基礎論文

モバイル MR システム構築のための機能分散型フレームワークの設計と実装

柴田 史久[*1]　松田 祐樹[*1*2]　レ ヴァン ギア[*1]
川端 大輔[*1]　山﨑 賢人[*1]　木村 朝子[*1]

Design and Implementation of a Distributed Framework for Mobile Mixed Reality Systems

Fumihisa Shibata[*1], Yuki Matsuda[*1*2], Nghia Van Le[*1],
Daisuke Kawabata[*1], Kento Yamazaki[*1], and Asako Kimura[*1]

Abstract --- This paper describes design and implementation of a distributed framework for creating mobile mixed reality (MR) systems. The goal of the framework is providing the same MR space for a variety of mobile devices which connect via wireless network. This paper discusses the following three topics which are essential for designing our framework: system architecture for supporting diverse mobile devices, an easy-to-implement script language for developing applications, and a communication method which takes into account the operability of the users. We implemented our framework based on the proposed design and investigated its performance empirically. As a result, we confirmed that various kinds of mobile devices can share the same MR space using our framework.

Keywords: mixed reality, mobile devices, distributed framework, script language

1 はじめに

スマートフォンに代表されるモバイル機器の急速な普及を背景に，モバイル機器を利用した複合現実感(Mixed Reality; MR)システムへ注目が集まりつつある．特に近年は，モバイル機器で利用可能な位置合わせ手法が多数提案され[1-7]，本格的なモバイル MR システムの登場に多大な期待が寄せられている．しかるに，この種のシステムの普及を念頭に置くと，アプリケーションを個別に一から開発することは無駄が多い．そのため，アプリケーション開発を容易にする仕組みが今後のモバイル向け MR の発展においては重要な鍵となる．

このような背景から我々は，フィーチャーフォン，スマートフォン，ウエアラブルコンピュータなどの様々なモバイル機器において利用可能なモバイル型 MR システムのためのフレームワークの構築を目指している．周知のとおり，スマートフォンを中心にここ数年でモバイル機器は大きな進化を遂げており，Google Glass などの新たなデバイスの登場からも分かるように，今後も性能やスタイルなどで革新的な技術進歩が見込める．そのため，現状のハードウエアやソフトウエア環境に焦点をあてた開発は無駄が大きく，将来の技術革新をバリエーションとして吸収できるようなフレームワークの実現が望ましい．

このような前提の下で我々は，以下の 3 点を要求仕様としたモバイル機器向けの MR システム構築のための機能分散型フレームワークを設計・実装した．

(1) コンテンツの動きが同期した MR 空間を同時に複数のモバイル機器で共有可能である
(2) モバイル機器の種類や性能差を吸収可能である
(3) アプリケーション開発が容易になるような仕組みを有する

本論文では，提案フレームワークの設計と実装を通じて得られた知見について報告する．以降，2 章で関連研究を概観したのち，3 章において提案フレームワークの設計方針やシステム・アーキテクチャなどについて述べる．4 章では実装結果について考察し，5 章においてまとめと今後の展望を述べる．

2 関連研究

モバイル向けの複合現実感システムを実現するためのフレームワークとしては，既にいくつの試みがある．Bauer らは，ウエアラブルコンピュータのためのフレームワークとして DWARF(Distributed Wearable Augmented Reality Framework)[8]を提

*1　立命館大学大学院情報理工学研究科
*2　現在，任天堂(株)
*1　Graduate School of Information Science and Engineering, Ritsumeikan University
*2　Nintendo Co., Ltd.

案した．DWARF は，予め AR に必要な位置合わせなどの機能をモジュールとして準備し，各モジュールを自由に組み合わせることでアプリケーション開発が容易になるような設計となっている．しかしながら，提示するコンテンツの扱いについては各モジュールを連携させるようなプログラムを一から書く必要があり，複数端末間のコンテンツの同期についても検討されていない．一方，本稿で提案するフレームワークは，複数の端末間でコンテンツの同期がとれているようなアプリケーションの作成を想定し，そこにおけるコンテンツの扱いを容易にすることに主眼をおいており，MR 空間内のコンテンツの位置，姿勢やユーザからの入力に対するイベント処理をプログラムするだけで，アプリケーションが作成できる点が異なっている．

Schmalstieg らは，ハンドヘルドデバイスのためのフレームワークである Studierstube ES を提案している[9]．当該フレームワークは，スマートフォンから UMPC までをターゲットとしており，スタンドアロンで動作するアプリケーションやクライアントサーバによる複数ユーザ間の通信をサポートしているが，フィーチャーフォンなどの処理能力の低い機器は考慮していない．

