



## [우수논문] Android 기반의 다중 기기에서의 동영상 동시 재생을 위한 동기화 기법

A Synchronization Technique for Android Mobile Multivision Application

---

저자 (Authors) 김강현, 윤준호, 이법재, 김대영  
Ganghyeon Kim, Junho Yun, Bupjae Lee, Daeyoung Kim

출처 (Source) [한국정보과학회 학술발표논문집](#), 2014.6, 338-340 (3 pages)

발행처 (Publisher) [한국정보과학회](#)  
KOREA INFORMATION SCIENCE SOCIETY

URL <http://www.dbpia.co.kr/Article/NODE02444018>

APA Style 김강현, 윤준호, 이법재, 김대영 (2014). [우수논문] Android 기반의 다중 기기에서의 동영상 동시 재생을 위한 동기화 기법. 한국정보과학회 학술발표논문집, 338-340.

이용정보 (Accessed) 한국과학기술원  
143.248.53.234  
2016/01/07 13:44 (KST)

---

### 저작권 안내

DBpia에서 제공되는 모든 저작물의 저작권은 원저작자에게 있으며, 누리미디어는 각 저작물의 내용을 보증하거나 책임을 지지 않습니다.

이 자료를 원저작자와의 협의 없이 무단게재 할 경우, 저작권법 및 관련법령에 따라 민, 형사상의 책임을 질 수 있습니다.

### Copyright Information

The copyright of all works provided by DBpia belongs to the original author(s). Nurimedia is not responsible for contents of each work. Nor does it guarantee the contents.

You might take civil and criminal liabilities according to copyright and other relevant laws if you publish the contents without consultation with the original author(s).

# Android 기반의 다중 기기에서의 동영상 동시 재생을 위한 동기화 기법

김강현<sup>○</sup>, 윤준호, 이법재, 김대영

한국과학기술원 소프트웨어대학원

{gh0212, junho.yun, bupjae, gatorkim}@kaist.ac.kr

## A Synchronization Technique for Android Mobile Multivision Application

Ganghyeon Kim<sup>○</sup>, Junho Yun, Bupjae Lee, Daeyoung Kim

Software Graduate Program, KAIST

### 요 약

스마트폰과 태블릿 같은 스마트 기기의 사용자들은 멀티미디어 콘텐츠를 크게 보고 싶어한다. 그래서 제조사들은 스크린의 크기를 키우고 있으나 이것은 어느 정도 한계가 있다. 이를 해결하기 위해 여러 대의 스마트 기기를 이용하여 멀티비전을 구성할 수 있지만 이때는 다수의 기기 상호간에 발생하는 시간지연에 의해 영상 및 소리의 불일치가 발생할 수 있다. 본 논문에서는 스마트 기기로 멀티비전을 구성할 때 발생하는 시간 지연을 확인하고, 시간 지연을 최소화 시킬 수 있는 동기화 기법을 제안한다. 이 기법은 한 대의 재생 시간을 기준으로 하여 연결된 각각의 기기가 자신의 재생 시간을 반복 조정하여 재생의 동기화가 이루어 지게 하는 것이다. 이 기법을 사용하면 네트워크 품질이나 제조사 기기의 성능 차이 때문에 발생할 수 있는 시간 지연을 보정하여 동기화된 멀티미디어 콘텐츠를 제공할 수 있기 때문에 다양한 Android 기반의 기기들을 이용하여 멀티 비전을 구성할 수 있을 것으로 기대된다.

