Journal of Korea Multimedia Society Vol. 19, No. 8, August 2016(pp. 1479-1486)
http://dx.doi.org/10.9717/kmms.2016.19.8.1479

멀티 디스플레이 콘텐츠 전송 시스템을 위한 디바이스 연결 및 배치 인식 기법의 구현

전소연†, 임순범††

An Implementation of Device Connection and Layout Recognition Techniques for the Multi-Display Contents Delivery System

So-yeon Jeon†, Soon-Bum Lim††

ABSTRACT

According to the advancement of display devices, the multi-screen contents display environment is growing to be accepted for the display exhibition area. The objectives of this research are to find communications technology and to design an editor interface of contents delivery system for the larger and adaptive multi-display workspaces. The proposed system can find existence of display devices and get information without any additional tools like marker, and can recognize device layout with only web-cam and image processing technology. The multi-display contents delivery system is composed of devices with three roles; display device, editor device, and fixed server. The editor device which has the role of main control uses UPnP technology to find existence and receive information of display devices. extract appointed color in captured picture using a tracking library to recognize the physical layout of display devices. After the device information and physical layout of display devices are connected, the content delivery system allows the display contents to be sent to the corresponding display devices through WebSocket technology. Also the experimental results show the possibility of our device connection and layout recognition techniques can be utilized for the large spaced and adaptive multi-display applications.

Key words: Multi-Display Contents, Device Layout Recognition, Web-related Technology

1. 서 론

디바이스의 종류와 사양이 다양해지면서 디바이스는 개인의 소유물로서의 역할과 더불어 예술, 문화 등의 분야에서 활용 가능한 디스플레이로서의 역할을 수행할 수 있게 되었다[1,2]. 이에 따라 멀티 디스플레이 환경이 소수의 대형 디스플레이 환경에서 멀티 디바이스, 멀티 디스플레이 환경으로 발전하게 되었다. 그러나 멀티 디스플레이 환경에 대한 기존의 연구들은 개인을 위한 소규모 작업 공간을 조성하는 것에 집중하고 있어 보다 넓은 규모에서의 멀티 디스플레이 환경 조성이 어렵다. 따라서 전시나 광고를 위한 넓은 환경에서의 멀티 디스플레이 환경 조성에 있어 디바이스의 변경이나 이동 등 다양한 환경적 변화에 맞춘 콘텐츠 배치의 수정이 불편하다는 문제점이 존재한다.

여러 대의 디바이스를 연동하여 멀티 디스플레이

※ Corresponding Author: Soon-Bum Lim, Address: Sookmyung Women's University, Cheongpa-ro 47-gil 100, Yongsan-gu, Seoul, 04310, Korea, TEL: +82-2-710-9424, E-mail: sblim@sm.ac.kr
Receipt date: Feb. 5, 2016, Revision date: May 9, 2016
Approval date: May 31, 2016

† Dept. of Multimedia Science, Sookmyung Women's University
(E-mail: jsoyean92@gmail.com)
†† Dept. of Multimedia Science, Sookmyung Women's University

환경을 조성하기 위해서는 디바이스 간에 배치 상태 인식을 통한 적응적 연결을 해야 한다. 이러한 연결을 위한 디바이스 발견 및 연결 기술로는 AR마커(Augmented Reality Marker)와 같은 매개체를 통한 방법과 네트워크를 이용하는 방법이 있다. AR마커를 이용하는 방식은 AR 저작도구를 이용해 패턴에 의미를 부여하고 알맞은 디바이스에 마커를 발급한 뒤 이를 카메라 등으로 촬영하여 디바이스의 존재를 인식하고 기능을 제공하는 것이다[3]. 네트워크를 이용하는 방법의 경우 근거리 통신망이나 인터넷을 이용해 디바이스를 발견하고 연결하는 방식이다. 근거리 통신망을 사용하여 멀티 디스플레이 환경을 실현한 대표적인 사례로는 HuddleLamp[4]와 Connichiwa[5]이 있는데 이들은 특별한 장치를 사용하거나 주로 테이블 위와 같이 좁은 범위의 공간에만 적용이 가능하다는 단점이 있다.

