결과를 Protobuf 파일로 저장하여, 이를 실시간 제스처 타입 추론에 사용하였다.

5. 제스처에 따른 파티클 효과 적용 실험 결과

[표 12]는 제안 인터랙티브 미디어 콘텐츠 프레임워크를 통하여 구현한 콘텐츠 구동 결과를 보여 주고 있다. 아무도 없는 공간에 사람이 들어간 직후 카메라가 사용자 인식에 성공하면 [표 6]의 '사용자 인식'에 해당하는 효과가 사용자를 중심으로 한 공간에 프로젝션 된다. 이후 기 모으기와 장풍 쏘기 및 발차기 동작을 취했을 경우, 아래와 같이 각 제스처 번호에 해당되는 파티클 효과가 사용자의 몸 위로 프로젝션 된다.

표 12. 제스처에 따른 파티클 효과 적용 화면
Table 12. Media effect mapped screen by gesture

Check the user 사용자 인식	
Power Charging 기 모으기	
Shooting-Left 장풍 쏘기 - 왼쪽	
Shooting-Right 장풍 쏘기 - 오른쪽	
Kicking-Left 발차기 - 왼쪽	
Kicking-Right 발차기 - 오른쪽	

V. 결 론

본 논문에서는 제스처 인식을 위해 적용하고 있는 방법

들을 간략히 설명하고 이를 활용한 여러 인터랙티브 미디어 콘텐츠 중 관람객들로 하여금 높은 몰입도를 불러일으키는 제스처 에 반응하는 미디어 콘텐츠 특히 동적 프로젝션 맵핑 콘텐츠 사례를 제시하였다. 본 제작 도구는 다수의 제스처와 파티클 효과들을 입력 받아 정의할 수 있고, 정의된 제스처를 기준으로 파티클 효과를 매칭시켜 사용할 수 있다. 추후 제스처와 미디어 파일과의 매칭 구조도 지원할 예정이다. 기술적 한계로 인해 인터랙티브 미디어 콘텐츠 제작에 어려움을 느끼는 창작자는 이를 기반으로 제스처 인식 기반의 간단한 인터랙티브 미디어 콘텐츠 제작을 경험하고 활용할 수 있다. 또한, 서로 다른 구조의 합성공 신경망에 대한 제스처 인식률과 추론 시간 측정 실험을 수행하여, 97% 이상의 인식률을 가지며 실시간으로 제스처 인식을 할 수 있는 합성곱 신경망 구조를 제시하고, 이를 제안 프레임워크에 적용하였다. 그 결과 사용자의 제스처에 따라 실시간으로 알맞은 파티클 효과가 맵핑됨을 확인할 수 있었다.

제안 도구는 제스처 데이터 생성 및 저장 프로그램(C++), 제스처 학습 프로그램(TensorFlow), 실시간 카메라 영상 기반 제스처 추론 및 동적 프로젝션 통합 프로그램(C++/TensorFlow)으로 구성되어 단계별로 각 프로그램을 수행하도록 하였다. 향후 연구로서, 사용자가 보다 편하게 사용할 수 있도록 3가지 프로그램을 통합하는 인터페이스 제작을 진행하고, 이를 이용한 사용자 테스트를 수행할 예정이다. 또한, 여러 명의 사용자에 대한 제스처 인식 기능을 추가하여, 사용자별 의미 있는 제스처 정의와 이에 따른 다중 사용자간 미디어 효과를 표현하도록 함으로써 보다 무대 공연에 적합한 사용하기 용이한 프레임워크로 확장하고자 한다.

참 고 문 헌 (References)

[1] J. Park, W. Kim, “A Study of the Value and Utility of Projection Mapping in the Contents of Dance Performances-Focused on the Work :“Our Karma”-”, Journal of Korean Dance, Vol.46, pp.9-28, August 2018.

[2] Projection Mapping Contents of Media Art – LG CNS Blog, https://blog.lgcns.com/1149 (accessed July. 14, 2016)

[3] S. Kim, Y. Koh and Y. Choi, “Design and Implementation of Immersive Media System Based on Dynamic Projection Mapping and Gesture Recognition” KIPS Transactions on Software and Data Engineering, Vol.9, No.3, pp.109-122, January 2020. https://doi.org/10.3745/KTSDE.2020.9.3.109

[4] H. Yang, “Deep-learning and Gesture recognition” Broadcasting and Media Magazine, Vol. 22, No.1, pp.67-74, January 2017.