### 1. 서 론

스마트폰과 태블릿 같은 스마트 기기의 전면 부는 스크린이 대부분을 차지하고 있으며, 최근에는 베젤<sup>1</sup> 역시 점점 좁아지고 있다. 이것은 기기 크기의 제약 속에서 콘텐츠를 최대한 크게 보려고 하는 사용자의 Needs 를 반영하고 있는 것이라고 볼 수 있다. 하지만 스마트 기기는 휴대성을 기본적으로 고려해야 하기 때문에 기기 또는 스크린의 크기를 무한정 늘릴 수는 없다. 그렇기 때문에 다수의 스마트 기기를 멀티비전<sup>2</sup>으로 구성할 수 있다면 스크린의 크기는 구성하는 기기의 수만큼 커질 수 있게 되어 넓은 화면으로 콘텐츠를 감상할 수 있게 된다. 하지만 다수의 기기를 연결하게 되면 기기간 발생하는 시간 지연에 따른 영상 불일치가 발생할 수 있게 된다. 시간 지연은 기기 별 소프트웨어/하드웨어의 차이로 발생할 수 있으며 네트워크 상의 전송 지연과 전송에러에 의해 발생할 수 있다. 네트워크 상의 전송 지연과 에러에 의한 지연을 제거하기 위해 연구[1]에서는 SNTP 를 참고하여 VOD 서비스에서 그룹 간의 동기화 이슈를 해결하는 방법을 제안하고 있으나 이는 기기 별 차이점에 대해서 고려하지 않고 있다. 연구[2]는 DLNA 기술 위에서 동작하기 때문에 DLNA 를 사용하지 않는 시스템에서는 사용할 수 없는 단점이 있다. N 개의 Android 기기를 연결하여 재생하는

기술[3]은 해당 제조사의 Group Play SDK 위에서 동작하기 때문에 적용할 수 있는 기기의 종류에 제약이 있다. 본 연구에서는 N 개의 스마트 기기로 멀티비전을 구성할 때 발생하는 네트워크 상의 시간 지연 및 기기의 특성에 따른 시간지연을 고려하여 동기화를 항상 시킬 수 있는 방법을 제안하려고 한다.

### 2. Mobile Multivision System

시간 지연을 확인하고 제안하는 알고리즘의 성능을 측정하기 위해 구현된 Mobile Multivision System 은 동영상의 재생 및 정지 제어를 하는 Controller 와 동영상을 보여주는 2 개<sup>3</sup>의 Viewer 로 구성할 수 있으며 그림 1 의 구조를 가진다. 시스템은 재생 기능을 하는 Controller 와 Viewer, 재생에 대한 동기화 알고리즘이 동작하는 Media Sync, 통신을 주고 받는 Interface 역할을 하는 Communicator, Wi-Fi Direct, Bluetooth, NFC 등의 통신 기술 모듈이 있는 Physical Communicator 를 포함한 Layered Architecture 형태로 구성된다.

<sup>1</sup> 기기의 스크린과 기기의 테두리 사이의 좁은 공간

<sup>2</sup> 하나의 영상을 확대시키거나 축소할 수 있고 이것을 다수대의 영상표시기기 등 영상 출력 수단을 연결디스플레이 하는 시스템

<sup>3</sup> 실험은 Viewer 를 2 개를 구성하였지만 Viewer 는 동일한 기능을 가지기 때문에 N 개로 추가가 가능하다.

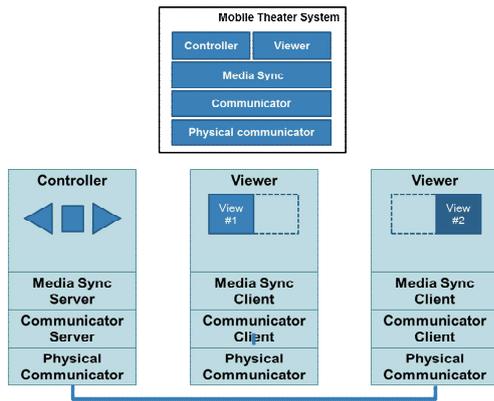


그림 1. System Architecture

동일한 소프트웨어를 가지고 사용자의 설정으로 어느 기기나 Controller 와 Viewer 의 역할을 자유롭게 변경할 수 있도록 하며 기기의 역할을 개념적으로 구분하기 위해 Server 와 Client 라는 명칭을 사용한다. 통신 단을 Communicator 와 Physical Communicator 로 분리한 이유는 현재 구현된 시스템에서는 wireless 환경에서 사용할 수 있는 통신 기술 중 가장 빠른 방법인 Wi-Fi Direct 기술을 사용하고 있는데 이 기술은 많은 데이터를 고속으로 전송할 수 있지만 전력 소모가 많기 때문에 적은 양의 데이터를 가지고 제어가 가능할 경우에 저전력으로 사용할 수 있는 Bluetooth, NFC 같은 통신 기술로 쉽게 변경하기 위해서이다.