따라서 본 논문에서는 어느 정도 떨어진 거리의 디바이스를 쉽게 발견하고 연결할 수 있는 기술을 사용하여 넓은 공간에서의 멀티 디스플레이 환경을 제공하는 것을 목표로 하였다. 또한 일반 카메라를 이용해 디바이스의 배치를 인식하고 웹을 기반으로 하여 효율성이나 비용을 낮추는 것을 목표로 하였다. 본 논문에서는 이와 같은 요구를 반영하여 작업 현장에서 저작자의 인터랙션을 통해 멀티 디바이스 환경을 조작할 수 있는 현장 저작용 시스템을 설계하였다. 본 시스템에서는 UPnP를 이용하여 보다 넓은 환경에서 디바이스의 존재를 인식하고 디바이스의 정보를 얻을 수 있으며[6,7], 웹캠과 영상처리기술을 통해 디바이스 배치 상태를 인식할 수 있다. 또한, 웹을 기반으로 한 저작 인터페이스 구현을 통해 다양한 사양의 멀티 디바이스에서 디바이스 간 연결을 구축하고 콘텐츠를 제어할 수 있도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서 본 연구의 유사 연구인 HuddleLamp와 Connichiwa를 분석하여 설계에 참고할 사항들을 도출한다. 3장에서는 2장에서 도출한 내용을 토대로 설계한 시스템에 대해 설명하고 각 단계 별 진행 과정과 구현한 시스템에 대한 내용을 서술한다. 그리고 4장에서는 제안한 시스템으로 구현한 결과와 구현 내용으로 실험한 결과를 통해 시스템이 연구의 목적과 목표에 부합한가에 대해 평가한 뒤, 5장에서 결론을 맺는다.

2. 관련 연구

2.1 HuddleLamp

HuddleLamp는 기존의 멀티 디스플레이 환경에서 다른 디바이스의 존재를 인식할 수 없어 디바이스 간 작업 공유가 불가능했다는 점에 착안하여 설계된 무료 오픈소스 소프트웨어이다[4]. Ad-hoc 통신을 이용하여 테이블 위에 존재하는 모바일 디스플레이들의 협업을 지원한다. 카메라를 이용해 디바이스의 존재를 확인하고 위치를 추적하기 때문에 별도의 어플리케이션을 설치하지 않아도 디바이스들을 지속적으로 감지할 수 있다. 본 소프트웨어에서는 HuddleLamp와 프로세싱을 위한 PC를 통해 멀티 디스플레이 환경을 조성한다. HuddleLamp의 갓 안쪽에 RGB-D 카메라가 설치되어 있어 디바이스의 존재와 위치를 인식한다. Fig. 1과 같이 RGB-D 카메라는 촬영되는 화면에서 RGB 값을 추출할 수 있고, Depth를 통해 객체와의 거리를 추측할 수 있게 한다.

HuddleLamp는 디바이스의 존재를 인식하고 위치를 추적할 뿐만 아니라 사용자의 인터랙션까지 멀티 디스플레이 환경에 적용할 수 있게 만든 사례이다. 하지만 RGB-D 카메라나 센서를 사용하므로 이에 대한 추가 비용이 발생된다. 또한 탁상 위의 좁은 범위에서 디바이스 협업을 목표로 하였기 때문에 이보다 넓은 공간에서 멀티 디스플레이 환경은 실현이 어렵다는 단점이 있다.

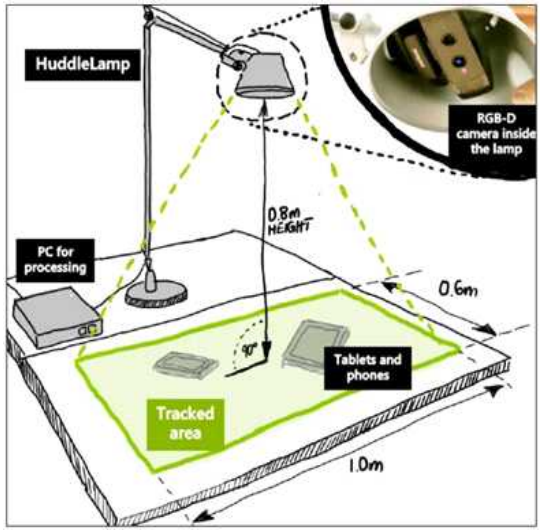

Fig. 1. Establishment of HuddleLamp.

2.2 Connichiwa

Connichiwa는 디바이스 간의 협업을 위해 블루투스 통신망을 형성하여 디바이스 간의 논리적인 연결을 가능하게 하는 웹 어플리케이션 프레임워크이다[5]. 기존의 멀티 디스플레이 환경에서 부가적인 하드웨어 장치나 인터넷 접속 없이는 환경 조성이 불가하다는 문제점을 해결하기 위해 개발되었다. Connichiwa는 인터넷 접속과 무관한 서비스를 제공하기 위하여 Fig. 2와 같이 사용자 인터랙션과 블루투스 통신망을 이용하여 디바이스 간의 의미적인 연결이 가능하도록 하였다. Connichiwa는 디바이스 협업 어플리케이션을 위한 프레임워크까지 개발되어 있다. 그러나 근거리 블루투스 통신을 사용하므로 디바이스 간의 거리가 멀어지면 네트워크에서 제외된다. 또한 디바이스간의 연결 상태만 인지할 뿐이며 물리적인 배치 상태는 알 수가 없다.

