卒業論文

co-Sound: Web AR によるインタラクティブな
視聴メディア及び空間同期

令和 2 年 2 月 7 日提出

指導教員　　江崎　浩　　教授
落合　秀也　准教授

電気電子工学科

03-180464　井口 和真

概要

インターネットを前提とした視聴サービスは，収録した映像音声データの IP ネットワーク化により各システム及びプロセスのソフトウェア制御を行っている．加えて収録対象へのオブジェクトベースによるアプローチは，視聴コンテンツの柔軟な再生を可能としている．一方で三次元映像投影技術の一つである拡張現実 (Augmented Reality，AR) は現実空間の要素とデジタル空間情報の両方と対話することを可能としているが，映像音声視聴環境の媒体として活用された事例は数少ない．そこで本研究では，AR を活用した音楽イベントのインタラクティブな映像音声再生アプリケーション「co-Sound」を提案する．co-Sound は，導入コストが低いウェブブラウザ上で，視聴者からの様々な入力に応じて，オブジェクト単位に構築された AR を動的にレンダリングするマルチモーダルなインターフェースとして設計された．さらに複数ユーザ間の AR オブジェクト操作をリアルタイムかつ双方向に共有することで，従来 1 対 1 に制限されていたユーザとコンテンツとの関係性を拡張し，同一 AR 空間でのユーザ同士のインタラクションを可能としている．試作したアプリケーションを実装し，AR 空間同期の性能評価および被験者からのアンケート評価を行った．複数ユーザからのオブジェクト操作を受け付ける十分な低遅延同期を確認し，WebAR を活用したインタラクティブな映像音声視聴メディアとして高い評価を得た．

目次

第 1 章　序論　1

1.1　背景 1

1.2　目的 2

1.3　本論文の構成 2

第 2 章　ディジタルメディアの現状と課題　3

2.1　三次元映像表現の多様化 3

2.2　Software Defined Media (SDM) 3

2.3　Web ブラウザ API 4

第 3 章　関連研究　7

3.1　SDM 試作システム 7

3.2　視聴メディアとしての AR 7

3.3　Collaborative AR 7

第 4 章　co-Sound: 設計と実装　9

4.1　要求事項の整理 9

4.2　設計 10

4.3　実装 11

第 5 章　実験と評価　16

5.1　定量的評価 16

5.2　主観評価 21

第 6 章　結論　24

6.1　まとめ 24

6.2　今後の課題 24

参考文献　26

図目次

2.1 SDM アーキテクチャ (Fig.1 [1]) 4

3.1 CloudAnchors における AR 空間の共有方式 8

4.1 co-Sound 設計図 11
4.2 co-Sound スクリーンショット 12
4.3 Three.js オブジェクトのシーン構造 13
4.4 co-Sound におけるピッキング処理 13

5.1 端末間通信トポロジ 17
5.2 実験 1: 通信プロトコル別 RTT 測定の結果 18
5.3 実験 2: メッセージサイズ別 RTT 測定の結果 19
5.4 実験 3: 接続端末数別 RTT 測定の結果 19
5.5 実験 4: 端末性能別 RTT 測定の結果 20
5.6 主観評価の集計結果 22

第 1 章

序論

1.1 背景

1.1.1 インターネット映像音声メディア

近年 YouTube や BiliBili といった動画配信サービスが登場し，スマートフォンやタブレットなどインターネットに接続された高性能デバイスが定着したことで，多くの人々が世界中から発信された動画にアクセスすることが可能になった．これは Sandvine が公開した 2018 年の統計レポート [2] によると，全世界インターネットトラフィックのうち動画のダウンストリームトラフィックが 57.69% を占めていることからも明らかである．また大容量通信環境が一般化し，5G とも呼ばれる第 5 世代移動通信システムや次世代無線通信規格である IEEE802.11ax (WiFi-6) の登場に加え，MPEG4-ALS などの音声ロスレス配信技術発達していることから，4K 映像や 360 度動画に代表される高品質なリッチコンテンツがより普及していくと考えられる．

一方で上記の映像コンテンツは，一般に静的な視聴体験にとどまっている．特に音楽イベントにおいてライブやコンサートの需要は増大しているが，それらのパッケージメディアやライブビューイングは収録機器の性能設置位置などの制約により，ユーザが自由な視点でコンテンツを視聴する，といった再生方式は困難である．さらに視点だけでなく，映像や音像の位置関係等はあらかじめ設計されたものを収録・再生するだけにとどまり，視聴者からの動作を受け付けるメディアは数少ない．インターネットを介した映像音声コンテンツ，なかでも現実空間に存在する視聴対象を三次元空間再生方式により解釈・表現したコンテンツが注目されている．

1.1.2 オブジェクトベース音響システム

音響の録音・再生方式は主に 3 つに大別される [3]．なかでもオブジェクトベースが近年注目を集めている．収録対象をビットマップ映像情報と 2 チャンネル音声情報として平面的に捉えるチャネルベースや，HOA (High-order Ambisonics，高次アンビソニックス) と呼ばれる球面調和関数の数学的解析による音場再現を用いたシーンベースとは異なり，オブジェクトベース方式は，実際に存在する空間を三次元モデルとして解釈することを意味する．すなわち，送信側は収録対象を複数のオブジェクトとして分割し，受信側はそれらのオブジェクト情報をもとに空間を再構築することで，様々な三次元再生環境に適したアプローチをとることができる．例えば，Dolby Atmos[*1] や DTS:X[*2] に代表される立体音響などの音声情報だけでなく，近年は映像情報においても RICOH THETA[*3] など 360 度カメラを用いて複数視点から撮影することで，三次元表現された自由視点映像を得られる．このようにオブジェクトベースに基づいて解釈することで，映像・音声情報

[*1] `https://www.dolby.com/us/en/brands/dolby-atmos.html` (Accessed on 09/14/2019)
[*2] `https://dts.com/dtsx` (Accessed on 09/14/2019)
[*3] `https://theta360.com/ja/` (Accessed on 09/11/2019)

のソフトウェアによる制御，すなわちサラウンドスピーカ・ヘッドマウントディスプレイ・3D テレビ・立体音響装置など任意の機器での表現が可能となり，従来のコンテンツと組み合わせた新たな視聴体験も期待されている．

1.2 目的

前節で述べた通り，インターネットの普及や通信の大容量化により映像音声視聴メディアの IP ネットワーク化・オブジェクト化が進んだ．それに伴う三次元空間の表現は，従来システムのようにユーザとコンテンツが 1 対 1 に制限されたものではなく，視聴コンテンツの受信者および送信者となったユーザ同士が双方向にオブジェクトの情報共有を行えることが求められる．本研究では，AR，特に WebAR を用いた音楽イベントの再生メディア方式を検討する．AR においても映像音声コンテンツをオブジェクト単位で解釈することで，AR コンテンツ群をソフトウェアにより状態管理し，視聴者から入力された動作を受け付けたインタラクティブな再生，かつ視聴者間で双方向にコミュニケーション可能な映像音響システムの構築が期待される．ユーザとシステム，ユーザとユーザがインタラクティブに体験可能な AR オブジェクト視聴システムの構築を目的とする．

1.3 本論文の構成

第 2 章では，XR を中心とした映像音声メディアについて概観する．第 3 章では，前章に続き本研究の提案システムとの関連が深い Software Defined Media について述べ，加えて既存の空間共有型 AR 技術についてまとめる．第 4 章では WebAR を用いたインタラクティブな映像音声視聴システムである「co-Sound」の提案を行う．第 5 章では，第 4 章で提案したシステムについての評価を行い，第 6 章で本研究のまとめと今後の課題について述べる．

第 2 章

ディジタルメディアの現状と課題

2.1 三次元映像表現の多様化

三次元映像表現の手法としてディジタル環境と実世界の環境を組み合わせた技術は，総称して XR (cross Reality) と呼ばれる．XR 市場は年率 78% で成長し，2023 年には 17 兆円規模に達すると予測されており [4]，アカデミックの分野でも日本バーチャルリアリティ学会[*1]をはじめ研究が進められている．

消費者の関心や業界の誇大宣伝の大半は，まず仮想現実 (Virtual Reality，以下 VR) に関心を向けていたが，専用ハードウェアが必要という高いハードルがあること，現実との親和性が低いことなどから，AR も関心が集まっている．AR は現実の風景にディジタルコンテンツを重ね合わせる技術であり，ユーザの視界を支配する VR とは異なり物理世界とディジタル表現されたオブジェクトを合成提示することが可能である．さらに，コンピュータビジョン・画像処理分野の急速な発展により機械学習を利用したリアルタイムな画像認識が可能となった．これにより AR 視聴体験のリアルタイム性やパフォーマンスが改善されている．従来のマーカー型 AR で用いられていた単純な平面画像認識だけでなく，立体認識や Visual SLAM の技術により，マーカーレス型と呼ばれる，画像情報のみによる AR オブジェクトをリアルタイムで高精度な特徴点＋特徴点周辺の 3 次元構築という形で表現可能となっている．エンターテインメント性の高いコンシューマ向けコンテンツだけでなく，医療や教育分野での活用 [5] も提案されている．

2.2 Software Defined Media (SDM)

近年の映像音声コンテンツに対し，特に現実に存在する対象を三次元的に解釈・表現可能とした，インタラクティブな視聴プラットフォームを構築する取り組みとして SDM コンソーシアム [1] が活動している．SDM は，オブジェクトベースのディジタルメディアを対象に，インターネットを前提とした視聴空間をソフトウェア制御することで新しい技術や利用可能性を創出する研究である．

2.2.1 SDM アーキテクチャ

SDM は図 2.1 に示される SDM アーキテクチャを構成し，以下に挙げられる要求事項を満たす．

1. 三次元の映像音声演出のソフトウェア制御
2. ソフトウェアレンダリングによる拡張演出
3. 複数収録対象のミキシング可能性
4. ユーザ・インタラクション性

[*1] `https://vrsj.org/` (Accessed on 09/15/2019)