### 3. 시간 지연의 원인 분석

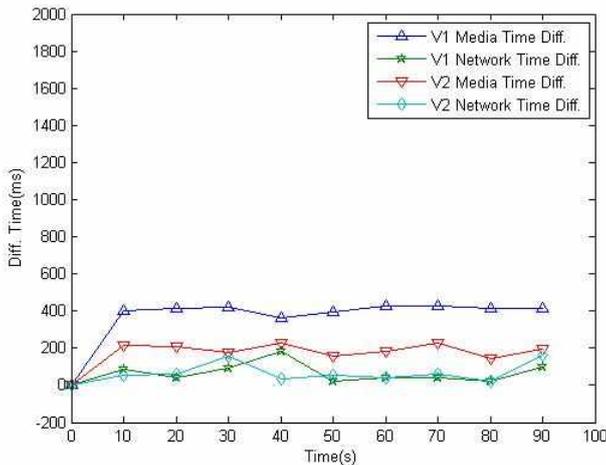


그림 2. Controller와 각 Viewer의 시간 차이

동기화 방법을 적용하지 않고 Controller 에서 재생 명령을 Broadcast<sup>4</sup> 방식으로 전달했을 때 Viewer 와 Controller 사이에서 발생하는 Network delay 및 Media Play 시간 차이는 그림 2 와 같다. 실험에 사용된 스마트 기기(V1, V2, Controller)<sup>5</sup> 는 다른 제조사의 Android 스마트폰으로

<sup>4</sup> Controller 가 송신한 메시지를 모든 Viewer 가 동시에 수신하는 방식

<sup>5</sup> V1, Controller : LG Optimus LTE

구성하였으며 실험 결과를 보면 기기간의 Network Time Diff<sup>6</sup>. 는 유사하지만, Media Time Diff.<sup>7</sup> 에서 차이가 나기 때문에 두 개의 Viewer 간의 영상이 어긋나게 재생 된다는 것을 알 수 있다. 이는 네트워크 환경과 상관없이 기기의 특성으로 인해 Media Time 의 시간 차가 발생한다고 볼 수 있으며 영상의 불일치에는 Media Difference 가 Network Delay 보다 더 많은 영향을 주고 있다는 것을 알 수 있다. 그렇기 때문에 Network delay 시간을 고려하여 Network Difference 를 보정하더라도 Media Difference 의 차이가 더 크기 때문에 영상의 동기화가 이루어 지지 않는다는 것을 알 수 있다.

### 4. 제안하는 알고리즘

Media Difference 를 줄여 동기화를 이루기 위해 본 연구에서는 각각의 Viewer 가 Controller 의 재생 정보를 가지고 자신의 재생 시점을 동기화 하는 방법을 제안한다. 이것은 각각의 Viewer 가 재생명령을 받으면 주기적으로 Controller 의 재생 시간을 확인하여 자신의 재생 시점과의 차이가 THRESHOLD 시간( $\pm 50ms$ )을 벗어났을 경우 SeekTo<sup>8</sup> 나 Pause 를 이용하여 동기화 시키는 방법이다