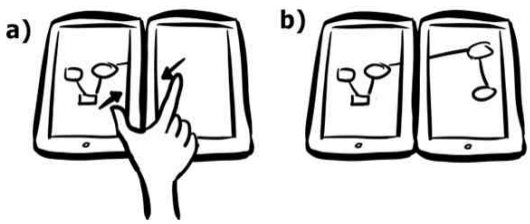

Fig. 2. "Device Connection" Examples in Connichiwa.

3. 멀티 디스플레이 콘텐츠 전송 시스템 설계 및 구현

3.1 시스템 개요

본 연구의 목표는 좁은 범위의 멀티 디스플레이 환경을 조성하는 데에 집중되어있는 기존 연구에서 벗어나 보다 넓은 범위의 멀티 디스플레이 환경을 조성하는 것이다. 또한 디스플레이 디바이스의 배치가 미리 고정되어 있는 환경이 아니라 콘텐츠의 성격에 따라 위치가 조정되거나 아예 재배치 할 수 있도록 하는 것을 목표로 하였다. 넓은 공간에서의 가변적인 멀티 디스플레이 환경을 조성하기 위한 기반 기술을 이용해 기본적인 시스템을 설계하고 구현하였다.

멀티 디스플레이 콘텐츠 전송 시스템은 Fig. 3과 같이 메인 콘트롤 역할을 하는 저작 디바이스, 콘텐츠를 보여주는 디스플레이 디바이스, 콘텐츠 전송을 담당하는 고정 서버로 구성되어 있다.

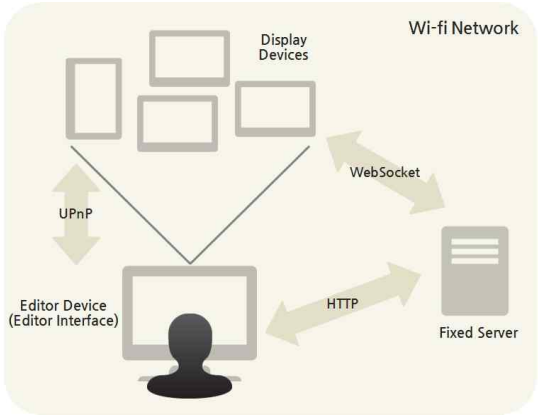

Fig. 3. Configuration of Multi-Display Contents Delivery System.

먼저 디스플레이 디바이스는 UPnP를 이용해 자신의 정보를 저작 디바이스로 보내고, 지정색을 표시해 자신들의 배치 상태를 인식할 수 있게 하며, 고정 서버로부터 콘텐츠를 전달받아 재생하는 역할을 한다. 저작 디바이스는 UPnP를 이용해 디스플레이 디바이스들의 정보를 수신하고, 웹 페이지를 통해 저작 인터페이스를 제공하여 복합 콘텐츠를 저작하며, 저작한 복합 콘텐츠를 고정 서버로 전달한다. 고정 서버는 저작 디바이스로부터 복합 콘텐츠 정보를 전달받고, 이를 토대로 웹 소켓을 통해 콘텐츠를 디스플레이 디바이스로 전송한다. Fig. 3과 같이 세 종류의 디바이스는 HTTP 통신을 기반으로 하는 UPnP 통신과 웹 소켓 통신으로 연결되어 있으며, 각각의 디바이스는 동일 Wi-fi 네트워크에 연결되어 있어야 한다.

3.2 시스템 설계

시스템의 작업 순서는 Fig. 4에서 보듯이, 크게 디바이스 연결, 디바이스 배치 인식, 디바이스 정보와 물리적 배치 상태 연결, 복합 콘텐츠 저작, 콘텐츠 전송 등 다섯 단계로 나눌 수 있다.

첫 번째인 디바이스 연결 단계에서 디스플레이 디바이스와 저작 디바이스가 UPnP 연결을 수립하고, 연결 과정에서 발견한 디스플레이 디바이스 목록과 디바이스 정보를 수집한다. 수집한 정보는 목록화하여 저작 디바이스에서 보여준다.

