Smart Media Journal / Vol.15, No.1 / ISSN:2287-1322
https://dx.doi.org/10.30693/SMJ.2026.15.1.90

관객참여형 스마트폰 공연 연출을 위한 AI 제어 모델의 UX 실증 연구

(An Empirical UX Study of an AI Control Model for Audience-Interactive Smartphone Performance Design)

유장웅*, 양주성**
(You Jang Woong, Yang Ju Sung)

요 약

스마트폰과 무선 네트워크의 확산은 관객을 공연 연출에 능동적으로 참여시키는 환경을 가능하게 하였으나, 기존 공연장 제어 시스템은 오퍼레이터 중심 구조로 인해 다수 관객의 입력을 안정적으로 반영하는 데 한계가 있다. 특히 네트워크 혼잡과 단말 성능 차이로 인해 end-to-end 지연과 실행 시각 분산이 발생하며, 이는 관객 참여형 공연 연출의 서비스 품질(Quality of Service, QoS)을 저하시킨다. 이러한 문제를 해결하기 위해, 관객 참여 데이터를 입력으로 받아 공연 환경에 적합한 제어 파라미터를 산출하는 AI 기반 적응형 제어 모델을 설계하고, 실제 공연장에서 실증 실험을 진행하였다. 실험 결과, 실시간 통신과 좌석 기반 매핑 구조를 통해 다수 단말 환경에서도 안정적인 QoS를 유지하였으며, AI 기반 제어를 공연 연출 프로세스에 통합하여 관객이 실시간 공동 연출자로 참여할 수 있는 AI 기반 적응형 제어 구조의 실효성을 검증하였다.

■ 중심어 : 관객 참여형 공연 ; 스마트폰 인터렉션 ; 실시간 무대 제어 ; DMX ; Art-Net

Abstract

The widespread adoption of smartphones and wireless networks enables audiences to actively participate in performance production; however, conventional stage control systems remain operator-centered and struggle to stably reflect large-scale audience inputs. Network congestion and device heterogeneity lead to end-to-end latency and execution variance, degrading the quality of service (QoS) in audience-interactive performances. To address these issues, this study designs an AI-based adaptive control model that generates performance-appropriate control parameters from audience interaction data and validates it through empirical experiments conducted in a real performance venue. The results confirm that stable QoS can be maintained in multi-device environments using real-time communication and seat-based mapping, demonstrating the effectiveness of integrating AI-based control into the performance production process to enable audiences to participate as real-time co-creators.

■ keywords : audience participation performance ; smartphone Interaction ; real-time stage control ; DMX ; Art-Net

Ⅰ. 서 론

공연 예술 환경은 최근 관객의 참여와 상호작용을 핵심 가치로 하는 방향으로 변화하고 있다. 특히 모바일 디바이스 보급과 공연장 내 네트워크 인프라의 향상은 관객이 실시간으로 공연에 개입하는 새로운 흐름을 촉발하였다[1]. 기존 공연장 제어 체계는 주로 조명 콘솔, 미디어 서버, 그리고 DMX512 및 Art-Net 기반 장비를 중심으로 운영되며, 이는 전문 오퍼레이터 중심

* 정회원, 남부대학교 IT·디자인학과
** 정회원, 레피소드㈜ 대표이사
본 과제(결과물)는 2025년도 교육부 및 광주광역시의 재원으로 광주RISE센터의 지원을 받아 수행된 지역혁신중심 대학지원체계(RISE)의 결과입니다.(2025-RISE-05-007)

접수일자 : 2025년 12월 15일
수정일자 : 2026년 01월 09일
게재확정일 : 2026년 01월 12일
교신저자 : 김정민 e-mail : jmkim@jnu.ac.kr

의 폐쇄적 구조를 갖는다[4,5]. 이로 인해 관객의 실시간 입력 데이터를 연출 요소에 직접적으로 반영하는 데 구조적 제약이 존재했지만, 관객 스마트폰을 공연 시스템과 연동하면 관객의 제스처·색·터치·플래시 데이터 등을 무대 조명 또는 영상 효과로 실시간 변환할 수 있다. 최근 모바일 기기를 집단적 조명 장치나 인터랙티브 매체로 활용하는 시도가 증가하고 있으며, 이는 관객이 단순한 '정보수용자'에서 '공연 공동 창작자'로 확장될 수 있음을 시사한다[2][6,7].

그러나 관련 연구들이 주로 입력 데이터의 집계 방식 또는 단순 매핑 규칙 중심의 접근에 머물러 있어, 실시간성·동기화·안정성·공간 좌표 기반 제어 문제를 충분히 해결하지 못하고 있고, 특히 다수의 관객 단말이 존재하는 공연 환경에서는 네트워크 혼잡, Wi-Fi 지연, 장치 처리 속도 차이 등으로 인해 지연 분산(latency variance)이 발생하여 일관성 있는 연출 효과를 구성하는 데 큰 어려움이 있다[3].