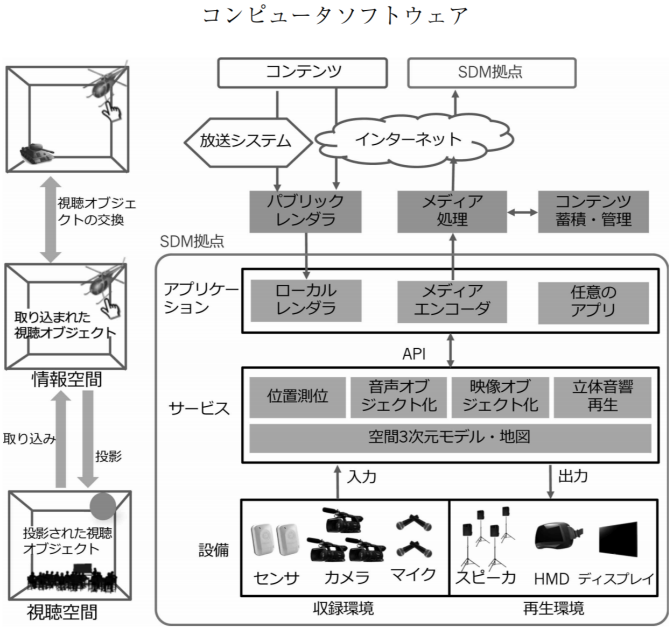


図 2.1　SDM アーキテクチャ (Fig.1 [1])

SDM はこれらの要求事項を満たすことで，入出力設備機能を抽象化，3 次元空間表現を持つ情報空間上で収録対象を管理し，これを利用したアプリケーションと収録環境の入出力デバイスとを引き離して (decoupling) 扱う．このように送信側が複数コンテンツを抽象化することは，受信側でも柔軟な視聴体験が期待されている．現在のコンテンツ配信においては送信側でカプセル化された単一のコンテンツを単一のソースから受信する形態が一般的であるが，SDM での収録データのソフトウェア管理により，リアルタイム＋アーカイブなど複数ソースのミキシングやリアルタイムでより臨場感のある空間表現といった，各再生環境での拡張演出が提供可能となる．

上記のように送受信者ともにコンテンツに対する利用可能性が拡張することで，視聴者の関心・需要を再生側のソフトウェアシステムにフィードバックした演出制御，すなわち時間と空間の制約を超えた視聴者間の相互作用が実現される．

2.3　Web ブラウザ API

ブラウザは Web サーバに接続し主に Web ページの閲覧等に用いられるソフトウェアであるが，近年ではブラウザ組み込みの機能が開発され，JavaScript API としてアクセス可能となっている．以下では WebAR，WebAudio，WebRTC について概観する．いずれの機能も W3C (World Wide Web) で標準化が進められている．

2.3.1　WebAR

AR は過去 10 年間において学術・産業界共に研究開発が進められてきたが，Mobile AR は，ハードウェア，オペレーティングシステムともに多様化が進み，その多くは特定のプラットフォームに依存しクロスプラットフォーム性を欠いたものであった．このような背景のもと，軽量かつ広範な環境で動作する Web ブラウザを用いた AR である Web AR が登場した [6]．Web AR 開発へのアプローチは大きく 2 種類に分類される．1 つは Pure な JavaScript フロントエンドのライブラリ・プラグイン，もう 1 つはブラウザカーネルの拡張機

能である．前者の代表例として AR.js [7] が挙げられる．AR.js は，ARToolkit [8] と同様の軽量なアルゴリズムを用いて AR マーカの検出を行う JavaScript ライブラリであり，マーカ型 AR ライブラリとして主流である．一方後者としては，Chrome や Firefox 等で近年開発が進められている WebXR が挙げられる [9, 10]. WebXR はブラウザを動作環境の基盤としており，Pure な JS ライブラリと比較して処理性能が高く，Mobile AR に近い操作性を提供可能としている．

2.3.2 WebAudio

WebAudio [11] とは，Web アプリケーションにおけるオーディオの処理および合成に関する高レベルの JavaScript API である．これらは近年のゲームオーディオエンジンで要求されているようなミキシング・プロセシング・フィルタリング処理を，JavaScript から制御される最適化された C++ エンジンを使って実装できるようになっており，インタラクティブ・アプリケーション等に広く活用されている．WebAudio は基本的にモジュールルーティング構造を基盤としている．ベースとなる AudioContext オブジェクト上で，音声入力源となるソースノードから出力先となるディスティネーションノードの間に開発者が自由に AudioNode オブジェクトを鎖状に配置していくことで，フィルタ，ゲイン制御や空間音響といった様々な効果を音響に連結させることが可能である．

2.3.3 WebRTC

WebRTC [12] とは，ウェブブラウザやモバイルアプリケーションにおいてシンプルな API 経由で P2P リアルタイム通信を可能とする技術である．通信には以下の表 2.1 の通り，様々なプロトコルが用いられている．

表 2.1 WebRTC で用いられるプロトコル

名称	プロトコル
media channel	UDP, RTP, RTCP
data channel	SCTP
signaling	STUN, TURN, ICE
data security	DTLS

W3C および IETF はピア間のシグナリングについて規定していないため様々な実装が存在するが，現在主に選択されている手法として，シグナリング時のみ外部サーバを利用し，Session Description Protocol (以下，SDP) および Interactive Connectivity Establishment (以下，ICE) candidate を交換することによって NAT 下に存在するデバイス同士の通信を実現する．ICE プロトコルでは，はじめに Session Traversal Utilities for NAT (以下，STUN) により， NAT 下にいるデバイスから外部の STUN サーバにグローバルアドレスとプライベートアドレス (およびポート番号) の紐づけを問い合わせる．この情報から ピア 間を接続する経路を探索し，その候補 (ICE candidate) 一覧を取得する．これら ICE candidate のリストに対し，送信するメッセージには SDP プロトコルを用いてメディアタイプやフォーマット，コーデック，その他の WebRTC のセッションに必要な情報を伝送する．シグナリングサーバ経由で通信相手と相互に候補リストを含む情報の交換が終了すると，UDP ホールパンチングによる接続可能性の確認を行い P2P 通信を確立する．

なお，Firewall や多段 NAT 等で STUN 情報から経路が探索出来ないネットワークが存在する．この場合は Traversal Using Relays around NAT (以下，TURN) に基づきピア間のパケットを仲介し，外部サーバでリレーを行うことで疑似的に双方向通信を行う．一般に WebRTC を用いたアプリケーションはリアルタイム性が求められているが，Marechal らの研究 [13] により 4G ネットワークを用いた media channel 通信において

は P2P が確立した場合と TURN サーバでリレーを行った場合とでは大きな遅延の差は生じないと報告されている．しかし TURN サーバが NSP (Network Service Provider) 終端から別大陸へと物理的位置が遠くなるにつれ，IP パス内のルータやネットワーク機器で生じるバッファの待機時間や，自立システム間のネットワークリンクによる処理等でジッタや遅延，輻輳の影響が大きくなり安定性は落ちる．

WebRTC は映像・音声ストリーム通信用の media channel とバイナリファイル通信用の data channel が実装されている．data channel は，上述の通りメッセージ指向で多重ストリームの機能を持つ SCTP (Stream Control Transmission Protocol) を採用している．SCTP はメッセージの到着保証および順序性を任意に行うことができる．一般に，信頼性のあるモードにおいてパケットロスが多く発生する場合，これらの処理にオーバーヘッドが生じるため潜在的に動作が遅くなる．しかしながら WebRTC は NAT 越えを行うにあたり SCTP over DTLS over UDP の形式を採っており，実質的に SCTP では輻輳制御とフロー制御を，NAT 越えのトランスポート機能は UDP に担っている．すなわち再送処理・順序保障を行う TCP をベースとする，リアルタイムなデータ送信制御プロトコルの RTSP (Real Time Streaming Protocol) を用いる場合よりもパケット伝送速度が速い [14].

第 3 章

関連研究

3.1 SDM 試作システム

前章で述べた SDM コンソーシアムでは，独自アーキテクチャを基盤とした試作システムを作成している．LiVRation [15] は，収録された音楽イベントをヘッドマウントディスプレイを用いて自由視点からインタラクティブ再生することを目的としたシステムである．複数地点から 360 度カメラで映像を，指向性/無指向性マイクで音声を収録した実際の音楽イベントを，オブジェクトベースオーディオの手法をもとに三次元映像音声として仮想空間内にレンダリングする．また Web360[2] [16] は，Unity[*1]で実装をしていた LiVRation と異なり，WebVR を用いてタブレット端末等のブラウザ上で 3D コンテンツを視聴することを前提としている．いずれも視聴者の頭部動作やタッチ動作に追従してコンテンツをインタラクティブに提示するものであり，7 段階のリッカート尺度による被験者実験においても，最高評価の 7 および次点の 6 を合わせた回答数は全回答数の半数以上を占めている．

3.2 視聴メディアとしての AR

近年博物館や美術館において，AR を展示品の視聴メディアとして利用するユースケースが増加している．Fenu らの研究 [17] では，Svevo 博物館に訪れた 34 人の被験者を対象に，AR を用いて個人のスマートフォンで自律的に博物館内を鑑賞してもらった上で，被験者の行動記録と体験後の 5 段階評価付けを分析した．評価付けでは，全体的な満足感，経験の新規性，美的感覚，内容への関心等のパラメータで非常に高いスコアを記録し，AR による展示物の鑑賞は有用であることが示されている．また Tillion ら [18] は，美術鑑賞の活動を(i) *Analytical Activity*; (ii) *Sensitive Activity*; に分類し，AR ガイドを用いた鑑賞が各々の活動にどのような影響を与えるか調査した．AR ガイドは，美術品への没入感を阻害し Sensitive Activity に負の影響をもたらすこともあるが，適切な情報の提示 (絵画の素材や他の芸術作品の紹介など) は訪問者にとって Analytical Activity を促進させる可能性を持つ，といった結論を得ている．

以上のように AR を媒体とした美術品・芸術品鑑賞の研究は，一定の有用性を提示している一方，オーディオビジュアルコンテンツ及びその AR 視聴に注目した研究はほとんど行われていない．

3.3 Collaborative AR

AR は単一ユーザのアプリケーションだけでなく，対面式及び遠隔コラボレーションのインターフェイスとしても高い注目を受け，研究が進められている．Collaborative AR としての最初のインターフェイスの一つ

[*1] `https://unity.com/` (Accessed on 09/15/2019)