두 번째로 디바이스 배치 인식 단계에서는 저작

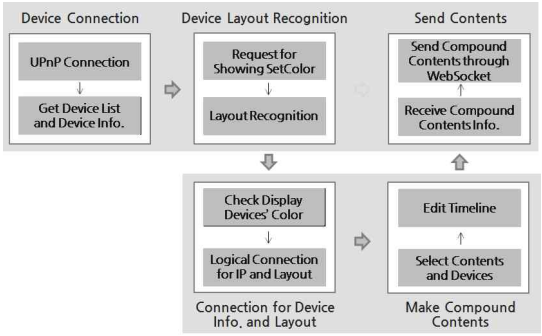

Fig. 4. A Flow Diagram for Multi-Display Contents Delivery System.

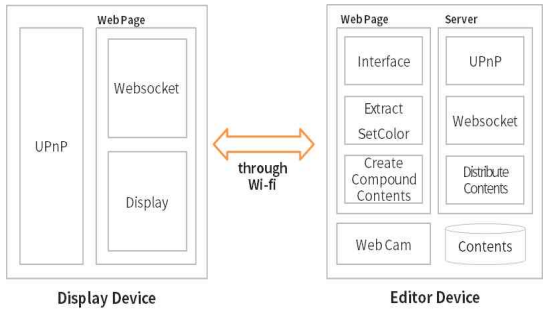

Fig. 5. Overall System Structure.

디바이스가 디스플레이 디바이스에게 지정색 표시를 요청하고, 요청을 수신한 디스플레이 디바이스가 지정색을 표시하면 웹캠을 통해 이를 캡쳐한다. 저작 인터페이스에서는 캡쳐된 화면에서 지정색을 추출하여 디스플레이 디바이스들의 물리적인 배치 상태를 인식하고, 이를 인터페이스에 표시하여 저작자가 복합 콘텐츠 저작 시 이를 사용할 수 있게 한다.

다음으로는 첫 번째 단계에서 얻은 IP와 같은 디스플레이 디바이스들의 정보와 두 번째 단계에서 얻은 디스플레이 디바이스들의 물리적 배치 상태를 연결하기 위해 디스플레이 디바이스에게 제 2 지정색을 요청한다. 이 과정은 두 정보의 논리적인 연결을 만들어주기 위한 것으로, 목록에 있는 IP를 선택하면 해당하는 IP를 가진 디스플레이 디바이스에 제 2 지정색이 표시되고, 이를 인식하여 물리적 배치 상태에 반영하는 것이다. 이 과정을 통해 추출된 객체가 해당 IP를 가진 디바이스라는 것을 알 수 있게 된다.

이후, 저작자가 복합 콘텐츠를 생성하기 위한 콘텐츠 정보와 디바이스 정보를 선택하고 타임라인을 이용하여 선택한 사항들이 재생되는 시점을 정하는 등 복합 콘텐츠를 저작 과정을 수행한다. 저작된 복합 콘텐츠는 저작자가 재생 버튼을 누르면 고정 서버를 통해 디스플레이 디바이스로 전송된다. 일련의 과정을 통해 디바이스 배치 상태를 인식하여 복합 콘텐츠를 전송할 수 있는 시스템을 설계하였다.

3.3 멀티 디스플레이 콘텐츠 전송 시스템의 구현

3.3.1 전체 시스템의 구성

멀티 디스플레이 콘텐츠 전송 시스템은 디스플레이 디바이스와 저작 디바이스, 고정 서버 세 가지로 이루어진다. 하지만 구현 환경의 복잡성을 줄이기 위해 Fig. 5와 같이 고정 서버와 저작 디바이스를 한 대의 디바이스로 묶어서 구현하였다. 추후 크고 복잡한 콘텐츠 전송이 필요한 경우 두 디바이스를 분리하여 사용할 수 있다.

디스플레이 디바이스의 경우 크게 UPnP와 웹 소켓[8] 두 부분으로 나눌 수 있다. UPnP는 저작 디바이스에서 디스플레이 디바이스를 발견할 수 있게 한다. 저작 디바이스는 저작 인터페이스를 나타내는 웹 페이지와 고정 서버의 역할을 하는 Node.js[9] 서버로 이루어진다. 저작 인터페이스는 인터페이스 화면 자체와 지정색 추출, 복합 콘텐츠 저작 세 부분으로 이루어져 있으며, 웹캠을 이용하여 디스플레이 디바이스의 물리적 배치 상태를 인식한다. 또한 고정서버와 인터페이스 서버를 합쳐 디스플레이 디바이스를 발견하기 위한 UPnP와 콘텐츠를 전송하기 위한 웹 소켓, 그리고 추후 추가될 복합 콘텐츠 재생에 필요한 콘텐츠 분배 기능을 구현하였다.