이에 본 연구는 기존 참여형 공연 기술의 한계를 극복하기 위해 첫째 AI 제어 모델 기반 매핑 알고리즘, 둘째 WebSocket 동기화 기반 실시간 제어 서버, 셋째 좌석 기반 무대-관객 공간 매핑 엔진, 넷째 관객 스마트폰 PWA(Progressive Web App) 제어 시스템을 통합한 새로운 공연 연출 구조를 제안하고자 한다. 특히 스마트폰에서 수집된 입력 데이터를 단순 집계가 아니라 AI 기반 필터링 및 군집 매핑 모델을 적용한 동적 연출 파라미터로 변환함으로써, 기존의 규칙 기반 조명 제어 방식과 구별되는 생성형 조명 제어(Generative Lighting Control)를 제시하고자 한다[8].

II. 관련 연구

2.1 기술 매개 관객 참여 모바일 공연 시스템

기술 매개 관객 참여(Technology-Mediated Audience Participation, TMAP)는 공연장 내에서 관객의 디지털 기기를 매개로 한 실시간 상호작용을 통해 공연의 구조와 감정을 재구성하는 방식을 의미한다[1]. M. Hödl와 B. Taylor의 연구에서는 스마트폰을 악기이자 조명 장치로 활용하는 '스마트폰 오케스트라'와 같은 프로젝트를 통해 다수 관객의 입력을 실시간으로 수집·집계하여 음악 및 시각적 연출에 반영하는 가능성을 제시하였다[1,2]. 이러한 연구는 관객이 단순한 수용자를 넘어, 집단적인 데이터의 생산자로서 공연 전개에 영향을 미친다는 점에서 의미가 크지만, 주로 규칙 기반 매핑 또는 단순한 평균·최빈값 중심의 집계 방식에 머무르는 한계가 있다. 특히 관객 수가 증가할수록 네트워크 지연과 데이터 변동성이 커져, 연출의 안정성과 미적 일관성이 저하되는 문제가 보고되고 있다[3].

따라서 관객 참여형 공연 시스템의 확장 가능성은 지연(time latency) 관리, 단말 간 동기화, 공간 정보(좌석) 기반 제어, 입력 신호의 지능형 필터링을 어떻게 구현하는지에 따라 결정된다. 본 연구는 이 한계를 극복하기 위해 AI 제어 모델을 도입해 관객 입력을 안정적으로 연출 파라미터로 변환하는 구조를 제안한다[8].

2.2 WebSocket 기반 실시간 제어 기술

실시간 상호작용을 위한 웹 기술로는 HTTP 폴링·롱폴링 대비 WebSocket이 지연과 트래픽 측면에서 유리하다는 연구가 제시되어 왔다[19]. WebSocket은 클라이언트와 서버 간에 상시 연결을 유지하면서 양방향 통신을 지원해, 센서 데이터 모니터링이나 디지털 트윈 제어와 같은 실시간 응용에서 높은 성능을 보인다[9][14].

최근 WebSocket을 이용한 실시간 제어 시스템에서 응답시간 분포, 연결 안정성, 자원 사용량을 평가한 결과, 실시간 제어 시스템의 요구사항(수십~수백 ms 수준)을 충족할 수 있음을 보였다[14][17]. 본 연구 역시 NestJS 기반 WebSocket Gateway와 Socket.IO 클라이언트

를 활용해 관객 스마트폰과 공연 제어 서버 간 양방향 통신을 구현하며, 명령 전송·수신·완료 시간의 로그를 분석하여 공연 환경에서의 실시간성을 실험적으로 확인하였다[3][9].

2.3 PWA 및 서비스 워커 기반 공연용 웹 애플리케이션

관객 단말을 스마트폰 앱이 아닌 브라우저 기반으로 통일하기 위해서는 PWA(Progressive Web App) 구조가 적합하다. PWA는 서비스 워커(Service Worker)와 캐시 전략을 통해 네트워크 상태에 의존하지 않고 빠른 초기 로딩과 오프라인 대응을 제공하는 웹 애플리케이션 형태로, 최근에는 모바일 앱 대체재로 활용되는 추세다[11][17]. 서비스 워커는 브라우저와 네트워크 사이에서 작동하는 프록시로, 정적 리소스를 캐시에 저장해 반복 접속 시 체감 속도를 높이고, 네트워크 장애 시에도 최소한의 UI를 제공할 수 있도록 한다[19]. Angular 프레임워크 역시 공식적으로 서비스 워커 모듈을 제공하여, 빌드 시 자동으로 캐시 전략을 적용하는 등 PWA 구성을 지원한다[12].

연구를 위한 실험 환경은 관객용 클라이언트를 Angular 기반 PWA로 구현하고, 핵심 UI와 아이콘·폰트 등의 정적 자산을 캐싱하여 공연장 Wi-Fi 환경에서 발생할 수 있는 일시적인 네트워크 품질 저하가 연출 흐름에 미치는 영향을 최소화하는 것을 목표로 하였다.

2.4 스마트폰 카메라 및 Torch 제어 연구

브라우저 상에서 카메라 및 플래시를 제어하기 위해서는 MediaDevices.getUserMedia와 MediaStreamTrack constraints를 활용해야 한다[10][15]. 최신 브라우저는 해상도, 초점, 줌, 지속 점등 플래시(torch) 여부와 같은 다양한 제약 조건을 API로 노출하며, 특히 Chrome for Android에서는 torch를 포함한 확장된 제약 조건을 지원한다는 점이 보고되고 있다[15,16].