として，Schmalsteig らの StudierStube プロジェクト [19] が挙げられる．これはシースルーヘッドマウントディスプレイを装着した 2 人のユーザが，現実世界に重ね合わされた 3D 仮想モデルを視聴するものである．Billinghurst らの MagicBook[20] は，実際の本を，複数視点に対応したタンジブル AR インターフェースとして活用した研究である．携帯型ディスプレイで本を見ると実際のページに 3 次元の仮想シーンが重ねて表示されるという AR 没入体験が可能となっており，他のユーザもその本をディスプレイ越しに覗くことで，同様の視聴体験が可能となっている．Tobias らの研究 [21] では，マルチユーザの AR のプラットフォームとして Massively Multiplayer Online Worlds (MMO Worlds) システムになぞらえて設計した．AR Stage と定義した，AR コンテンツの関連情報と操作を全て包括した仮想領域を構築し，AR システムの拡張性を向上させている．

さらに 2019 年には，Cloud Anchors[*2]や Azure Spatioal Anchors[*3]など，空間認識情報の共有・保存により複数デバイス間で同一の AR コンテンツを視聴可能としたアプリケーションが登場している．これらの空間共有型 AR は，図 3.1 のようなクライアントサーバ型により複数デバイス間でのデータ同期を行っている．ホストデバイスが空間認識により作成した 3D マップ情報を，各社が提供するクラウドデータベース上に保存し，他の参加デバイスはその三次元空間情報を取得することでセッションを開始する．以降同様にその空間内で行われた AR オブジェクトに対する情報をクラウドデータベースに送信し，受信側はその変更を受け取る方式である．

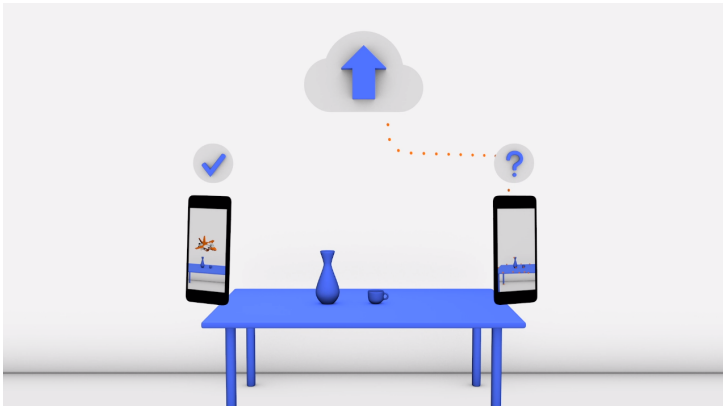


図 3.1　CloudAnchors における AR 空間の共有方式[*5]

以上の仮想空間共有型とした AR アプリケーションは，対面/遠隔地における単純な共同作業を行うためのもの・あらかじめ設計された AR コンテンツのみ他ユーザと共有され，自分または他者からのインタラクティブな操作は受け付けないもの・各座標に配置したオブジェクト情報を保存・共有するにとどまり，複数ユーザによるリアルタイムな操作性想定されていないものなどといった問題点及び課題が存在する．

[*2] https://developers.google.com/ar/develop/java/cloud-anchors/overview-android (Accessed on 09/15/2019)
[*3] https://azure.microsoft.com/ja-jp/services/spatial-anchors/ (Accessed on 09/15/2019)
[*5] 出典: https://developers.google.com/ar/images/cloud-anchors/cloud-anchor-part-2.mp4 (Accessed on 01/07/2019)

第 4 章

co-Sound: 設計と実装

本研究の要件を満たすシステムとして，co-Sound アプリケーションの設計・実装を行った．4.1 節で，これまでに述べた従来の課題およびシステム要件を再度定義し，4.2 節以下で，収録されたコンテンツデータ，システムの設計指針，実装について詳細を述べる．

4.1 要求事項の整理

本研究の目的は，1.2 節でも述べた通り収録された音楽イベント，特にそれらを 3D モデル等に変換し，仮想性の高い (実空間にとどまらない) イベントを対象として，Web 上で AR を活用したインタラクティブな再生を行い，複数人で同一コンテンツの視聴・操作を可能とするアプリケーションの提案である．この実装にあたり，以下の要件が挙げられる．

視聴者 – コンテンツ間におけるインタラクティブな視聴

収録された音楽イベントを AR 上に展開する．実空間から完全に分割された VR ではなく親和性の高い AR を用いることで，実空間に合わせたディジタル空間の投影が可能となる．また視聴者からの動作に応じた個別の映像音源データへのアクセスを許可し，従来の静的なコンテンツ再生では行えない音像の選択・再配置など視聴者自身の動的な再生および編集メディアを実現する．映像表現に加え，音声表現も立体音響での提示を目指し，視聴者の動作に追従し没入感・臨場感を与える効果が求められる．

視聴者 – 視聴者間における双方向コミュニケーション

従来方式は視聴者とコンテンツが 1 対 1 に限定されたものであったが，AR が描画されるディジタル空間を複数人で共有・同期することで新たなインタラクティブ性を創造する．メディアデータをインターネット上で管理し，ソフトウェア制御を行う視聴メディアにおいて，視聴者は受信者としてだけでなく，コンテンツの送信者としての役割も担える．メディアは単純な再生機器ではなく複数人からの操作を受け付けリアルタイムに提示可能な機能が求められる．

AR メディアデータのオブジェクトベース構造化

AR コンテンツ群をオブジェクトベースな映像音声データに構造化し，SDM アーキテクチャに基づいたソフトウェア制御処理を可能とする．上述の立体音響も，チャンネルベースや HOA/アンビソニックスではなくオブジェクトベースで設計することにより容易な個別の音源制御も期待できる．

視聴デバイスに依らない柔軟な AR 視聴体験

現在の AR 開発環境を表 4.1 において比較する．提案システムは Web アプリケーションで実装する．Web アプリケーションは専用のアプリケーションのインストールを必要とせず，Web ブラウザから直接利用可能な手軽さなため視聴者へのコンテンツ導入コストが非常に低い．ブラウザ組み込み型 WebXR はアプリケーション型やハードウェア型と同様にクロスプラットフォームの点で劣っている．OS や

端末に依存しており，視聴者だけでなく再生環境の開発者にとってもコストが大きいため，Pure な Frontend で設計した WebAR アプリケーションを採用するのが適当である．

表 4.1　AR アプリケーション 開発環境比較

	実行環境	導入コスト	動作性	クロスプラットフォーム
Pure Frontend based	ブラウザ	非常に低い	やや悪い	✓
Browser kernel based	ブラウザカーネル	非常に低い	やや良い	ブラウザ依存
Application based	スマホ，タブレット等	やや高い	やや良い	OS 依存
Hardware based	HMD など専用端末	非常に高い	良い	ハードウェア依存

4.2　設計

co-Sound アプリケーションは，収録された音楽イベントを対象とし，立体音響効果および複数端末間での空間同期を可能としたマーカ型 WebAR アプリケーションである．視聴者はブラウザから該当する Web ページに接続することでアプリケーションにアクセスすることができ，マーカ上に存在する音楽シーンを模した 3D オブジェクトを自由視点で視聴可能，かつその視聴位置での立体音響空間を体験することも可能である．また特定のルーム内に参加したピア同士で AR オブジェクト情報を双方向で通信し合うことにより，複数端末間でのリアルタイムな空間同期を可能としている．

4.2.1　収録データ

co-Sound アプリケーションで利用した音楽イベントは，2017 年 1 月 26 日に六本木ミッドタウン内 Billboard Live Tokyo で開催された Musilogue Band のコンサートを収録したものである．このコンサートでは，Drums，Electric Bass，Keyboard の三種からバンドが構成されており，各楽器単体でマイクを設置し，個別に音源を収録している．[22]

なお SDM では，コンソーシアムが独自に保有する音楽イベントのデータを利用する機能性や応用の自由度を高め Linked Open Data として外部に公開するため，以下のようなデータ構造を持つ SDM Ontology[23] を定義している．

- SDMEvent:イベント全体の情報
- Context: イベントにおいて行われたプログラムの情報
- Target: 収録対象の情報
- Recorder: 収録機器の情報
- MediaObject: 生成されるメディアの情報

本研究ではこのうち Target 情報を対象とし，上述のコンサートにおける各楽器の属性，位置情報および音声情報を利用した．

4.2.2　co-Sound のシステム概要

co-Sound のシステム設計の概要を図 4.1 に表す．視聴者はタブレット端末等から AR マーカをカメラに写すことにより，co-Sound へ映像情報を入力する．入力された映像情報から，マーカの検出およびカメラ座標の推定が行われ，前節で述べた収録データが持つ位置情報と照会することで AR オブジェクトの座標および原

点を決定する．また AR マーカの検出，AR 映像の位置やカメラ座標の推定に加え，視聴者からのタッチ動作を入力情報として，これに応じた AR 映像や音響をシステム内でリアルタイムにレンダリングして出力し，インタラクティブ性を実現する．

さらに，co-Sound 内で状態管理が行われた視聴オブジェクトのメタデータをシリアライズし，他端末間で相互に通信することで，リアルタイムに AR 空間を同期させる．入力となるカメラ映像は端末依存の情報となるため，各ユーザによる自由視点視聴の要求事項は満たされる．