Tinmith-evo5 は，Piekarski らの手によって開発された AR アプリケーション構築のためのソフトウエア・アーキテクチャである[10]．Tinmith-evo5 では，シーングラフなどを扱える高レベルな描画コンポーネントや，位置姿勢を推定するトラッカーなどを，階層構造を持つライブラリとして提供しているが，ノート PC と HMD，入力用グローブなどを組み合わせたようなハードウエアを想定しており，昨今のモバイル機器向けの設計とはなっていない．特に屋外を対象としたオンサイトでの AR コンテンツの編集に向いたシステムへと発展している[11]．

AR-PDA というプロジェクトでは，サーバとクライアント間で画像をストリーミング方式で送受信し，AR を実現するという方式が試みられた[12]．

最近の研究事例としては，MacIntyre らによる Argon AR Web Browser がある[13]．これは，WWW の様々なサービスと AR を連携させることに主眼をおいたアーキテクチャの提案となっている．

また最近では，ゲームエンジンと呼ばれるゲーム開発向けのミドルウエアやオーサリングツールを組み合わせたソフトウエアが数多く公開されており，その中には AR をサポートするようなものも存在する．Unity Technologies 社の Unity[14]がその一例であるが，ゲームエンジンという性質上，コンテンツの共有などを実現するためにはネットワーキングに対応した処理をコーディングする必要がある．

一方，モバイル機器向けの位置合わせ手法については，マーカを利用する手法[1,15]や平面上の自然特徴点を利用する手法[2,3,5]，3 次元の特徴点を利用する手法[4,7]，パノラマ画像を用いるもの[6]など数多くの手法が提案されている．これらは，基本的にはフレームワークではなく，AR を実現する上で利用するライブラリにすぎない．

3 機能分散型フレームワーク

3.1 設計方針

一般に普及しているモバイル機器は多岐にわたり，その性能には著しい差異が存在する．本研究では，据置型 PC で現在達成されているような最高水準の MR システムをモバイル機器へ搭載することを目指すのではなく，モバイル機器の性能差に起因するシステム実現上のトレードオフを容認し，利用形態やモバイル機器の性能に応じて情報提示のスタイルを変更可能な枠組みの実現を目指す．

システムの利用形態としては，異なるモバイル機器を保持する複数のユーザが同じ空間内で協調しながら作業を実施するような状況を想定する．これには，複数の作業従事者が大型機器のメンテナンスを実施するようなものからショッピングモールなどでのナビゲーションやガイド，テーマパークでのアトラクションやゲームなどを含んでいる．

3.2 システム・アーキテクチャ

1 章で述べた要求仕様(1)～(3)を満たすフレームワークのシステム・アーキテクチャを**図 1** のように設計した[16]．

システムは，MR サーバ，クライアント，メディエータ，シンクライアントの 4 つの構成要素から成り，各々は位置姿勢推定など複数のモジュールから

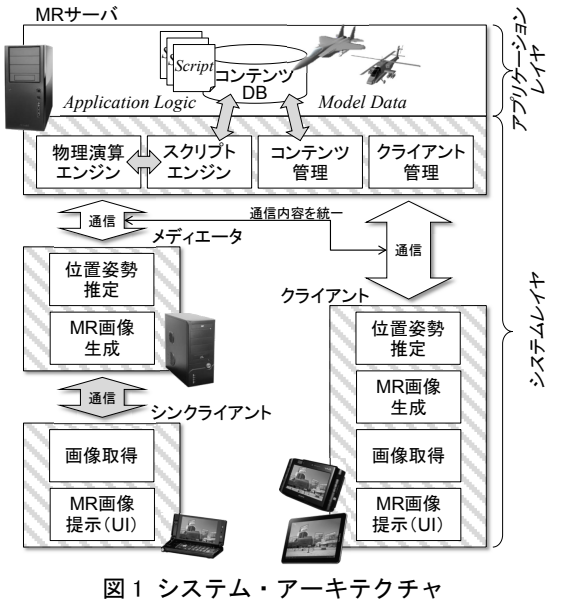


図 1 システム・アーキテクチャ
Fig.1 System Architecture

構成される．システム・アーキテクチャの設計では，(1)を満たすためにクライアントサーバモデルを採用し，MR 空間のコンテンツはすべて MR サーバで一元管理することとした．(2)については，モバイル機器を性能や用途に応じて，クライアントもしくはシンクライアントいずれかのコンフィグレーションに分けた上で，両者の差異をメディエータで吸収する仕組みを取り入れた．また，各機能をシステムレイヤとアプリケーションレイヤに分離してシステムレイヤの再利用性を高めると同時に，コンテンツを制御するためのスクリプト言語を設計することで，(3)を達成する．それぞれの役割は以下の通りである．

【MR サーバ】

MR サーバは，MR 空間の管理・制御およびクライアントの管理を担う．MR 空間内のコンテンツ（仮想オブジェクト）を一元管理し，ネットワークを介してクライアントとメディエータにこれを提供することで，MR 空間の共有を実現する．コンテンツの動きやユーザのインタラクションに対する処理は，後述するスクリプト言語によって記述され，MR サーバ内のスクリプトエンジンによって処理される．物理法則に則った動きをコンテンツにさせたい場合には，スクリプトエンジンに物理演算エンジンを連携させることもできる[17].