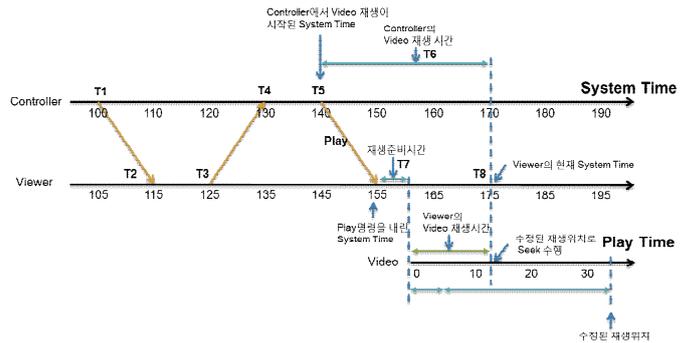


그림 3. Concept

그림 3 을 보면 Controller 에서 재생 명령을 송신한 후에 실제로 Viewer 에서 재생을 시작할 때 까지는 통신을 수신한 시점으로부터 재생 준비를 하는 시간(T7)이 발생하게 되는 것을 알 수 있다. 그렇기 때문에 각각의 Viewer 는 기기의 System 시간을 이용하여 재생 준비 시간(T7)과 실제 재생 시간(T6-T7)을 계산한 다음 수정된 위치로 재생 위치를 조정하면 Media Difference 를 줄일 수 있다.

이때 Viewer 는 자신의 재생 위치를 상대적으로 앞, 뒤로 조정해야 하기 때문에 현재 재생 위치가 전체 재생 위치보다 뒤에 있는 경우는 SeekTo 로 이동하고 현재 재생 위치가 전체 재생 위치보다 앞에 있는 경우에는 동일 위치에 올 때까지 Pause 상태로 둔다.

V2 : Samsung Galaxy S2

<sup>6</sup> Controller 의 System 시간과 Viewer 의 System 시간의 차이

<sup>7</sup> Controller 와 Viewer 각각의 Media 재생 시간의 차이

<sup>8</sup> Android MediaPlayer 클래스에서 재생 시점을 이동시키는 방법

```

1 PROCEDURE AutoSync
2   variable
3   // 변수 선언
4   BEGIN
5     IF difference of system time between Controller and this viewer = 0 THEN
6       Call AutoSync Procedure after 3 seconds;
7     ELSE
8       currentPlayTime := current play time of Media Player in viewer device;
9       currentSystemTime := current system time of viewer device;
10
11      idealPlayTime := currentSystemTime -
12      (startedSystemTimeOfVideo - difference of system time between Controller and this viewer);
13      diffTime := idealPlayTime - currentPlayTime;
14
15      elapsedTime := currentSystemTime - previousSystemTime;
16      readyTime := (previousModifiedPosition + elapsedTime) - currentPlayTime;
17
18      modifiedPosition := idealPlayTime + readyTime;
19
20      IF Media Player is not play THEN
21        start media;
22        Call AutoSync Procedure after (readyTime + 3) seconds;
23      ELSE IF THRESHOLD < diffTime THEN // idealPlayTime > currentPlayTime
24        seek to modifiedTime;
25        Call AutoSync Procedure after (readyTime + 3) seconds;
26      ELSE IF THRESHOLD < -diffTime THEN // idealPlayTime < currentPlayTime
27        pause media
28        Call AutoSync Procedure after (-diffTime) seconds;
29      ELSE
30        // Synchronized State
31      ENDIF
32
33      previousSystemTime = currentSystemTime;
34      previousModifiedPosition = modifiedPosition;
35    ENDIF
36 END PROCEDURE
    
```

알고리즘 1. Self-Sync of Each Viewer

SeekTo 와 Pause 를 함께 사용하는 이유는 Pause/Play 의 동작이 SeekTo 보다 빠르게 응답하기 때문에 명령 전달 후 지연이 상대적으로 적기 때문이다. 알고리즘 1 은 이것을 나타낸 것이다.

### 5. 실험 및 결과

제안한 알고리즘을 구현하여 Android 4.1(Jelly Bean) Version 의 삼성전자 갤럭시 S2 와 LG 전자 옵티머스 LTE 모델에서 실험을 진행 하였다. 재생 중에 시각 및 청각을 이용하여 Viewer 간의 동기화가 자연스러운지 확인할 수도 있지만 정확한 분석을 위해 Viewer 에서 사용된 network time, media time, system time 등의 데이터를 로그 형태로 저장하여 성능 평가에 사용하였다. 멀티미디어의 동기화의 성능을 측정하기 위한 기준으로 연구[1]은 Video synchronization accuracy 를 120ms(±60ms) 미만으로 user response time 은 100ms(±50ms) 이하여야 한다고 밝히고 있으며, 연구[4]는 IEEE802.1AVB 에서 제시하고 있는 요구사항인 ±80ms 를 사용하고 있는데 본 연구에서는 연구[4]의 기준을 참고하여 100ms(±50ms)를 동기화 성능의 평가 기준 시간으로 사용한다.