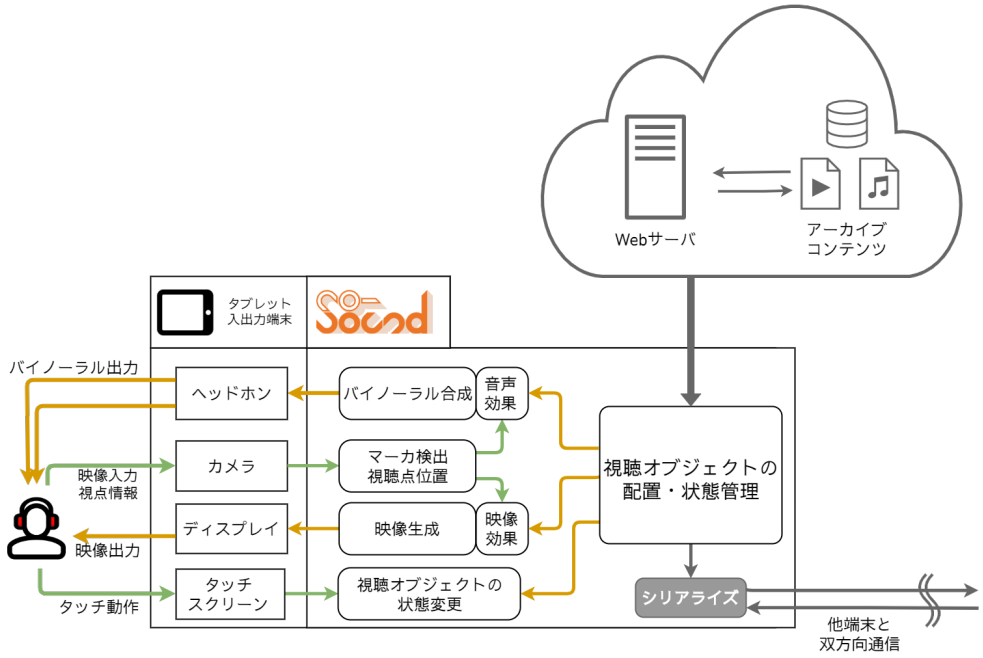


図 4.1　co-Sound 設計図

4.2.3　リアルタイムな AR 空間同期

リアルタイムで AR 空間情報を同期するにあたり，表 4.2 にていくつかの通信方式を比較する．本研究では低遅延通信方式として 2.3.3 節で述べた WebRTC を採用する．P2P によって各ノード間を結びフルメッシュネットワークを構築し，AR オブジェクトのメタデータの同期を行う．同期のための通信量は $O(n^2)$ となるが，デバイス間の通信における経由ホップ数が少なく，よりリアルタイム性が保証されることが期待される．

表 4.2　メッセージ同期トポロジの違い

	通信トポロジ	代表的プロトコル	通信量	送信先
P2P	1 対 1/フルメッシュ	WebRTC	$O(n^2)$	各ピア
client/server	1 対 1/スター	WebSocket, HTTP	$O(n)$	通信相手
Pub/Sub	1 対 n/マルチキャスト	MQTT	$O(\log n)$	broker

4.3　実装

前節で述べた設計アプローチをもとに，co-Sound を実装した．本システムは，主に WebAR 用フレームワークである AR.js [7]，Web ブラウザ上で音声を扱うための WebAudio API [24] を用いて実装した．アプ

リケーションのスクリーンショットは図 4.2 の通りである．



図 4.2　co-Sound スクリーンショット

4.3.1　WebAR 動作概要

実行環境は以下の表 4.3 の通りである．Web サーバからは co-Sound のアプリケーションデータが送られ，新しくルーム (詳細は 4.3.4 節に記述) を設定した場合，4.2 節で述べた音楽イベントの基本データである各楽器の 3D モデルファイル，位置情報および映像音声メディアファイルが伝送される．

Web ブラウザにおいて，視聴端末 (タブレット端末やスマートフォンなど) に付属したカメラからの映像は`<video>`タグに，生成される AR 映像は`<canvas>`タグにレンダリングされるが，マーカの画像認識およびカメラ位置の推定は AR.js，AR 映像や音響可視化オブジェクトの描画は Three.js により行われる．Three.js オブジェクトのシーンは図 4.3 のように構成し，対象となる楽器ごとに描画グループを分割した．このうち camera オブジェクトを用いて AR.js の`ArMarkerControls`クラスを定義することで，視聴端末の物理カメラと Three.js シーン内でのカメラ位置が同期され，任意の位置で AR 映像を視聴することが可能となる．

表 4.3　co-Sound 実行環境

WebAR ライブラリ	AR.js v1.5.0, aframe.js v0.9.2
WebGL ライブラリ	Three.js v0.110.0
実行ブラウザ	Chrome v79, Safari v604.1

4.3.2　ユーザインターフェース

co-Sound は AR コンテンツの視聴に加え，タブレット端末等を想定し視聴者からのタッチ動作により AR オブジェクトに対し簡易的なインタラクティブ操作を可能とする実装を行った．視聴者は picking と呼ばれる処理により，視聴端末上の AR 映像をタップすることで co-Sound 内部オブジェクトのデータにアクセス可能となる．co-Sound の picking 処理の概要を図 4.4 に示す．視聴者のタップした座標を，`MouseEvent.clientX`，`MouseEvent.clientY`プロパティにより取得し，スクリーン上で-1 – 1 へ正規化する．次にこのスクリーン座標系をカメラ座標系に変換する．`THREE.Vector3`クラスの`unproject(camera: Camera)`メソッドがこれにあたるが，Three.js v0.110.0 において正常な計算が行われないため，原義に基づき以下のプログラム 4.1 でこれを求めた．更に`THREE.Raycaster`クラスに，光線が発

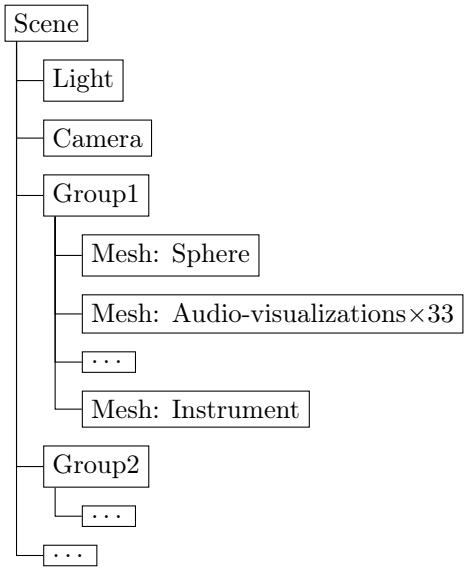


図 4.3　Three.js オブジェクトのシーン構造

する元となるカメラの位置ベクトルおよび上式で求めた光線の方向ベクトルを指定する．これによりレイキャスタを作成し，タップした点の位置ベクトルから延長した光線ベクトルを導出する．この光線ベクトルに対し`intersectObjects`メソッドを利用することで，シーンに存在するオブジェクトとの交差判定を行い，光線ベクトル上のオブジェクトが返される．以上の手順でタップ動作による AR オブジェクトの選択を可能とした．

Listing 4.1　スクリーン座標系からカメラ座標系への変換

```
1 const THREE = require('three')
2
3 const mat = new THREE.Matrix4();
4 const mouseVec = new THREE.Vector3(screenX, screenY, 1)
5   .applyMatrix4(mat.getInverse(camera.projectionMatrix))
6   .applyMatrix4(camera.matrixWorld);
```

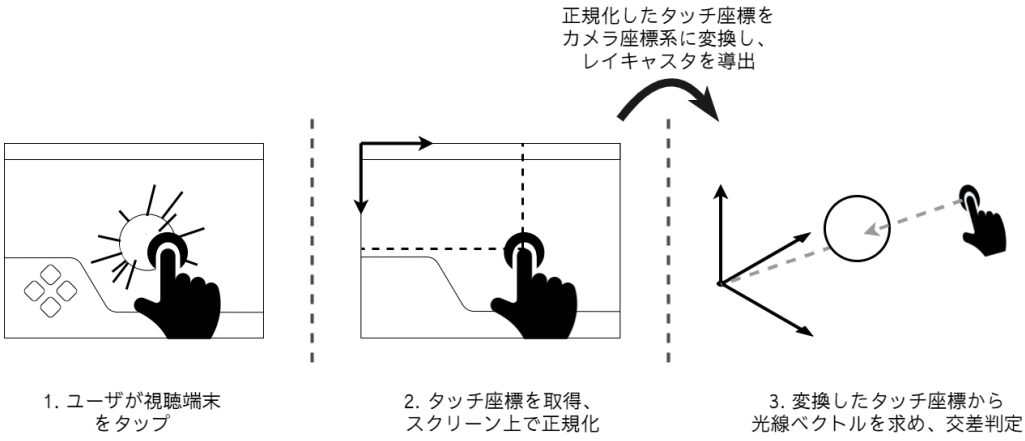


図 4.4　co-Sound におけるピッキング処理

選択したオブジェクトに対して，(1) 音量の ON/OFF; (2) 座標の変更; のインタラクションを実装した．目

的の AR オブジェクトへのタッチ動作により音量の ON/OFF が切り替わり，加えて各楽器オブジェクトの中心に位置する Sphere オブジェクトが不透明状態となることで，選択状態を表すフラグが与えられたことを示す．フラグが与えられたオブジェクトを操作する十字コントローラ型 UI (図 4.2 アプリケーション画面左) を実装し，対象となった AR 映像の xyz 軸三方向への移動を開発した．

4.3.3 co-Sound の音響制御

Web ブラウザによる音響効果を WebAudio を利用して実装した．AudioContext オブジェクトをベースとして，Web サーバから HTTP リクエストにより取得した Buffer Source から出力である distination まで，ノードを鎖状に連結し音響をリアルタイムレンダリングする．

音響の ON/OFF 動作は，ゲイン調整ノードである Gain オブジェクトの gain 値を 0 または定数にすることで表現した．Box オブジェクトを用いた音響の可視化は，Web360^2 [16] 同様，WebAudio の `AnalyzerNode.getByteFrequencyData()`メソッドにより時間領域データから周波数領域データを取得し，有効周波数帯を Box オブジェクトの長さおよび色に変換することで表現した．

最後に，立体音響のリアルタイムレンダリングを実装した．AR 映像は AR.js を用いたマーカー検出およびオブジェクトの位置情報により適当な座標に描画されるが，音像はこれに追従しないため随時座標を計算する必要がある．座標計算に必要な座標系を表 4.4 にまとめる．以下では，ある点 P を座標系 A で見たとき，${}^{\mathrm{A}}\boldsymbol{P}_{\mathrm{subscript}}$ と表記する．AR.js では検出したマーカーを原点としたローカル座標系上に AR 映像が配置されるため，これを co-Sound のワールド座標系として扱う．すなわちマーカ中心は ${}^{\mathrm{W}}\boldsymbol{O}$，ある楽器オブジェクト a は ${}^{\mathrm{W}}\boldsymbol{a}$ およびカメラオブジェクトは ${}^{\mathrm{W}}\boldsymbol{P}_V$ と表される．一方，WebAudio において三次元位置情報を扱う Panner ノードは視聴者を原点としたローカル座標系を渡す必要があるため，ワールド座標系をビュー座標系に変換する．ビュー座標系から見た楽器 a は ${}^{\mathrm{V}}\boldsymbol{q}$，ワールド座標系の原点は ${}^{\mathrm{V}}\boldsymbol{P}_{\mathrm{W}}$ と表され，座標系の回転を ${}^{\mathrm{V}}\boldsymbol{R}_{\mathrm{W}}$ と表記すると，以下の式 4.1 が成り立つ．