멀티 디스플레이 콘텐츠 전송 시스템의 구현은 웹을 기반으로 구현되었다. 디바이스 간의 발견을 위해 UPnP 프로토콜을 사용했고 콘텐츠 전송을 위해 웹 소켓을 사용하였다. Network Service Discovery[10]를 이용하여 UPnP 통신을 가능하게 하는 자바스크립트 API plug.play.js[11]을 이용해 UPnP 연결을 수립했고, Socket.io[12]를 사용하여 웹 소켓 연결을 수립했다. 이를 통해 Wi-fi네트워크를 이용한 시스템의 전체적인 연결을 구현했다. 저작 인터페이스의 경우 웹캠과 영상처리를 위한 자바스크립트 라이브러리를 이용해 물리적 배치 상태 인식 기능을 구현하였다. 또한 자바스크립트 라이브러리와 CSS를 이용해 웹페이지 기반의 인터페이스 환경을 구현하였다.

3.3.2 UPnP를 이용한 디바이스 발견 및 제어

디바이스 연결은 저작 디바이스가 디스플레이 디바이스들의 존재를 확인하고 디스플레이 디바이스들의 정보를 요청하여 필요한 정보를 추출해내는 과정이다. 먼저 디바이스 연결은 UPnP를 통해 이루어진다. 디스플레이 디바이스의 경우 웹 페이지를 통해 동작하는데 UPnP 통신의 수립은 웹 브라우저 자체에서 이루어진다. 따라서 Network Service Discovery를 이용하기 위한 자바스크립트 API인 plug.play.js와 Node.js의 peer-upnp 모듈[13]를 이용해 UPnP의 피 제어 장치를 구현하였다. 저작 디바이스의 경우 plug.play.js를 이용해 UPnP가 가능한 디바이스를 모두 발견할 수 있는 UPnP 서버를 구축하였다.

본 연구에서 저작 디바이스와 디스플레이 디바이스의 UPnP 네트워킹 과정은 Fig. 6과 같이 주소 지정 단계, 장치 검색 단계, 장치 기술 단계, 제어 단계, 이벤팅 단계로 이루어져 있다.

UPnP 는 IP 기반의 프로토콜이므로 디스플레이 장치들 마다 IP 가 필요하다. 주소 지정 단계에서는 저작 디바이스와 디스플레이 디바이스가 동일한 네트워크에 접속할 때 IP 주소를 할당 받는다.

장치 검색 단계에서 디스플레이 디바이스가 SSDP 프로토콜을 사용하여 저작 디바이스를 검색한다. 피 제어 장치인 디스플레이 디바이스에서는 UPnP를 사용하기 위한 peer를 생성하고, 준비가 되었다는 신호를 제어 장치로 보낸다. 제어 장치인 저작 디바이스에서는 피 제어 장치에서 보낸 신호를 받아 연결을 수립한다.

장치 기술 단계에서는 디스플레이 디바이스의 장치 기술 문서와 서비스 기술 문서를 요청하고 XML 형식으로 작성된 이 문서들을 통해 디스플레이 디바이스에 대한 정보를 얻는다. 장치 기술 문서에는 피 제어 장치의 정보나 실행 가능한 서비스 목록 등이 담겨있고 서비스 기술 문서에는 디스플레이 디바이스에서 실행시키고자 하는 서비스가 응답할 명령, 각 명령에 대한 매개변수 리스트 등이 담겨있다.

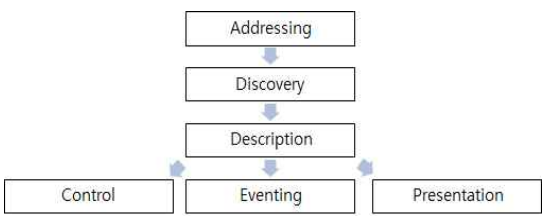

Fig. 6. UPnP Networking Process.

제어 단계에서 저작 디바이스는 장치 기술 단계에서 얻은 정보들을 분석하고 디스플레이 디바이스에게 명령어를 보낼 수 있다. 저작 디바이스에서 제공하는 서비스의 상태가 변화하게 되면 서비스의 상태를 나타내는 변수들이 변화하게 된다. 이벤팅 단계에서 저작 디바이스는 상태 변수들의 변화를 통보하고 디스플레이 디바이스가 이 정보를 수신 받는다.