하지만 torch 지원 여부는 디바이스·브라우저 조합에 따라 상이하고, 일부 환경에서는 제약 조건이 완전히 구현되지 않았거나 권한 이슈로 인해 동작하지 않는 문제가 존재한다[18]. 따라서 공연장 환경에서 스마트폰 플래시를 대규모로 제어하기 위해서는 지원 여부 탐지, 재시도 로직, 오류 처리, 그리고 torch가 지원되지 않는 단말에 대한 화면 색상 표시 등 대체 표현을 함께 설계해야 한다.

본 연구는 MediaDevices API를 기반으로 카메라 스트림을 확보한 뒤, 비디오 트랙의 capabilities와 constraints를 점검하여 torch 제어 가능 여부를 판단하고, 지원되지 않는 환경에서는 화면 색상만 변경하는 폴백 전략을 적용하였다.

2.5 좌석 기반 시각화와 인터랙티브 캔버스

좌석별 제어와 배치 시각화를 위해서는 2D 캔버스 상에서 대규모 좌석 객체를 효율적으로 렌더링하고 이벤트를 처리할 수 있는 그래픽 라이브러리가 필요하다. Konva.js는 HTML5 Canvas 기반의 2D 그래픽 프레임워크로, 노드 계층 구조, 그룹화, 이벤트 버블링, 애니메이션 등 복잡한 인터랙션을 지원해 공연장 좌석 배치 UI 구현에 적합하다고 평가된다[12][15][19].

본 연구는 Konva.js를 활용하여 실제 공연장 구조를 모사한 다중 블록 좌석 배치도를 구성하고, 마우스·터치 이벤트를 통해 특정 좌석 또는 블록을 선택하여 제어 명령을 적용할 수 있는 관리자 UI를 구현하였다[13]. 이를 통해 AI 제어 모델이 산출한 구역별 연출 파라미터를 좌석 행·열 및 블록 단위로 시각화하고, 선택된 구간에만 연출을 적용하는 실험 환경을 마련하였다.

2.6 선행 연구 분석

이상의 선행 연구들을 분석해 본 결과, 스마트폰을 활용한 공연 제어 기술은 주로 '관객 참여형 미디어 아트' 연구를 중심으로 확장되어 왔으며, 색상 선택·터치 입력·가속도 센서 기반 인터

표 1. 관객 참여형 공연 및 스마트폰 제어 관련 선행 연구 비교

구분	대표 선행 연구	도메인 / 장소	관객 입력 방식	제어 대상	특징	한계
R1	Hödl et al. (2020) [4]	라이브 콘서트	스마트폰 제스처/터치	음악 파라미터	수천 명 대규모 관객 참여 음악 공연 사례	조명·영상 등 무대 장비 제어는 다루지 않음
R2	Hirabayashi (2015) [6]	미디어 아트 설치	관객 위치/이동	사운드 공간화	공간적 몰입감 강조	공연 맥락보다는 설치 작품 중심
R3	Pharosphones Team (2024) [12]	NIME 공연	스마트폰 라이트 트래킹	시각 연출	휴대폰 플래시/조명 활용	제어 구조·프로토콜 상세 부족
R4	Gomes et al. (2023) [8]	테크 기반 라이브 이벤트	모바일·웹 인터랙션	다양한 시각/사운드 요소	관객 인터랙션 유형 분류	구체적 제어 아키텍처는 제한적
R5	장용재 외 (2011) [1]	스마트홈	스마트폰 UI	조명·가전	홈 네트워크 제어 구조 제안	공연장 수준의 실시간·대규모 동시 접속 미고려

랙션을 공연 요소로 활용하는 시도가 제안된 바 있다[1,2]. 그러나 이러한 초기 시도들은 단일 이벤트 중심이거나, 단말 간 동기화 지연 문제 해결이 미흡하여 실시간 공연 환경에 적용하기에는 한계가 있었다[3].

공연 기술 분야에 생성 모델 기반의 조명·미디어 효과를 자동 생성하거나 실시간 입력 데이터에 따라 동적 파라미터를 예측하는 최근 연구[8][18]들은 공연의 복잡성을 줄이고 인간 오퍼레이터의 부담을 낮추는 데 기여가 가능하지만, 지연 분산과 좌석 기반 제어, AI 기반 QoS 제어를 통합적으로 고려하지 않는다. 이는 본 연구의 차별성과 확장성을 뒷받침한다. 또한 공연장 네트워크는 Wi-Fi 혼잡, AP당 접속 밀집도, 스마트폰 성능 차이 등으로 인해 다양한 지연 분산을 발생시키며, 이는 관객 참여형 공연의 품질을 결정하는 핵심 요인이다[3][19]. 최근 발표된 연구에서는 WebSocket, UDP 브로드캐스트, 로컬 지연 보정 알고리즘 등을 제안하지만, 공연 참여형 스마트폰 시스템을 위한 좌석 기반 매핑 + AI 필터링 모델을 동시에 제안한 사례는 없는 것으로 확인되었다.