表 4.4　立体音響のレンダリングに関する座標系

座標系 (および略称)	説明
ローカル座標系 (以下 L)	オブジェクトを原点に取った座標系.
ワールド座標系 (以下 W)	全オブジェクト共通の座標系.　AR.js では検出したマーカを基準点とするため，これを原点とした.
ビュー座標系 (以下 V)	カメラを原点に取った座標系. カメラとオブジェクトの相対位置を表す.

$$
\begin{aligned}
{}^{\mathrm{V}}\boldsymbol{q} &= {}^{\mathrm{V}}\boldsymbol{P}_{\mathrm{W}} + {}^{\mathrm{V}}\boldsymbol{R}_{\mathrm{W}} \cdot {}^{\mathrm{W}}\boldsymbol{a} \\
&= {}^{\mathrm{V}}\boldsymbol{R}_{\mathrm{W}} \cdot ({}^{\mathrm{W}}\boldsymbol{a} - {}^{\mathrm{W}}\boldsymbol{P}_{\mathrm{V}})
\end{aligned} \tag{4.1}
$$

上式よりワールド座標系の任意のオブジェクトをビュー座標系で表せるため，カメラ中心即ち視聴者を基準とした立体音像をレンダリングすることが可能となる．

4.3.4 端末間シグナリングおよびコネクション

複数端末間での AR 空間同期のために，リアルタイム対話型マルチメディアサービスとして設計されたオープンソースの PaaS (Platform as a service) である SkyWay v2.0.1 [*1] を利用した．SkyWay は WebRTC のコネクション接続用のシグナリングサーバ，パケットリレー用の TURN サーバ，そして WebSocket サーバを提供しており，各ピア上で立てた Peer インスタンスから呼び出すことで接続から離脱までのコネクション管理が可能となる．なお，これらのサーバは東京に配備していると公開されている [*2]．本研究では SkyWay を利用して名前空間で分割されたルームを作成し，co-Sound を立ち上げたピアがルーム上で相互にコネクションを確立するよう設計した．最も有用なケーススタディであるブロードキャスト (すなわち 1 対 n の通信) を用いて，オブジェクトベースに構成された AR 映像音声のメタデータを送信することでリアルアイムな同期を実装した．

ここで，通信プロトコルとして(1) WebRTC によるメッシュ型通信; (2) WebSocket によるスター型通信; の 2 種類を用いて実装を行った．公開されている SkyWay JavaScript SDK (Software Development Kit) のソースコードでは，ルーム型バイナリデータ通信用には WebSocket のみ実装されている．そこで 1 対 1 型に実装されている WebRTC DataChannel のソースコードをもとに，ルーム型においても各ピア間で相互に DataChannel コネクションを構築するよう改良を行った．

[*1] `https://github.com/skyway/skyway-js-sdk`

[*2] `https://support.skyway.io/hc/ja/articles/115003112067-TURN` サーバの配置場所について(Accessed on 01/04/2020)

第 5 章

実験と評価

5.1 定量的評価

AR 空間情報の同期遅延を測定することにより，提案手法の試作アプリケーションである co-Sound の定量的評価を行った．

5.1.1 実験方法

4.3.4 節で述べた通り，co-Sound では AR 空間同期のため名前空間で分割されたネットワーク上のピア間でオブジェクト情報を双方向に通信する．ネットワーク実環境下で(1) 通信プロトコル; (2) メッセージサイズ; (3) 接続数; (4) 端末性能; をそれぞれパラメータとした実験を行い，co-Sound の AR 空間同期性能を評価した．実験に用いた端末は表 5.1 の通りである．なお，ブラウザのバージョンは表 4.3，利用する WebSocket サーバ，Signaling サーバおよび TURN サーバは NTT Enterprise Cloud[*1]が提供するものである．本測定において利用した各サーバのパケット処理性能の違いにより遅延が増減する可能性については考慮しないものとする．WebRTC による通信は信頼性モードを ON にしている．

表 5.1 co-Sound 実験環境

	OS	**CPU**	メモリ
PC	Windows 10 version 1809	Intel® Core™ i7-8550U	16 GB
タブレット	iOS 12.3.1	Apple A10X Fusion	4 GB
スマートフォン	Android 9, EMUI version 9.1.0	HiSilicon Kirin 960	4 GB

以降では n 番目のピアを P_n と表す．

co-Sound は，あるピア P_1 が AR 空間に存在するオブジェクト情報を JSON 形式で同一ルームに属する他のピア全てに送信することで同期を行っている．従って以下の実験では，RTT (Round Trip Time) を計測することで AR 空間同期の遅延を評価する．P_1 で生成されたパケットを，ブラウザで送信命令を出してから P_1 のユーザ空間，カーネル空間，NIC (Network Interface Card) を経てインターネットに送出されるまでの内部処理時間を t_1^A，LAN (Local Area Network) または WAN (Wide Area Network) を経由して P_1 から P_2 までへの転送にかかる時間を t_{12}^B と表すとする．P_2 は P_1 から受信したパケットをブラウザまで渡し，そのまま同じパケットを P_1 へ送り返す．ブラウザ上でのパケットの受送信にかかる処理時間を t_2^P とする．以上より一連のプロセスによる時間 RTT を表すと，式 5.1 の通りである．

[*1] https://ecl.ntt.com/documents/service-descriptions/webrtc/webrtc.html

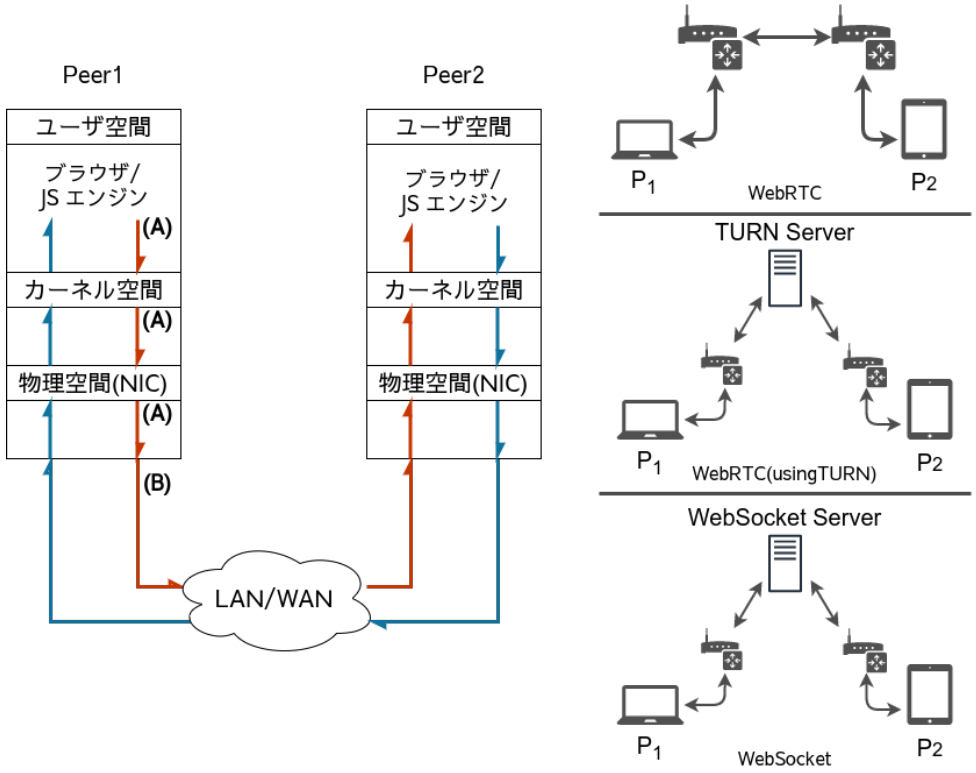


図 5.1　端末間通信トポロジ

なお，各ピアにおける内部処理時間やピア間における転送時間は，上り下りで等しいものとする．また，AR 空間の同期には上記のデータ転送遅延に加えレンダリングする時間も含まれる．しかし各ピアは，JavaScript がシングルスレッドであることから，他ピアへのオブジェクト操作のデータ送信命令を発した後に自身の画面上に反映させている．転送時間に比べ，各端末によるレンダリングの差は十分に小さいと考えられることから，レンダリング含めた AR 空間の同期についても各データ転送遅延のみを評価対象とした．

$$\mathrm{RTT} = 2 \times (t_1^A + t_{12}^B + t_2^A) - t_2^P \tag{5.1}$$

実験 1–4 の結果を図 5.2–5.5 に示す．100 回計測した RTT を箱ひげ図にプロットし，第 1 四分位数 − (中央値 − 第 1 四分位数) × 1.5 以下，または 第 3 四分位数 + (第 3 四分位数 − 中央値) × 1.5 以上のものを外れ値とした (以下図 5.3–5.5 についても同様)．図中および以降で「WebRTC (host)」は STUN のみで疎通した P2P 通信による WebRTC，「WebRTC (relay)」は TURN サーバを介したパケットリレーが発生した場合の WebRTC と表現する．

5.1.2　実験 1: 通信プロトコルによる遅延

実験 1 では，AR オブジェクト情報のピア間通信に用いるプロトコルをパラメータとして，co-Sound がより低遅延で同期可能であることを示す．比較する通信プロトコルは Web ブラウザで利用可能なリアルタイム通信プロトコルとして(1) 低遅延通信の提案手法として co-Sound で採用した WebRTC による LAN 内 P2P 通信; (2) TURN サーバによるパケットリレーが行われた際の WebRTC 通信; (3) HTTP の双方向通信プロトコルとして一般的な WebSocket 通信; の 3 種類を選択した．また P_1 として表 5.1 の PC を，P_2 としてタブレットを選択し，1KiB の JSON データを 5 秒間隔で 100 回転送させた際の RTT を計測した．