3.3.3 디스플레이 디바이스의 배치 상태 인식 및 연결

디바이스 배치 상태 인식 과정은 저작 디바이스가 설치된 카메라를 이용하여 디스플레이 디바이스들의 배치 상태를 캡쳐하고 캡쳐한 화면에서 디스플레이 디바이스들을 추출해내 가지고 있는 정보와 연결짓는 과정이다. 저작 디바이스에서는 디스플레이 디바이스의 물리적인 배치 상황을 이미지로 캡쳐한 화면에서 지정색을 추출하도록 하였다. 또한 HTTP 연결을 통해 디스플레이 디바이스에 2차 지정색을 표시하게 하여 저작자가 디바이스 정보와 배치 상태를 수동으로 연결할 수 있게 하였다. 디스플레이 디바이스는 요청을 수신하여 지정색을 화면에 표시한다.

디바이스의 물리적 배치 상태 인식을 위해 저작 디바이스에서 필요한 기능은 카메라 캡쳐 기능과 캡쳐 화면에서의 지정색 추출 기능, HTTP를 통한 신호 송신 기능, 디바이스 정보와 추출한 색 객체 연결 기능이다. 먼저 카메라 캡쳐 기능과 캡쳐 화면에서의 지정색 추출 기능은 자바스크립트 라이브러리인 tracking.js[14] 라이브러리를 사용하여 구현하였다. tracking.js 라이브러리는 HTML5 표준 사양에서 실시간으로 색을 트래킹하고, 얼굴을 인식하는 등의 기능을 제공할 수 있는 가벼운 자바스크립트 라이브러리이다. tracking.js 라이브러리는 HTTP 통신이 가능한 웹 페이지와 웹 카메라가 있는 PC 환경에서 'tracking.js' 파일을 참조함으로써 사용 가능하다.

디바이스의 물리적인 배치 상태를 인식하고 나면 디바이스 정보와 이를 논리적으로 연결시켜 주어야 한다. 디바이스의 정보는 앞 절에서 UPnP를 통해 얻은 디바이스 리스트로, 디바이스에 부여한 ID와 디바이스의 IP로 이루어져 있다. 먼저, Fig. 7에서 보듯이 저작 인터페이스에 존재하는 디바이스 리스트에서 한 행을 선택하면 해당 행의 디바이스 IP로 제 2 지정

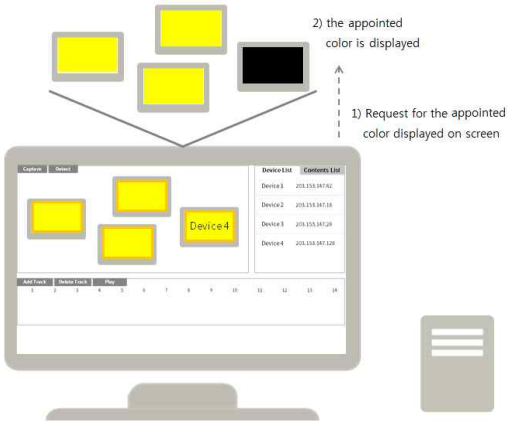

Fig. 7. Process to Recognize the Devices Layout.

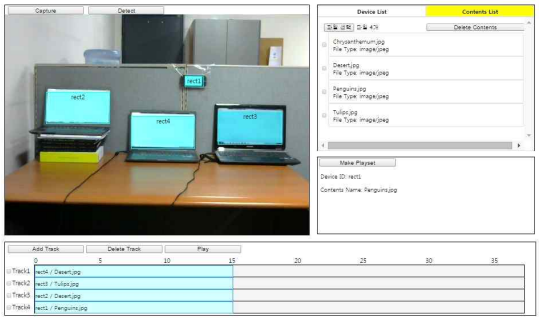

Fig. 8. The Interface to Recognize Device Layout.

색을 화면에 띄우도록 요청한다. 선택 후 저작자는 디스플레이 디바이스 배치를 육안으로 확인하여 인식한 디바이스 배치 상태 화면에 있는 디스플레이 사각형 객체 중 해당 디바이스와 일치하는 위치에 있는 객체를 선택한다. 이와 같은 수작업을 통해 디바이스의 물리적인 배치상태와 디바이스의 정보를 논리적으로 연결시킨다. Fig. 8은 디바이스 배치를 위한 인터페이스의 구현 결과 화면이다.