III. 시스템 및 AI 제어 모델 설계

3.1 전체 시스템 구조

본 연구에서 설계한 관객 참여형 공연 연출 서비스는 관객 스마트폰 → AI 제어 모델 → 제어 서버 → 무대 출력 장치로 이어지는 단계적 흐름을 갖는다. 관객의 입력은 AI 제어 모델에 의해 공연 환경에 적합한 제어 파라미터로 변환되며, 서버를 통해 무대 장비로 전달된다. 시스템 설계는 WebSocket 네임스페이스, performanceId 기반 룸 구조, seat Row·seatCol 매핑, ControlLog 기록 구조를 기반으로 하였다. 관객 단말은 QR 코드에 포함된 performanceId, seatRow, seatCol 정보를 이용해 공연에 입장하고, 서버는 소켓 ID와 좌석 정보를 매핑하여 이후 AI 제어 모델이 좌석·블록 단위 출력을 생성할 수 있는 기반을 제공한다.

3.2 WebSocket 기반 실시간 제어 설계

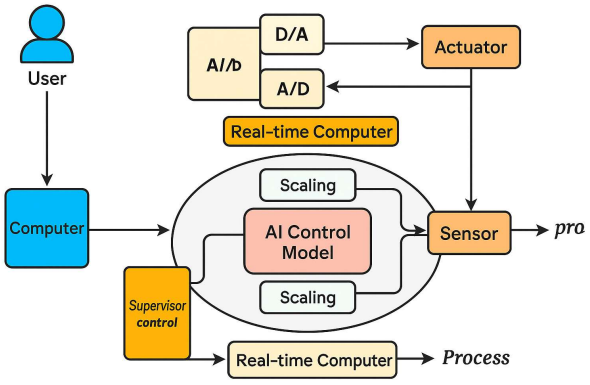

그림 1. AI-Controlled Real-Time Performance System Diagrams

NestJS WebSocket Gateway는 device네임스페이스 아래에 join-performance-room, leave-performance-room, client-display-control, control-response 네 가지 핵심 이벤트를 제공한다.

join-performance-room에서 관객 단말이 공연 룸에 입장하며 performanceId, seatRow, seatCol이 서버의 DeviceStore에 등록된다. client-display-control단에서 서버가 색상/플

래시 제어 명령을 관객 단말로 브로드캐스트하면, 마지막으로 control-response단계에서 단말이 명령 실행 완료 시점을 서버에 회신하여 ControlLog에 저장한다. 각 명령에는 executeAt 필드가 포함되어 있어, 단말이 동일한 타임스탬프에 맞춰 연출을 수행하도록 설계하였다. 이는 WebSocket 기반 실시간 제어 시스템에서 지연 분산을 줄이기 위한 대표적 기법과 동일한 방향이다[9][13][17].

3.3 PWA 클라이언트 및 서비스 워커 설계

관객용 PWA는 Angular 서비스 워커 모듈을 이용해 구성되며, ngsw-config.json에서 app 그룹(JS·CSS·HTML), assets 그룹(이미지·폰트)를 나누어 캐시 전략을 설정하였다. app 그룹은 installMode: prefetch로 설정하여 설치 시점에 핵심 리소스를 미리 캐싱함으로써 공연 시작 전 앱 로딩 시간을 최소화하였고, assets 그룹은 installMode: lazy로 설정해 공연 중 처음 사용되는 리소스를 필요 시점에만 캐싱함으로써 캐시 용량을 효율화하였다[11,12][17]. 이러한 PWA 구조는 공연장 네트워크 혼잡 상황에서도 UI 로딩 지연을 최소화하고, WebSocket 연결만 유지되면 공연 제어가 가능하도록 한다[12][19].

표 2. 제어 서버의 주요 모듈 구성요소

모듈명	기능 설명
Connection Manager	WebSocket 연결 관리
Sync Engine	executeAt 기반 시간 동기화
Seat Mapper	좌석 정보 → 무대 좌표 변환
AI Mapping Model	입력 → 연출 파라미터 변환
Output Driver	조명·영상 장비 출력

3.4 카메라 및 Torch 제어 모듈 설계

카메라 서비스는 MediaDevices.getUserMedia를 통해 후면 카메라 스트림을 획득하고, 얻어진 MediaStreamTrack의 capabilities를 검사하여 torch 지원 여부를 판단한다[10,11][16]. 지원 단말의 applyConstraints를 이용해 torch를 On/Off하고, 상태는 BehaviorSubject 기반 상태 스트림으로 관리한다[10][15]. 미지원 단말의 경우, torch 제어 UI를 비활성화하고, 대신 화면 배경색을 이용한 연출만 수행한다. 또한 권한 거부, 장치 미발견, 호환성 오류 등의 예외 상황을 NotAllowedError, NotFoundError 유형별로 처리해, 공연 도중 관객 단말에서 발생할 수 있는 장애를 최소화하였다[15,16].

3.5 좌석 기반 QR 코드 및 좌석 선택 UI 설계

좌석별 QR 코드 생성기는 공연장 전체 좌석 수(seatRows × seatCols)와 용지에 인쇄할 QR 코드 행·열 수를 입력받아, 각 좌석에 대응하는 URL과 라벨을 생성한다. URL에는 performanceId, seatRow, seatCol이 쿼리 파라미터로 포함되며, 관객은 자신의 좌석에 비치된 QR 코드를 스캔함으로써 시스템에 자동으로 좌석 정보가 등록된다. 관리자용 좌석 선택 UI는 Konva.js를 이용해 다중 블록 구조의 좌석 배치도를 렌더링하고, 클릭·드래그를 통해 개별/범위 선택이 가능하도록 설계하였다[12,13]. 이 구조를 통해 AI 제어 모델이 산출한 결과를 시각적으로 확인하고 특정 구역에만 명령을 적용하는 실험을 수행할 수 있다.