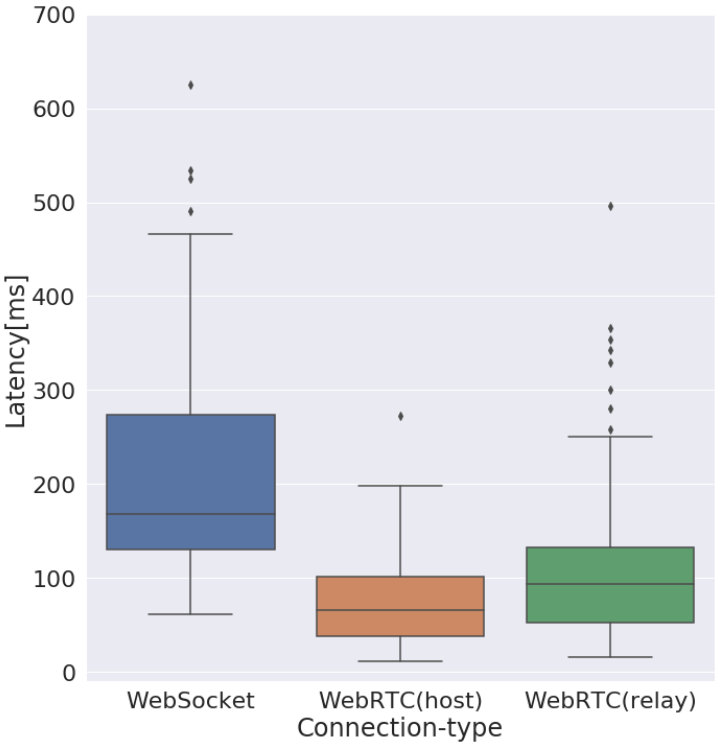


図 5.2　実験 1: 通信プロトコル別 RTT 測定の結果

表 5.2　実験 1: 通信プロトコル別の平均 RTT と中央値 (ms)

	通信プロトコル		
	WebSocket	**WebRTC (host)**	**WebRTC (relay)**
平均 **RTT** (ms)	210 ± 116	73 ± 47	106 ± 87
中央値 (ms)	168	65	92

図 5.2 より，端末間通信プロトコルとして WebSocket を選択したときの平均 RTT は 210 ms，WebRTC (host) を選択したときの RTT は 73 ms と，65.0% 短縮された．WebRTC (relay) のときも平均 RTT は 107 ms となった．これより，提案手法となる WebRTC による端末間通信は，パケットリレーを行った際でも WebSocket 通信と比較して低遅延でパケット転送が可能である．また標準偏差は同順に 116 ms，47 ms，87 ms と導出されたことから，遅延時間のバラつきも抑えられた．

5.1.3　実験 2: メッセージサイズによる遅延

実験 2 では，メッセージサイズをパラメータとした際の転送遅延について評価を行う．メッセージサイズは昇順に 20bytes，120bytes，220bytes，420bytes，820bytes，1KiB，2KiB および 4KiB である．各々のメッセージを転送し合った際の RTT を計測した．実験 1 同様，P_1 として表 5.1 の PC を，P_2 としてタブレットを選択し，5 秒間隔で 100 回転送させた．

メッセージサイズを 20 bytes から 4096 bytes まで変更させた場合は，図 5.3 および表 5.3 より WebRTC・WebSocket ともに平均 RTT・標準誤差に大きな変化は見られなかった．20–4096 bytes の間では，WebRTC の平均 RTT は約 80 ms，WebSocket の平均 RTT は約 200 ms と，メッセージサイズに依らずおよそ一定であった．

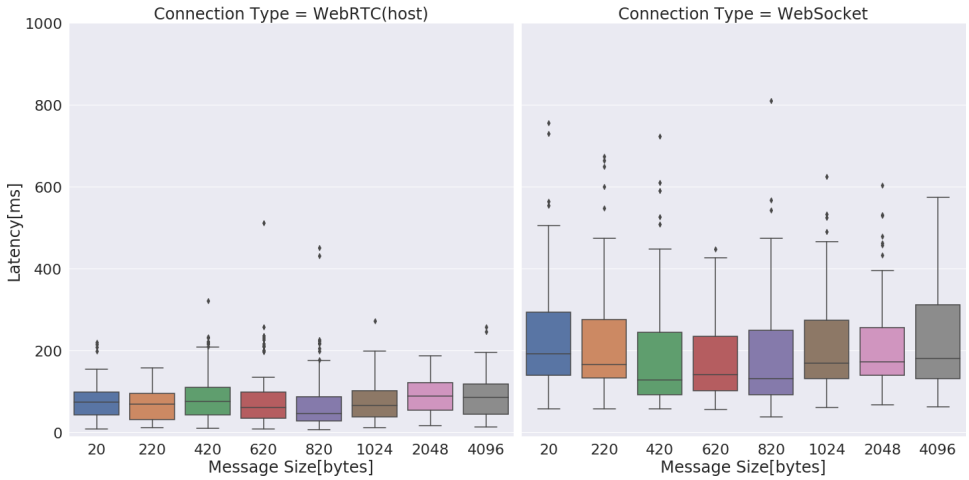


図 5.3　実験 2: メッセージサイズ別 RTT 測定の結果

表 5.3　メッセージサイズ別の平均 RTT (ms)

	メッセージサイズ (bytes)							
通信プロトコル	**20**	**220**	**420**	**620**	**820**	**1024**	**2048**	**4096**
WebRTC (host)	76 ± 45	66 ± 37	91 ± 66	79 ± 74	70 ± 74	73 ± 47	89 ± 42	85 ± 48
WebSocket	224 ± 120	214 ± 130	182 ± 130	166 ± 90	181 ± 120	210 ± 120	212 ± 110	229 ± 130

5.1.4　実験 3: 接続端末数による遅延

実験 3 では，接続端末数をパラメータとした際の転送遅延について評価を行う．表 5.1 の PC，タブレットに加え同表のスマートフォン 1 – 3 台を同一ルーム内に参加させ (これらを $P_3, ..., P_5$ とする)，WebRTC または WebSocket の 2 通りで接続した．5 秒間間隔で 1KiB の JSON データを 100 回転送させ，RTT を計測した．

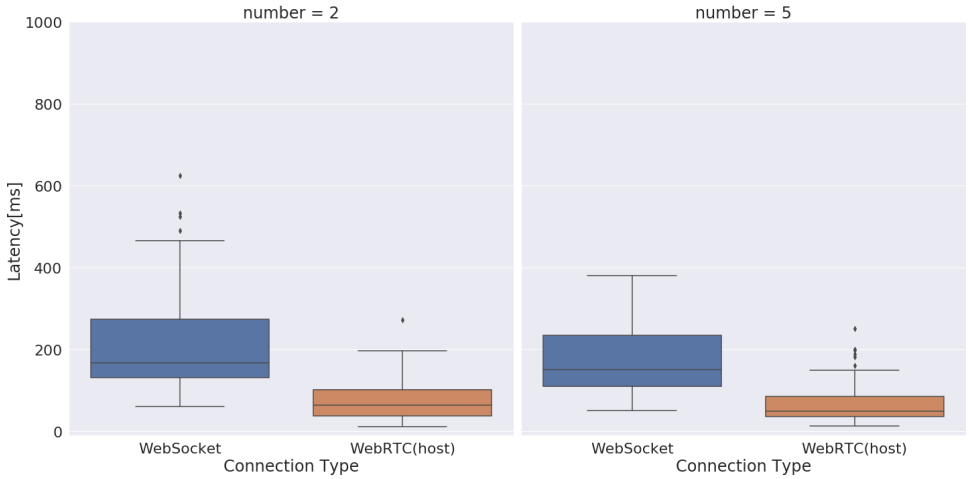


図 5.4　実験 3: 接続端末数別 RTT 測定の結果

接続端末数を P_1 と P_2 の 2 台から P_5 までの 5 台に増やしたときの平均 RTT は，図 5.4 および表 5.4 より WebSocket のとき 65 ms，WebRTC のとき 170 ms であった．表 5.2 との比較より，小規模では接続台数が増加してもピア間の遅延には影響が無いと言える．

表 5.4　接続端末数を 5 台とした場合の平均 RTT と中央値 (ms)

	通信プロトコル	
	WebSocket	**WebRTC (host)**
平均 **RTT** (ms)	65 ± 46	170 ± 78
中央値 (ms)	49	150

5.1.5　実験 4: 端末性能による遅延

実験 4 では，接続端末の性能差による転送遅延への影響について評価を行う．P_1 には表 5.1 の PC を用い，P_2 としてタブレットまたはスマートフォンを選択した際の RTT を計測し，比較した．通信プロトコルには WebSocket または WebRTC を用い，5 秒間隔で 1KiB の JSON データを 100 回転送させた．

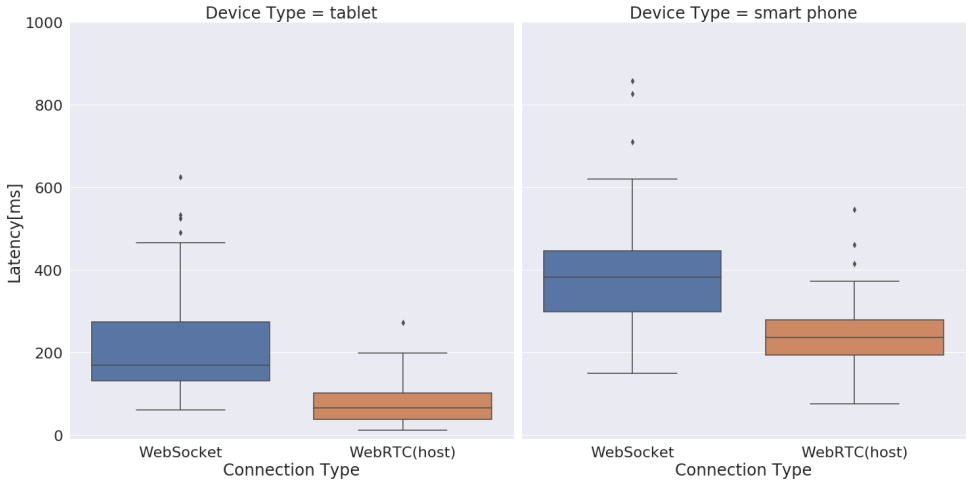


図 5.5　実験 4: 端末性能別 RTT 測定の結果

表 5.5　P_2 としてスマートフォンを選択した場合の平均 RTT と中央値 (ms)

	通信プロトコル	
	WebSocket	**WebRTC (host)**
平均 **RTT** (ms)	363 ± 100	239 ± 76
中央値 (ms)	379	236