4. 실험 결과 및 분석

본 장에서는 기존의 타 연구에서는 시도하지 않았던 다양한 범위의 멀티 디스플레이 환경에서 디바이스의 연결 및 배치상황의 인식이 잘 이루어지는지 확인하고자 실험한 결과를 설명한다. 이를 위하여 몇 가지의 특정 범위를 설정하여 디바이스 발견 및 배치 인식 실험을 진행하고 결과를 분석하였다. 다양한 환경적 변수로 인해 정확한 발견 및 인식 가능 범위를 측정하는 방식이 아닌 특정 범위에서 발견 및 인식이 가능함을 보였다. 실험은 Fig. 9와 같이 약 1.5m 범위의 방 안, 약 3m 범위의 좁은 전시장, 약 5m 범위의 넓은 전시장으로 범위를 설정하여 진행되었다.

4.1 디바이스 발견 실험

디바이스 발견 실험은 UPnP를 이용하면 좁지 않은 범위에서도 디바이스 발견이 가능하다는 것을 보이기 위한 실험이다. 각각의 실험에서는 4 대의 디스플레이 디바이스들과 저작 디바이스를 각각 설정된 거리의 간격으로 배치한 뒤, 동일한 Wi-fi 네트워크에 연결하여 저작 디바이스가 디스플레이 디바이스를 발견하는 방식으로 진행되었다.

실험은 각각의 장소에서 세 번씩 진행되었다. 각각의 장소에서 세 번의 실험을 진행한 결과, Table 1의 결과에서 보듯이 방 안과 좁은 전시장, 넓은 전시장에서 각각의 장소에서 4 대의 디바이스를 모두 발견했다. UPnP는 무선 인터넷 망을 기반으로 통신을 제공하는 기술로, 동일한 인터넷 망에 연결되어 있다면 대부분의 디바이스를 발견할 수 있다. 따라서 동일한 AP(Access Point)를 통해 Wi-fi 망에 연결할 수만 있다면 연결된 디바이스 중 UPnP 연결을 지원하는 디바이스를 모두 발견할 수 있을 것이다.

Fig. 9. Comparison of Plot Plans according to Experiment Place.

Table 1. Experimental Result of Device Connection Test

	Room (about 1.5m)	Medium-size Hall (about 3m)	Large-size Hall (about 5m)
Experiment 1	discover all 4 devices	discover all 4 devices	discover all 4 devices
	100%	100%	100%
Experiment 2	discover all 4 devices	discover all 4 devices	discover all 4 devices
	100%	100%	100%
Experiment 3	discover all 4 devices	discover all 4 devices	discover all 4 devices
	100%	100%	100%
average	100%	100%	100%

4.2 디바이스 배치 인식 실험

디바이스 배치 인식 실험은 웹캠과 자바스크립트 영상처리 기법을 이용하면 좁지 않은 범위에서도 마커와 같은 복잡한 매개체 생성 없이 디바이스 배치 인식이 가능하다는 것을 보이기 위한 실험이다. 실험 환경은 앞 절과 동일하다. 디바이스를 주어진 간격에 맞게 배치한 뒤 저작 인터페이스에서 지정색 표시 요청을 보내면 디스플레이 디바이스가 지정색을 표시한다. 이 때, 모든 디바이스는 UPnP 연결이 수립되어있다는 것을 가정한다. 이후 웹캠을 통해 저작 인터페이스가 화면을 캡쳐하고 캡쳐된 화면에서 지정색을 추출하여 디바이스의 배치를 인식한다.

실험은 Table 2에서 보듯이 각각의 장소에서 세 번씩 진행되었다. 각각의 장소에서 세 번의 실험을 진행한 결과, 방 안과 좁은 전시장에서는 4대의 디바이스를 모두 발견했다. 그러나 넓은 전시장의 경우 세 번의 실험 중 두 번의 실험에서 한 대의 디바이스를 발견하는데 실패했다. 즉. 대부분의 장소에서 모든 디바이스를 발견했으나 약 5m 범위의 넓은 전시장에서는 1대의 발견을 실패했다. 지정색 추출만으로도 물리적인 배치 상태 인식이 가능하지만 웹캠의 성능에 따라 그 거리에는 차이가 있을 수 있다는 것을 알 수 있다.

5. 결 론

현재의 멀티 디스플레이 환경 조성에 대한 연구는 개인의 디바이스를 이용한 소규모 작업 공간을 조성하는 것에 집중되어 있다. 이에 본 논문에서는 보다 넓은 범위에서 가변적인 멀티 디스플레이 환경 조성을 위한 기반 기술과 이를 이용한 시스템을 설계 및 제안하였다. 제안한 시스템을 이용하면 마커와 같은 매개체나 고가의 장비 없이도 디바이스의 존재를 인식하고 동일 Wi-fi 네트워크에 접속하는 것만으로도 디바이스의 정보를 수신하며 필요한 콘텐츠를 전송할 수 있다.