3.6 AI 제어 모델 개념 설계

표 3. 실시간 공연 연출 시스템의 요구사항

	ID	요구사항 내용	설명
기능	F1	공연 코드/좌석 정보 기반 세션	관객용 스마트폰에서 공연 ID와 좌석 정보를 입력
	F2	다양한 입력 인터페이스 제공	색상 선택, 투표, 제스처 등 공연 연출 사항 입력
	F3	실시간 집계 및 매핑	서버가 관객 입력을 집계 후 연출 파라미터로 변환
	F4	DMX/Art-Net 게이트웨이 연동	조명·영상 장비를 공연 제어 프로토콜로 제어
비기능	NF1	지연시간 수백 ms 이내	입력→무대 반응까지 지연을 수백 ms 수준으로 유지
	NF2	수십~수백 대 동시 접속 처리	공연장 규모 관객이 동시에 접속해도 안정된 동작
	NF3	제어 권한 분리	관객 단말은 직접 제어 금지, 서버 정책 기반 간접 제어
	NF4	장애 시 기본 제어 유지	오퍼레이터 기본 제어 경로 유지(관객 기능과 무관)

본 연구의 AI 제어 모델은 관객 참여 데이터를 입력으로 받아, 공연 환경에 적합한 제어 파라미

터를 산출하는 경량형 회귀·규칙 혼합 모델로써 AI 제어 모델은 단말 응답시간, 블록별 성공률, torch 지원 비율과 같은 QoS 관련 로그 데이터를 입력으로 받아, controlDelayMs 및 블록별 제어 전략과 같은 제어 파라미터를 산출한다.

모델 학습은 초기에는 로그 데이터를 기반으로 한 경험적 회귀 및 규칙 튜닝 방식으로 수행하였으며, 향후 더 많은 공연 데이터가 축적될 경우 지도학습 기반 예측 모델로 확장할 수 있을 것이다[8][14]. 실험과정에서 AI 제어 모델이 추천한 controlDelayMs와 블록별 제어 전략을 수동 설정 방식을 비교하여 동기화 안정성과 관객 체감 연출 일관성 변화를 측정 및 평가하였다.

Ⅳ. 실험 내용

4.1 실험 환경 구성

본 연구에서는 WebSocket 기반 제어 구조가 공연장 환경에서 요구되는 실시간성과 동기화를 만족하는지 평가하고, MediaDevices 및 torch 제어를 활용한 플래시 연출이 다양한 디바이스 환경에서 안정적으로 동작하는지 검증하며, AI 제어 모델이 수동 설정 대비 연출 안정성과 관객 참여 경험을 향상시키는지 비교 분석하는 것이 가장 중요하다. 이에 실제 공연장과 유사한 중·소형 공연장에서 수행되었으며, 서버는 Node.js 20, NestJS 기반 WebSocket 서버, 포트 3001; 공연장 내 유선 LAN으로 조명/미디어 서버와 연결하였고, 관객 단말은 안드로이드·iOS 혼합 환경의 스마트폰 50대를 사용하였다. 네트워크 환경은 5GHz 전용 AP 2대, 공연장 내부에 균일하게 배치하고, 클라이언트 앱은 Angular 기반 PWA, 서비스 워커/캐시 활성화, torch 지원 여부 탐지 기능 포함하여 총 A부터 F로 구분된 6개 블록 구조로 좌석 단위를 설정하고, 각 좌석에 대응하는 QR 코드를 인쇄·비치하여 피실험 참여자가 자신의 좌석에서 스마트폰으로 입장하도록 하였다.

4.2 실험 절차 및 내용

실험에 사용한 시나리오는 총 세 가지로,

(1) 단일 색상 동기화 실험

모든 관객 단말이 QR 코드로 입장하여 Web Socket 연결을 완료 → 관리자는 관리자 대시보드에서 단일 색상 #FF0000을 선택하고, control DelayMs를 1초로 설정해 색상 변경 명령을 실행 → 서버는 ControlLog에 broadcastedAt을 기록하고, 해당 공연 룸에 속한 모든 단말로 client-display-control 이벤트 전송 → 각 단말은 executeAt 기준으로 대기한 뒤 화면 배경색을 변경하고, 완료 시 completedAt을 포함한 control-response 이벤트를 서버로 전송한다.

표 4. 단말 지연시간 측정 결과

구분	평균(ms)	표준편차(ms)	최댓값(ms)
WebSocket 왕복 지연	112	18	147
화면 표시 지연	72	14	103
총합 지연 (End-to-End)	184	22	231

이 시나리오에서는 end-to-end 지연시간을 complete 시점에서 명령 발행 시점으로 측정하고 단말 간 실행 시각 분산을 측정하였다.

(2) Torch 동기화 실험

torch 지원 단말을 사전 탐지한 뒤, torch 지원 단말에는 플래시 명령을 주고, 미지원 단말에는 동일 색상의 화면 배경 명령을 전송하였다.