実験 4 の測定より，表 5.1 のスマートフォンを選択した場合，平均 RTT は WebRTC では約 240 ms，WebSocket では約 360 ms となった．図 5.5，表 5.2・表 5.5 より，ピアとして用いる端末性能は，WebSocket・WebRTC ともに遅延に大きく影響を与えると言える．

5.1.6　考察

以上の測定より，提案手法で用いた WebRTC による端末間通信は，4.1 節で述べた要求事項を考慮し WebSocket による通信と比較して適当であると言える．

第一に，パケット転送のリアルタイム性が挙げられる．空間共有型 AR における QoE (Quality of Experience) についての評価は未だ定まっていないが，西堀らのインターネットを介した音楽セッションの遅延認

知に関する研究 [25] では，30 ms 以上で遅延を認知し，50 ms 以上では演奏が困難になると報告されており，また一方で VR での FPS (First Person Shooting) ゲームでは，100 ms 以上でプレイヤーのスコアおよび QoE (Quality of Experience) が低下すると示されている [26]．本実験の結果より，WebSocket を用いた通信では遅延が 100 ms 以上であるが，WebRTC を用いた通信では平均遅延が 50 ms 以下である．同一 LAN 内での P2P はワンホップで通信が可能であり，加えて 2.3.3 節でも述べた通り，WebRTC は SCTP プロトコルを用いることでオーバーヘッドを低減し，HTTP プロトコルを用いた通信よりも低遅延性を保つことが可能であることが確認できた．

第二に，実験で測定した範囲内で転送遅延はメッセージサイズ・接続端末数に非依存であったことが挙げられる．co-Sound では，ユーザ操作により変更のあった AR オブジェクトデータを他端末へブロードキャストする方式であるが，オブジェクト数の増加や属性の複雑化により送信パケットのペイロードが長くなった場合でもリアルタイム性に問題がなく拡張性が高いと言える．また一般的にサーバ–クライアント間でコネクションを確立する WebSocket と異なり，WebRTC ではサーバレスな P2P 通信方式，すなわち接続端末数分のコネクションを確立する必要があるが，10 台以下の小規模であればコネクションの増加に伴う CPU 負荷は遅延に影響しないと結論付けられる．

5.2 主観評価

定量的評価に加え，アンケート調査により co-Sound の主観評価を行った．

5.2.1 アンケート収集

12 月 6 日から 17 日にかけて co-Sound を体験してもらった被験者にアンケートを回答してもらい，主観評価の調査を行った．視聴端末として Apple iPad Pro (10.5 inch) iOS 12.3.1，ヘッドホンとして Sony WH-1000XM2 を利用し，有線接続して体験を行った．はじめにアプリケーション概要と操作方法を説明し被験者に自由に視聴体験を行ってもらった後に，アンケートの回答をお願いし評価を取得した．体験者の総数は 25 名，内男性 24 名，女性 1 名であった．年齢構成は 20 代 20 名，30 代 2 名，40 代 1 名，50 代 2 名であり，職業構成は学生 21 名，教員 2 名，社会人 2 名となった．

5.2.2 評価項目

アンケートは，以下に示す 8 項目から成る．それぞれ 1 を最低評価，7 を最高評価とした 7 段階のリッカート尺度を用い，co-Sound 体験後に適当な評価付けを行ってもらった．またアンケートの最後には自由記述欄を設け，体験に関するコメントを得た．

Q1　音声は AR 映像の方角から聴こえましたか？
Q2　音声は AR 映像の距離感と一致していましたか？
Q3　AR 映像を動かしたとき，音響も追従して動いたと感じられましたか？
Q4　AR マーカに対して自分の位置を変えたとき，音響も追従して動いたと感じられましたか？
Q5　音量可視化による音響オブジェクトの ON/OFF 操作は直感的でしたか？
Q6　コントローラによる AR 映像の移動は直感的でしたか？
Q7　AR マーカの認識精度は十分でしたか？
Q8　Web ブラウザ上でも 3D コンテンツのインタラクティブな視聴体験ができましたか？

設問 Q1 – Q4 は音響について問う設問である．加藤らの先行研究 [16] においても音響の立体感に関する評

価を行ったが，評価が分散していた．これは音響の立体感を構成する要素のうち，いずれを問うた設問であるかが被験者にとって曖昧であったことが原因であると結論付けられている．したがって本研究では，音響の立体感および視聴者と AR オブジェクトの位置関係を考慮し，4 項目を通して音響の評価を行う．

設問 Q1 および Q2 は，AR マーカと視聴者の位置を固定した状態での立体音響効果を知覚できるかを問う設問である．設問 Q1 は映像上の AR オブジェクトの位置と，出力される音響から認知される方向・位置と差異がないかを問い，設問 Q2 では同じく AR オブジェクトの位置と，出力される音響から認知される距離感と差異がないかを問うものである．設問 Q3 では，視聴者の位置を固定した状態で十字コントローラによる AR オブジェクトの移動を行った際に，出力される音響が移動後の AR オブジェクトの位置から聴こえる音像と差異がないかを問い，設問 4 では，AR オブジェクトの位置を固定した状態で視聴者が移動を行った際に，出力される音響が AR オブジェクトの位置から聴こえる音像の方向・位置と差異がないかを問う．設問 5，6 ではユーザインタフェースの評価としてそれぞれ音響の ON/OFF 操作，十字コントローラによる AR 映像の移動操作に対する評価を問う．設問 7 では AR.js によるマーカ検出の精度を問い，設問 8 では総合的に Web ブラウザを用いた co-Sound による視聴体験の評価付けを行った．

5.2.3　評価結果

主観評価の結果を図 5.6 に示す．縦軸は前項で述べた Q1 – Q8 までの質問項目およびその有効回答数を，横軸は 1 – 7 までの 7 段階評価の回答比率を積み上げ棒グラフで表し，中間評価を表す 4 の回答比率の中間を原点に置いた．評価 5，6，7 が多いほど横軸正方向に偏り，評価 1，2，3 は負方向に偏る．

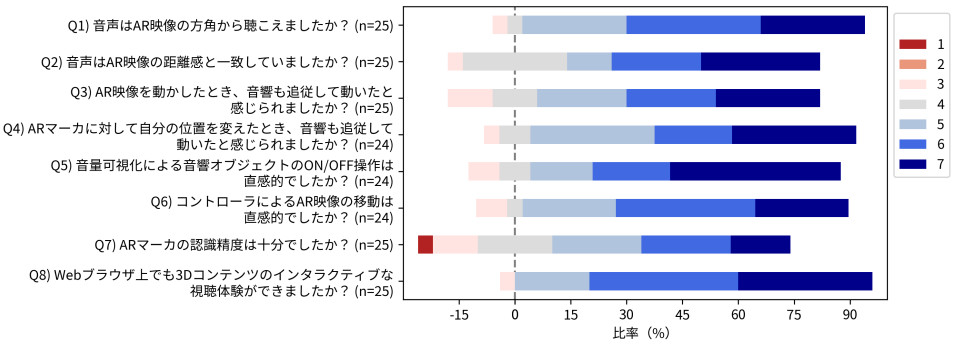


図 5.6　主観評価の集計結果

設問 Q7 を除いた全ての項目において，評価 6 および評価 7 を合わせた回答比率が 50% を上回り，総合評価である Q8 においては 76% であった．評価 3 以下の合計回答比率は，設問 Q3 が 12%，Q7 が 16% であったが，その他設問は全ては 10% 未満となった．一方，設問 Q7 は AR.js のマーカ推定精度を問うものであったが，平均評価は 4.96，最高評価の評価 7 の回答比率は 16%，更に最低評価である評価 1 の回答も存在した．全項目中で平均評価が 5 未満となったのは設問 Q7 のみであり，評価 7 の回答比率も最低であった．

5.2.4　考察

はじめに，音響に関する設問である Q1 – Q4 について述べる．Q1 および Q4 の評価 5 – 7 の合計回答比率はそれぞれ 92.0%，87.5% となった一方で，Q2 および Q3 は評価 5 – 7 の合計回答比率が 68%，76% と前者の設問に比べて大きく評価が下がった．5.2.2 節でも述べた通り，Q1 および Q4 は AR 映像に対する音像の方向・位置を問うものであった．Q2 および Q3 に関しても音響についての設問であったが，前者と異なり AR

映像に対する音像の距離感を問うたものであった．すなわち本研究で試作したアプリケーションにおいて AR 映像に対する音像の方位追従性は優れているが，音像の距離追従性が悪いと言え，加藤らによる先行研究 [16] において音響の立体感に対する評価が分散した原因に音像の正しい距離感があると結論付けられる．方位追従性に比べ距離追従性の評価が低いという結果は，WebAudio PannerNode で用いられているバイノーラルアルゴリズムが簡易なものであるため，実空間とのキャリブレーションが不十分であることを示唆している．頭部伝達関数を考慮した HRTF モードも存在するが，CPU 負荷が高く，WebAudio の仕様上，方位角と仰角に依存する遅延が加えられることから本研究におけるリアルタイムレンダリングの要求を満たせないため不適当である．

ユーザインタフェースに関する設問 Q5,6 は，どちらも LiVRation，Web360^2 と同等に高い評価を受けた．

次に AR.js に関する設問 Q7 について述べる．前節で述べた通り，他の項目に比べマーカの検出精度は低い評価であった．AR.js で利用される ARToolkit はマーカ検出に原始的なアルゴリズムを用いており，偽陰性率が高い [27] ことで知られているが，本研究においても主観評価に現れる結果となった．表 4.1 で述べた通り WebAR はブラウザベースのものが主であるが，視聴者にとって高い評価を得るには精度が不十分であることが結論付けられる．

最後に，自由記述項目について述べる．「立体的な音でおどろいた」等肯定的意見は音響に関するものであった一方で，否定的意見はマーカの認識精度，ユーザインタフェースの二つに大別された．前者は Q7 にも反映されていた通りである．後者は「オブジェクトをタップしても反応しないことがあった」「音響オブジェクトは指で動かしたい」等，AR オブジェクトの動かし方が直感的でない・精度が悪いという意見も多かった．設問 Q5 および Q6 の結果よりタッチによる音響操作はインタラクティブ性が高いと言えるため，AR オブジェクトの移動操作 UI には改善の余地がある．