그러나 본 연구에서 제안한 시스템은 재배치는 가능하지만 고정된 디바이스 레이아웃 인식과 단순한 콘텐츠 전송만이 가능하다. 지속적으로 변하는 유동적인 디바이스 레이아웃 인식과 다양한 콘텐츠 스케줄링 기법이 본 시스템에 추가된다면 넓은 범위의 멀티 디스플레이 환경을 보다 완전하게 조성할 수 있을 것으로 보인다.

Table 2. Experimental Result of Device Layout Recognition Test

	Room (about 1.5m)	Medium-size Hall (about 3m)	Large-size Hall (about 5m)
Experiment 1	discover all 4 devices	discover all 4 devices	fail on 1 device
	100%	100%	75%
Experiment 2	discover all 4 devices	discover all 4 devices	discover all 4 devices
	100%	100%	100%
Experiment 3	discover all 4 devices	discover all 4 devices	fail on 1 device
	100%	100%	75%
average	100%	100%	83.3%

REFERENCE

[1] S.Y. Jeon, S.A. Jung, J.H. Park, H.J. Yim, B.J. B, and S.B. Lim, "A Study on Contents Delivery between Connected Devices in Second Screen Service," *Proceeding of the 2015 Korea Computer Congress of the Korean Institute of Information Scientists and Engineers*, pp. 398-400, 2015.

[2] J.H. Park, S.Y. Jeon, S.A. Jung, H.J. Yim, B.J. Bae, and S.B. Lim, "Development of a Second Screen Service for Personalized Contents Delivery," *International Journal of Control and Automation*, Vol. 8, No. 9, pp. 321-330, 2015.

[3] M. Hirzer, *Marker Detection for Augmented Reality Applications*, *Technical Report ICG-TR-08/05,* Inst. for Computer Graphics and Vision, Graz University of Technology, Oct. 2008.

[4] R. Rädle, H.C. Jetter, N. Marquardt, H. Reiterer, and Y. Rogers, "HuddleLamp: Spatially-Aware Mobile Displays for Ad-hoc Around-the-Table Collaboration," *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, pp. 45-54, 2014.

[5] M. Schreiner, R. Rädle, H.C. Jetter, and H. Reiterer, "Connichiwa: A Framework for Cross-Device Web Applications," *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 2163-2168, 2015.

[6] UPnP Forum, UPnP Device Architecture 1.1, http://www.upnp.org (accessed Aug. 14, 2015).

[7] K. Kim and E. Jung, "A UPnP A/V Multimedia System using Prediction of Mobility for Mobile User," *Journal of Korea Multimedia Society,* Vol. 12, No. 11, pp. 1509-1520, 2009.

[8] I. Hickson, The WebSocket API, http://www.w3.org/TR/websocket (accessed Aug. 14, 2015).

[9] Node.js Foundation, Node.js, https://nodejs.org/en (accessed Aug. 14, 2015).

[10] R. Tibett, Network Service Discovery, http://www.w3.org/TR/discovery-api/ (accessed Aug. 14, 2015).

[11] Plug.Play.js by Richtr, https://richtr. github.io/plug.play.js/ (accessed Oct. 30, 2015).

[12] Socket.io, SOCKET.IO 1.0 IS HERE, http://socket.io/ (accessed Aug. 14, 2015).

[13] Fraunhofer FOKUS, Peer-upnp: Nodejs Implementation of the UPnP Device Architecture 1.1, https://www.npmjs.com/pckage/peer-upnp (accessed Aug. 14, 2015).

[14] E. Lundgren, Tracking.js: A Modern Approach for Computer Vision on the web, http://trackingjs.com/ (accessed Aug. 14, 2015).

전 소 연

2014년 2월 숙명여자대학교 멀티미디어과학과 학사
2016년 2월 숙명여자대학교 멀티미디어과학과 석사
관심분야 : Multi-display technology, Web programming, User interface

임 순 범

1982년 2월 서울대학교 자연과학대학 계산통계학과 학사
1983년 8월 한국과학기술원 전산학과 석사
1992년 2월 한국과학기술원 전산학과 박사
1992년 2월 (주) 휴먼컴퓨터 창업(연구소장)
1997년 2월 (주) 삼보컴퓨터 부장(프린터개발부)
2001년 8월 건국대학교 컴퓨터과학과 교수
2001년 9월 ~ 현재 숙명여자대학교 IT공학과 교수
2006년 University of Colorado 방문교수
관심분야 : 컴퓨터그래픽스, 웹/모바일 멀티미디어 응용, 디지털 방송, 전자출판, User Interface