[5] B. Lee, D. Oh, T. Kim, “3D Virtual Reality Game with Deep Learning-based Hand Gesture Recognition.” Journal of the Korea Computer Graphics Society, Vol. 24, No. 5, pp.41-48, December 2018. https://doi.org/10.15701/kcgs.2018.24.5.41

[6] Maryam Asadi-Aghbolaghi, Albert Clapes, Marco Bellantonio, “A survey on deep learning based approaches for action and gesture recognition in image sequences.” Proceeding of IEEE International Conference on Automatic Face & Gesture Recognition, Washington, DC, pp. 476-483, 2017, https://doi.org/10.1109/FG.2017.150 (accessed June.1, 2017)

[7] J. Lee, Y. Kim, D. Kim, “Realtime Projection Mapping on Flexible Dynamic Objects” Proceeding Of HCI Korea Conference, Gangwon-do, Korea, pp. 187-190, 2014.

[8] KOCCA, Trends in stage production of large events and performances in the United States, Content Industry Trend of USA, Vol. 21, 2018

[9] bot&dolly – Box, https://youtu.be/lX6JcybgDFo (accessed Sep. 24, 2013)

[10] Ishikawa group Lab - DynaFlash v2 and Post Reality, https://youtu.be/QDppJ9NWtaE (accessed Mar. 5, 2018)

[11] connected colors / real-time face tracking and 3d projection mapping, https://youtu.be/nMvFwC3bo_E (accessed Mar. 14, 2016)

[12] 縁 ' NMARA Interdisciplinary Art Performance Works, https://youtu.be/0oa7kVbVxsA (accessed Dec. 16, 2019)

[13] Virtual Reality Interactive Sandbox (Contour Line), https://youtu.be/PyJfIdNbtv4 (accessed Feb. 4, 2015)

[14] Interactive wall and Floor Projection, https://youtu.be/AZA6X3mPdtg (accessed May. 7, 2013)

저 자 소 개

고 유 진

- 2017년 : 경희대학교 일본어학과(학사)
- 2018년 ~ 현재 : 서울미디어대학원대학교 미디어공학(석사)
- ORCID : https://orcid.org/0000-0001-7992-9960
- 주관심분야 : 증강현실, 가상현실, 프로젝션 맵핑, HCI

김 태 원

- 2013년 ~ 2018년 : 남서울대학교 소프트웨어학과(학사)
- 2017년 ~ 2018년 : ㈜듀코젠(연구원)
- 2019년 ~ 현재 : 서울미디어대학원대학교 인공지능 응용소프트웨어학과 초실감미디어전공(석사과정)
- ORCID : https://orcid.org/0000-0002-7739-252X
- 주관심분야 : 컴퓨터 그래픽스, 컴퓨터비전, HCI, 증강현실, 가상현실

김 용 구

- 1993년 : 연세대학교 전기공학과 공학사
- 1995년 : 연세대학교 전기및컴퓨터공학과 공학석사
- 2001년 : 연세대학교 전기전자공학과 공학박사
- 2002년 ~ 2006년 : ㈜온타임텍 멀티미디어연구소 연구소장/이사
- 2009년 ~ 현재 : 서울미디어대학원대학교 인공지능 응용소프트웨어학과 교수
- ORCID : http://orcid.org/0000-0002-8905-1984
- 주관심분야 : 딥-러닝 기반 미디어 콤퓨팅, 초실감 미디어 제작 및 응용 시스템, 방송 시스템

최 유 주

- 1989년 : 이화여자대학교 전자계산학과(이학사)
- 1991년 : 이화여자대학교 전자계산학과(이학석사)
- 2005년 : 이화여자대학교 컴퓨터공학과(공학박사)
- 1991년 : (주)한국컴퓨터 기술연구소 주임연구원
- 1994년 : (주)포스데이타 기술연구소 주임연구원
- 2005년 : 서울벤처정보대학원 컴퓨터응용기술학과 조교수
- 2010년 ~ 현재 : 서울미디어대학원대학교 인공지능 응용소프트웨어학과 부교수
- ORCID : https://orcid.org/0000-0001-7520-097X
- 주관심분야 : 컴퓨터 그래픽스, 컴퓨터 비전, HCI, 증강현실, 가상현실