第 6 章

結論

6.1 まとめ

本研究では，AR，特に WebAR を利用したインタラクティブな視聴を可能とする映像音声メディアを提案し，Web アプリケーション「co-Sound」を実装した．対象となる音楽イベントを，平面的かつ静的なビットマップ映像・チャネルベース音響ではなく，オブジェクトベースに基づいた収録空間の抽象化・構造化を行うことで再生環境において柔軟な設計が可能となる．三次元映像表現手法として AR を活用し，視聴者からのオブジェクト操作を取得しリアルタイムに動的な映像音声のレンダリングを行うマルチモーダル・インターフェイスを設計することで，実空間との親和性が高い映像音声のディジタル空間の合成提示およびインタラクティブなコンテンツ視聴が可能となった．従来 1 対 1 に制限された視聴者と映像音声コンテンツとの関係性を拡張する，視聴メディア端末間の低遅延双方向通信によって，視聴者間同士がコンテンツの送受信者となり，リアルタイムにインタラクションを実現することができた．これらの要求事項を満足する視聴メディアを Web アプリケーションとして実装し，視聴導入コストを抑えたマルチプラットフォームな再生メディアとしての役割も持つ．

低遅延双方向通信の実装にあたり利用した WebRTC は，co-Sound で想定されるメッセージサイズ・接続端末数には依存しない低遅延性を確保できた．その遅延を約 30–50 ms まで抑えられることをアプリケーションの性能評価・測定により確認し，約 100 ms の遅延があった WebSocket と比較するとリアルタイム性が要求される Web アプリケーションにおいて有効であると結論付けられた．しかし，視聴端末の性能差は，本研究で測定した遅延の評価では最も影響をもたらした．近年モバイルデバイスの性能は向上し続けているが，ハイエンドモデル以外の端末では大きな遅延が発生する可能性があると言える．また，25 名を対象として，アプリケーションの立体音響・AR・インタラクティブ性の 3 つに大別される 8 項目 7 段階のアンケートによる主観評価実験を行った．立体音響の方位追従性やタッチ操作による音響操作は QoE が高く，WebAR を活用したインタラクティブな視聴メディアとして高い評価であった．一方で，立体音響の距離追従性と WebAR のマーカ認識精度については評価が低く，実空間と齟齬のないディジタル音像のキャリブレーションや高精度なマーカ検出が必要であるという結果を得た．

6.2 今後の課題

co-Sound の課題として，第一に AR オブジェクトの永続的な状態管理が挙げられる．現在の co-Sound では，より低遅延な双方向通信を実現する WebRTC を利用しているが，サーバーレスな P2P 通信である．同一 LAN 内ではワンホップでの通信が可能となるため有意であるが，同時間帯に存在するメッシュネットワーク内ピア間でしかコンテンツ情報が保持されないという問題点がある．さらに端末間コネクションを各ピアが独

自に確立し，複数のピアでほぼ同時刻に送出されたオブジェクト操作情報のレンダリングはパケットの到達順に依存するため，空間情報が一意でなくなる可能性がある．タイムスタンプによる各ピアでのシーケンス処理は，高精度な時刻同期をブラウザ上で実現することが要求され実装は困難である．以上の理由から外部サーバによる永続的な状態管理システムが必要である．第二に，WebAR の性能限界が挙げられる．本研究で行った主観評価の結果にあるように，WebAR のマーカ検出精度は視聴者の QoE を下げる．ブラウザカーネルベース AR やアプリケーションベース AR であっても使用可能なリソースは限られることもあり，近年では AR にモバイルエッジコンピューティングを活用する研究も行われている [28]．これを WebAR にも応用し，現在モバイルで行っている AR 演算処理を外部リソースに分散し，高精度なトラッキング・リアルタイムなレンダリングが期待できるが，これは今後の課題とする．

参考文献

[1] M. Tsukada, K. Ogawa, M. Ikeda, T. Sone, K. Niwa, S. Saito, T. Kasuya, H. Sunahara, and H. Esaki. Software defined media: Virtualization of audio-visual services. In *2017 IEEE International Conference on Communications (ICC)*, pp. 1–7, May 2017.

[2] Sandvine. 2018 global internet phenomena report. `https://www.sandvine.com/hubfs/downloads/phenomena/2018-phenomena-report.pdf`. (Accessed on 12/19/2019).

[3] ITUR Rec. Itu r bs 2051-0 (02/2014). Advanced sound system for programme production. *Int. Telecommun. Union, Geneva, Swizerland*, 2014.

[4] 2023 年までの世界 ar/vr 関連市場予測を発表. `https://www.idc.com/getdoc.jsp?containerId=prJPJ45301519`. (Accessed on 12/19/2019).

[5] Sevda Küçük, Samet Kapakin, and Yüksel Göktaş. Learning anatomy via mobile augmented reality: Effects on achievement and cognitive load. *Anatomical Sciences Education*, Vol. 9, No. 5, pp. 411–421, 9 2016.

[6] Xiuquan Qiao, Pei Ren, Schahram Dustdar, Ling Liu, Huadong Ma, and Junliang Chen. Web AR: A Promising Future for Mobile Augmented Reality—State of the Art, Challenges, and Insights. *Proceedings of the IEEE*, Vol. 107, No. 4, pp. 651–666, 4 2019.

[7] jeromeetienne/ar.js: Efficient augmented reality for the web - 60fps on mobile! `https://github.com/jeromeetienne/AR.js`. (Accessed on 12/23/2019).

[8] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*, pp. 85–94, Oct 1999.

[9] New in chrome 79 | web | google developers. `https://developers.google.com/web/updates/2019/12/nic79#webxr`. (Accessed on 12/26/2019).

[10] Webxr device api. `https://www.w3.org/TR/2019/WD-webxr-20191010/`. (Accessed on 01/13/2020).

[11] Web audio api. `https://www.w3.org/TR/webaudio/`. (Accessed on 01/13/2020).

[12] Webrtc 1.0: Real-time communication between browsers. `https://www.w3.org/TR/webrtc/`. (Accessed on 09/15/2019).

[13] Antonin Marechal, Ewa Janczukowicz, and I M T Oln. TURN servers impacts over WebRTC QoE in 4G network. In *19th International ICIN Conference-Innovations in Clouds, Internet and Networks*, pp. 53–58, 2016.

[14] Iván Santos-González, Alexandra Rivero-García, Tomás González-Barroso, Jezabel Molina-Gil, and Pino Caballero-Gil. Real-time streaming: A comparative study between rtsp and webrtc. In Carmelo R. García, Pino Caballero-Gil, Mike Burmester, and Alexis Quesada-Arencibia, editors, *Ubiquitous Computing and Ambient Intelligence*, pp. 313–325, Cham, 2016. Springer International Publishing.

[15] Takashi Kasuya, Manabu Tsukada, Yu Komohara, Shigeki Takasaka, Takuhiro Mizuno, Yoshitaka Nomura, Yuta Ueda, and Hiroshi Esaki. Livration: Remote vr live platform with interactive 3d audio-visual service. In *IEEE Games Entertainment & Media Conference (IEEE GEM) 2019*, pp. 1–7, Yale University, New Haven, CT, U.S., 2019.

[16] 加藤慎, 池田友洋, 川守田光昭, 塚田学, 江崎浩. Web360²: インタラクティブな 3d 視聴体験を提供する web アプリケーション. Technical Report 14, 東京大学, Bitlet 合同会社, 東京大学, 東京大学, oct 2019.

[17] Cristina Fenu and Fabio Pittarello. Svevo tour: The design and the experimentation of an augmented reality application for engaging visitors of a literary museum. *International Journal of Human-Computer Studies*, Vol. 114, pp. 20 – 35, 2018. Advanced User Interfaces for Cultural Heritage.

[18] A. B. Tillon, I. Marchal, and P. Houlier. Mobile augmented reality in the museum: Can a lace-like technology take you closer to works of art? In *2011 IEEE International Symposium on Mixed and Augmented Reality - Arts, Media, and Humanities*, pp. 41–47, Oct 2011.

[19] Dieter Schmalstieg, Anton Fuhrmann, Gerd Hesina, Zsolt Szalavári, L. Miguel Encarnaçäo, Michael Gervautz, and Werner Purgathofer. The studierstube augmented reality project. *Presence: Teleoper. Virtual Environ.*, Vol. 11, No. 1, p. 33–54, February 2002.

[20] Mark Billinghurst, Hirokazu Kato, and Ivan Poupyrev. The magicbook: a transitional ar interface. *Computers and Graphics*, Vol. 25, No. 5, pp. 745 – 753, 2001. Mixed realities - beyond conventions.

[21] Tobias Lane, Blair Maclntyre, and Iker Jamardo Zuqaza. Massively Multiplayer Online Worlds as a Platform for Augmented Reality Experiences. *2008 IEEE Virtual Reality Conference*, pp. 67–70, 3 2008.

[22] 塚田学, 菰原裕, 粕谷貴司, 新居英明, 高坂茂樹, 小川景子, 江崎浩. SDM360^2：インタラクティブ 3d コンテンツの自由視聴点再生. 情報処理学会論文誌デジタルコンテンツ（DCON）, Vol. 6, No. 2, pp. 10–23, aug 2018.

[23] 菰原裕, 塚田学, 江崎浩, 曽根卓朗, 池田雅弘, 高坂茂樹, 新麗, 新善文. SDM ontology: Software defined media のメタデータ管理のための ontology. マルチメディア，分散協調とモバイルシンポジウム 2017 論文集, No. 2017, pp. 110–115, jun 2017.

[24] Web audio api. `https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API#Audio_spatialization`. (Accessed on 12/23/2019).

[25] 西堀佑, 多田幸生, 曽根卓朗. 遅延のある演奏系での遅延の認知に関する実験とその考察. 情報処理学会研究報告. [音楽情報科学], Vol. 53, pp. 37–42, dec 2003.

[26] S. Vlahovic, M. Suznjevic, and L. Skorin-Kapov. Challenges in assessing network latency impact on qoe and in-game performance in vr first person shooter games. In *2019 15th International Conference on Telecommunications (ConTEL)*, pp. 1–8, July 2019.

[27] M. Fiala. Artag, a fiducial marker system using digital techniques. Vol. 2, pp. 590 – 596 vol. 2, 07 2005.

[28] Ali Al-Shuwaili and Osvaldo Simeone. Energy-Efficient Resource Allocation for Mobile Edge Computing-Based Augmented Reality Applications. *IEEE Wireless Communications Letters*, Vol. 6, No. 3, pp. 398–401, 1 2017.

謝辞