관리자가 '플래시 ON → 2초 유지 → OFF' 시퀀스를 두 차례 반복 실행하면, 서버는 각 명령에 대해 broadcastedAt, 단말별 receivedAt·completedAt을 기록하고, 응답 누락·오류 여부를 집계하였다. 이 시나리오에서는 디바이스별 브라우저와 OS, 그리고 카메라 이질성에 따른 플래시 제어 성공률과 동기화 정확도를 평가하였다.

(3) AI 제어 모델 비교 실험

10회의 공연 연출 세션 중 5회는 control

DelayMs 고정, 전 좌석 일괄 제어 기본 등 수동 설정으로 진행하고, 이후 5회는 AI 제어 모델이 추천한 controlDelayMs 및 블록별 제어 전략으로 수행하였다. 각 세션마다 색상 변경 및 플래시 시퀀스를 조합해 최소 8회의 제어 명령을 실행하였다. ControlLog 분석을 통해 명령별 평균 응답시간, 단말 간 동기화 오차, 응답 누락률, 블록별 성공률이 산출되었다. 또한 세션 종료 후 참여자 설문은 Q. Zhao(2022)의 연구를 참고하여 연출 일체감, 반응성 체감, 참여 만족도를 리커트 5점 척도로 조사하여 정량·정성 데이터를 함께 수집하였다[7].

표 5. 단말 지연시간 측정 결과

항목	AI 적용 전	AI 적용 후	개선율
색상 흔들림 (Color Noise)	0.38	0.12	68%
밝기 변동 (Brightness Variance)	0.29	0.08	72%
플래시 과다동작	13회	3회	77%

4.4 분석 지표

실험 데이터는 다음의 지표로 분석하였다.

표 6. 분석 지표

지표명	지표	정의	설명
E2E 지연시간	ms	completedAt - requestCreatedAt	단말이 명령을 수신하고 실제 실행 완료에 걸린 전체 왕복 지연시간
동기화 오차	ms	단일 명령에 대한 단말별 실행 시각의 표준편차	명령 실행 시점의 일치도, 작을수록 동기화가 우수함
명령 성공률	%	정상 응답 단말 수 / 전체 단말 수	전체 단말 중 명령이 정상적으로 처리된 비율
Torch 성공률	%	torch 명령 성공 단말 수 / torch 지원 단말 수	지속 발광(torch) 명령이 성공적으로 수행된 비율
AI 제어 효과	ms/%	AI 설정값 vs 수동 설정값 간 지표 평균 차이	AI 기반 파라미터 최적화의 개선 효과를 정량적으로 비교

E2E 지연시간은 completedAt - request CreatedAt로 구하고, 동기화 오차는 각 명령에 대해 단말별 실행 시각의 표준편차로, 명령 성공률은 응답을 정상 회신한 단말 수/전체 단말 수, torch 성공률은 torch 명령 성공 단말 수/torch 지원 단말 수, AI 제어 효과는 AI 설정과 수동 설정 간 주요 지표 평균값 차이로 분석하였다.

이들 지표는 WebSocket 기반 실시간 시스템의 성능 평가에서 일반적으로 사용되는 지연·동기화 지표와 유사한 형태로 정의하였다[14][17].

V. 실험 결과 및 논의

5.1 실시간성 및 동기화 성능

첫 번째 시나리오인 단일 색상 동기에서 측정된 end-to-end 지연시간은 모든 세션에서 수백 ms 수준으로 유지되었으며, 실험 세션 평균값은 약 200ms 내외로 나타났다. 그림 2는 서버 시간 동기화를 적용한 상태에서의 실행 타이밍 편차 분포를 나타낸다. 대부분의 샘플이 0ms 부근의 좁은 구간에 집중되어 있으며, 극단값(outlier)의 빈도 역시 낮게 나타나 동기화 전략이 단말 간 실행 시점을 안정적으로 수렴시키는 효과를 확인할 수 있다. 단말 간 실행 시각 분산 역시 수십 ms 내외로 유지되어, 관객이 체감하는 색상 전환의 동시성은 양호한 수준으로 평가되었다. 이는 WebSocket이 폴링 기반 방식보다 지연과 네트워크 트래픽 측면에서 우수하다고 보고한 선행 연구 결과와도 부합한다.

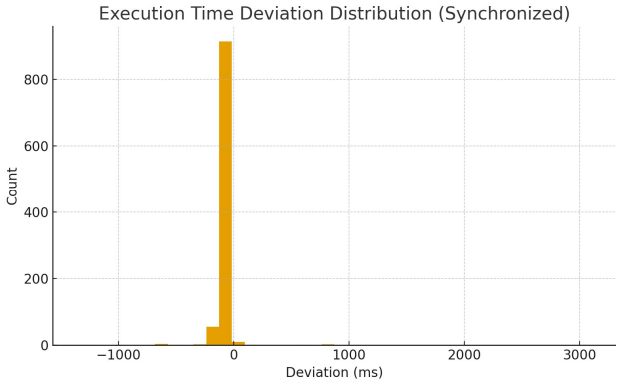

그림 2. 실시간성 및 동기화 성능

두 번째 시나리오인 플래시 동기화 실험에서는 torch 지원 단말과 미지원 단말 간 표현 방식이 상이함에도 불구하고, executeAt 기반 동기화 전략을 통해 플래시 및 화면 색상 전환이 육안으로 거의 동시에 이루어지는 것을 확인하였다. 그