알고리즘 적용 시 각각의 Viewer 에서 측정한 Controller 와의 Media 재생 시간 차이(Media Time Diff.)는 그림 4 와 같다. 왼쪽의 알고리즘 적용 전의 결과를 보면 Viewer 간 미디어 재생 시간의 차이, 즉 동기화의 시간차이는 200ms(±100ms) 정도이고 V1 의 경우는 Controller 대비 400ms 정도의 지연을 보여주고 있다. 오른쪽의 알고리즘 적용 후의 결과를 보면 Viewer 간의 동기화 시간 차이가 100ms(±50ms) 미만으로 유지하고 있으며 Controller 대비 100ms 이내의 지연을 유지하고 있는 것을 확인할 수 있다.

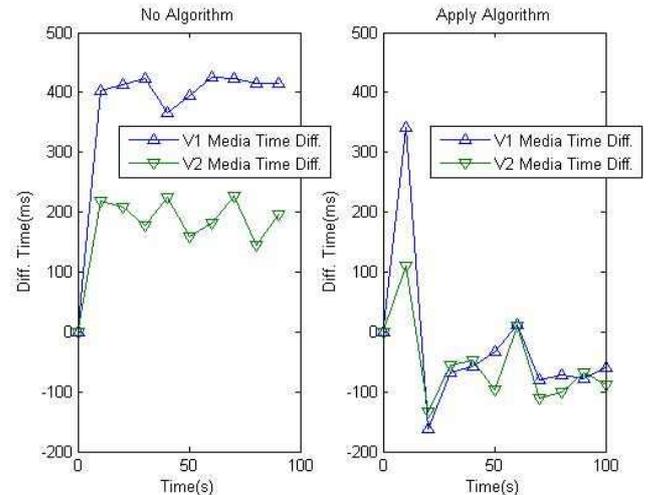


그림 4. 알고리즘 적용 후 Media 재생 시간 차이

### 6. 결론 및 향후 계획

이 논문에서는 Android 기반의 다중 기기에서의 동영상 동시 재생을 위한 동기화 기법을 제안하고 검증해 보았다. 실험 결과는 평가 기준에 부합했으며 보정을 하기 위해 발생하는 계산 부하를 생각해보면 Controller 는 자신의 시간 정보만 전송하고 각각의 Viewer 에서 모든 계산 및 제어가 이루어지기 때문에 Viewer 를 N 개로 확장했을 때 발생하는 계산 부하와 Viewer 가 2 개 일 때의 계산부하는 동일한 것이라는 것을 알 수 있다. 그렇기 때문에 본 알고리즘을 적용하면 네트워크 상태나 기기 사양에 상관없이 N 개의 기기로 멀티비전을 구성할 수 있을 것이다.

본 실험은 Android 환경에서 구현 검증하였으나 제안하는 알고리즘은 시스템의 기능에 종속적인 방법이 아니기 때문에 향후 Android 이외의 환경에서 동일한 결과를 얻을 수 있는지 확인해보도록 하겠다.

### 참고문헌

- [1] S. Moriyasu, K. Tajima, K. Ohshima, and M. Terada, "Group synchronization method with fast response time for VoD services," in *Information Networking (ICOIN), 2012 International Conference on*, pp. 182-187, 2012.
- [2] 강미란, 김세영, 김영일, and 김대진, "다중 기기에서 이어 보기를 위한 동기화 기법," *한국콘텐츠학회논문지*, vol. 11, pp. 543-551, 12 2011.
- [3] "Samsung Group Play SDK." from <http://developer.samsung.com/group-play>
- [4] B. Yonghwan, H. Jongpil, L. Kyusang, Y. Jongwon, J. Jinoo, Y. Sungbo, et al., "Wireless network synchronization for multichannel multimedia services," in *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on*, pp. 1073-1077, 2009.