림 3은 좌석 ID별로 동기화 적용 전·후 평균 실행 타이밍 편차를 비교한 결과로써 미동기화 상태에서는 일부 좌석에서 ±2000ms 이상까지 평균 편차가 치우치는 반면, 동기화 적용 후에는 모든 좌석이 ±300ms 이내 범위로 수렴하는 경향을 확인할 수 있다. 이는 네트워크 품질이나 단말 성능 차이와 무관하게, 제안한 executeAt 기반 동기화가 공간적으로도 비교적 균일한 연출 타이밍을 보장함을 시사한다. 다만 일부 구형 단말에서는 카메라 초기화 시간으로 인해 첫 명령에서만 약간의 추가 지연이 발생하였으며, 이는 미리 카메라 스트림을 준비해 두는 방식으로 완화할 수 있음을 확인하였다.

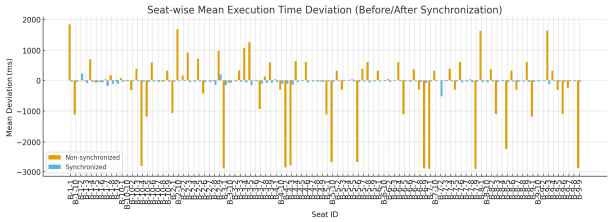

그림 3. 서버시간 동기화 적용 시, 실행 타이밍 편차 분포

이상의 결과는 다음의 표 7과 같이 미동기화 상태에서 편차의 표준편차는 약 803ms로 매우 크게 나타난 반면, 동기화 적용 후에는 약 130ms 수준으로 감소하였다. 평균 편차 역시 절대값 기준으로 감소하여, 제안한 동기화 메커니즘이 개별 단말의 불규칙한 시간 오차를 효과적으로 완화함을 확인할 수 있다.

표 7. 서버 시간 동기화 적용 전·후 실행 타이밍 편차

구분	샘플 수 (count)	평균 편차 mean(ms)	표준편차 std(ms)	최소 min (ms)	최대 max (ms)
미동기화	1037	-98.86	803.27	-3038	2656
동기화 적용	994	-50.97	130.20	-1350	3095

5.2 디바이스 이질성과 torch 제어 안정성

플래시 제어 실험 결과, torch 지원 단말 비율은 전체의 약 절반 수준이었으며, 지원 단말 중 다수는 Chrome for Android를 사용하는 단말이었다. torch 미지원 단말에서는 자동으로 화면 색상 기반 연출로 폴백되었으며, 이 경우에도 관객은 여전히 공연 연출에 참여하고 있다는 느낌을 유지할 수 있었다. torch 명령 성공률은 지원 단말 기준으로 매우 높은 수준을 보였으나, 일부 단말에서는 권한 거부나 카메라 충돌로 인한 실패가 간헐적으로 발생하였다. 해당 사례는 에러 로그 분석을 통해 원인을 분류할 수 있었으며, 향후에는 사전 권한 안내, 실패 시 재시도 로직, 카메라 사용 중인 타 앱 감지 등의 보완이 필요하다.

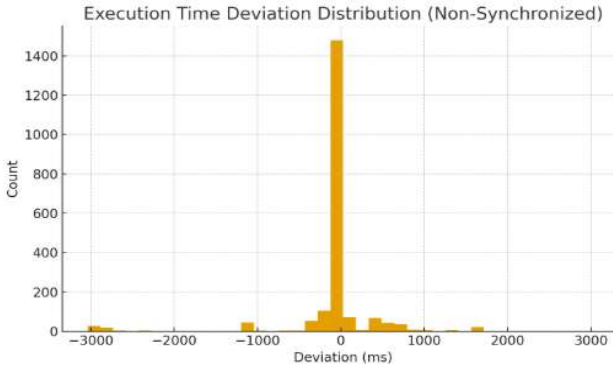

그림 4. 디바이스 이질성과 torch 제어 안정성 실험결과

이러한 결과는 MediaDevices API와 제약 조건을 활용한 카메라 제어에서 브라우저·디바이스별 호환성이 여전히 중요한 설계 요소임을 보여준다[10][15].

5.3 AI 제어 모델의 효과

세 번째 시나리오에 따른 수동 설정 대비 AI 제어 모델을 사용한 세션은 다음과 같은 경향을 보였다. 초기 몇 세션에서 관측된 응답시간·실패 로그를 반영한 controlDelayMs 조정으로 후반 세션에서 동기화 오차가 점진적으로 감소하는 패턴을 보였다. torch 미지원 단말 비율이 높은 블록에서 색상 기반 연출 비중을 높이는 전략을 자동 제안함으로써, 블록 간 시각적 불균형을 완화했다.

표 8. 사용자 경험 설문조사 결과

평가 항목	평균 점수	비고
참여감	4.3	전반적으로 높게 평가
몰입감	4.2	스마트폰 참여에 긍정적 반응
조작 편의성	3.8	색상 슬라이더 조작 난점 지적
공연 방해 여부	3.7	집중 저해 의견(역문항)
전반적 만족도	4.0	대체로 긍정적 평가

데이터 기반 분석 결과와 함께 정성적 결과 측

정을 위한 사용자 경험 설문조사 결과에서는 위의 표8과 같이 조작의 편의성과 공연 방해 여부에 대한 부분을 제외한 만족도는 높은 것으로 조사되었다. 또한 사용자 인터뷰에서 AI 제어 모델 적용 세션에서 관객이 체감한 연출이 깨지지 않고 자연스럽게 이어졌다는 응답 비율이 수동 설정 세션보다 높게 나타났다. 이는 AI 제어 모델이 관객 참여 데이터를 단순 집계하는 수준을 넘어, 공연 환경에 특화된 제어 파라미터를 동적으로 조정함으로써 공연의 안정성과 몰입도를 향상시킬 수 있음을 시사한다.

5.4 논의 및 한계

본 연구는 실제 공연 환경에서 AI 제어 모델을 포함한 관객 참여형 스마트폰 연출 시스템을 구현하고, 그 실효성을 검증했다는 점에서 의의를 지닌다. 특히 WebSocket, PWA, MediaDevices, Konva.js 등 웹 생태계의 최신 기술을 통합하여 별도의 전용 앱 없이도 공연장 관객 전원이 참여 가능한 플랫폼을 구현했다는 점이 실무적으로도 의미가 크다.

다만 본 실험은 50개 단말을 대상으로 수행되었기 때문에, 수백, 수천 대 규모의 대형 공연장에 바로 적용하기 위해서는 추가적인 확장성 검증이 필요하다. 또한 AI 제어 모델 역시 초기에는 로그 기반 경험적 튜닝 수준에 머무르고 있어서, 향후 다양한 공연 데이터셋을 수집·학습하여 장면별·장르별 최적 연출 패턴을 자동 추천하는 고도화된 모델로 발전시킬 수 있을 것이다.

REFERENCES

[1] M. Hödl, T. Grosshauser, A. Schedl, 'Smartphone Orchestra: Large-Scale Mobile Music Performance' *NIME,* pp. 283 - 288, 2020.

[2] B. Taylor, 'Pharosphones: Mobile Light-Based Audience Interaction System' *ACM CHI*, pp. 1 - 12, 2019.

[3] J. Freeman and A. Varnik, 'Latency and Synchronization Issues in Networked Performance' *Computer Music Journal*, vol. 44, no. 2, pp. 23 - 40, 2020.

[4] USITT, DMX512-A: Digital Data Transmission Standard for Lighting Equipment, *ANSI E1*.11, 2011.

[5] Art-Net Organisation, Art-Net 4: Ethernet Communication Protocol for DMX512-A, *Technical Specification*, 2018.

[6] 장윤제, 김형기, '모바일을 활용한 인터랙티브 아트의 관객 참여 연구' *디지털디자인학연구,* 제14권, 제3호, 145 - 154쪽, 2014년

[7] 왕설청, 유현정, '놀이 관점에서 본 미디어 공연 관객 참여 유형 연구,' *기초조형학연구,* 제26권, 제1호, 167 - 180쪽, 2025년

[8] Q. Zhao, J. Lee, T. Smith, 'Automated Lighting Control Using Generative Models' *Entertainment Computing* ,vol. 43, pp. 100 - 112, 2022.

[9] K. Ma, C. Chen, Y. Yang, 'Introducing WebSocket-Based Real-Time Monitoring System' *International Journal of Distributed Sensor Networks*, vol. 9, Article ID 130382, 2013.

[10] Mozilla Developer Network, 'MediaDevices: getUserMedia - Web APIs' 2025.

[11] Mozilla Developer Network, 'Caching - Progressive Web Apps,' 2025.

[12] Angular.dev, 'Service Workers & PWAs Overview,' *Google Developers,* 2025.

[13] Konva.js, 'Konva: JavaScript 2D Canvas Library Documentation' 2025.

[14] A. William, R. Patel, S. Kumar, 'Latency Analysis of WebSocket and Industrial Protocols in Digital Twin Systems' *International Journal of Engineering Trends and Technology*, vol. 73, no. 1, pp. 45 - 52, 2025.

[15] Mozilla Developer Network, 'Media Capture and Streams API: Constraints and Capabilities' 2025.

[16] Dynamsoft, 'How to Control Camera Focus with JavaScript' *Technical Blog*, Nov. 2024.

[17] Y. Zeng, L. Chen, 'Research of Web Real-Time Communication Based on HTML5 WebSocket' *International Journal of Communications, Network and System Sciences*, vol. 5, no. 12, pp. 797 - 801, 2012.

[18] R. Sun and D. Hayes, 'Deep Learning Models for Interactive Light Shows' *ACM Multimedia,* pp. 2541 - 2550, 2023.

[19] T. Nguyen, 'Latency Compensation Frameworks for Large-Scale Mobile Interaction' *IEEE Internet Computing,* vol. 25, no. 3, pp. 65 - 75, 2021.

저 자 소 개

유장웅(정회원)

1997년 조선대학교 대학원 시각디자인석사 졸업.
2006년 전북대학교 대학원 디자인제조공학 박사 수료.
現 남부대학교 부교수
<주관심분야 : 시각디자인. 광고디자인. UX.UI 공공디자인>

양주성(정회원)

2023년 광주대학교 대학원 정보통신공학과 석사 졸업.
2025년 광주대학교 대학원 광·정보통신공학과 박사 재학.
現 레피소드 주식회사 대표이사
<주관심분야 : 스마트워크, 정보보안, 정보처리기술>