【クライアント】

比較的処理能力の高いモバイル機器を想定したコンフィグレーションである．画像取得，位置姿勢推定，MR 画像の生成および MR 画像の提示機能を有し，これらの処理をリアルタイムで実行する．MR 画像は，MR サーバから送信される MR 空間のコンテンツの情報（以降，MR 情報と呼ぶ）に基づいて生成される．また，ユーザからのインタラクションやクライアントの位置姿勢情報（以降，クライアント情報と呼ぶ）を MR サーバに送信する．

【シンクライアント】

フィーチャーフォンのような処理能力の低いモバイル機器を想定したコンフィグレーションである．画像取得および MR 画像提示の機能のみを有し，位置姿勢推定や MR 画像生成などの負荷の重い機能はメディエータに委託する．MR 画像の提示は，静止画像で行われる．

【メディエータ】

メディエータは，シンクライアントと MR サーバを仲介し，シンクライアントに代わって位置姿勢推定，MR 画像生成の処理を実行する．メディエータによって，MR サーバからはシンクライアントの存在が隠蔽され，MR サーバから送信する通信内容を統一できる．

以降では，最初に処理の流れを説明したのちに，システム設計の主要な部分であるコンテンツ制御の仕組み，スクリプト言語の設計，通信機構の設計，アニメーションの同期処理について順に説明する．

3.3 処理の流れ

MR サーバとクライアントにおける MR 画像提示の流れを**図 2** に示す．MR サーバは，初期化処理ののち，3 つのスレッドを並列に実行する．1 つは，アプリケーションロジックにあたるスクリプトを実行するスレッドである．残る 2 つは，クライアント情報を受信するスレッドと MR 情報を送信するスレッドである．MR サーバでは，常時スクリプトエンジンによってコンテンツの状態が更新され，クライアントからのリクエストにしたがって MR 情報として送信される．一方，クライアントでは，最初に MR サーバからモデルデータなどを取得した上で，3 つのスレッドを並行に実行する．1 つは，MR を実現する上で要となる現実世界の光景を取得し，MR

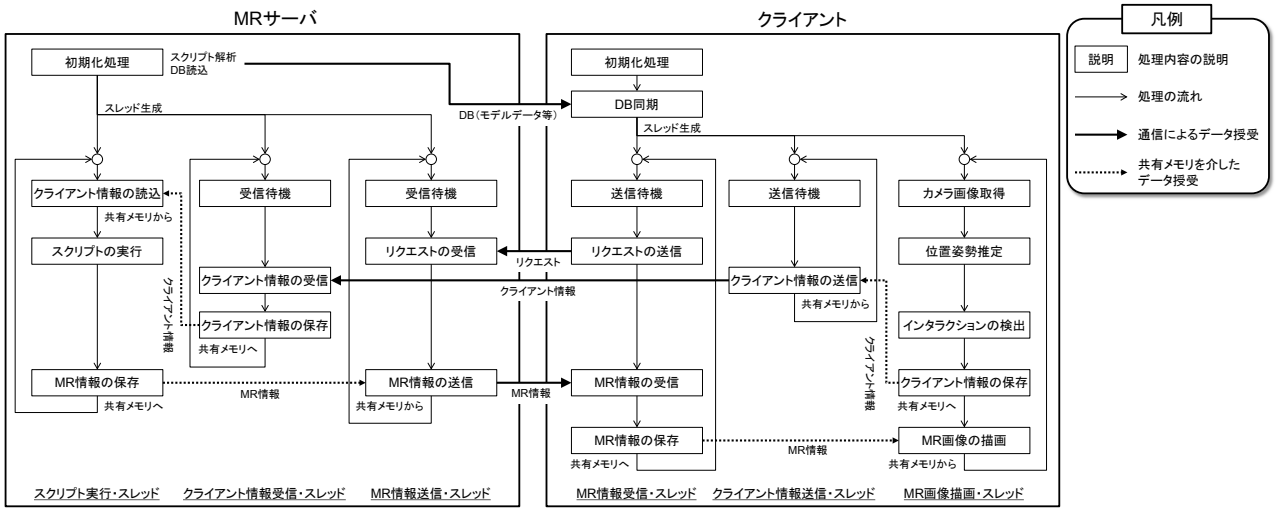


図 2 MR サーバとクライアントの処理の流れ

Fig.2 Flowchart of the MR server and the client

画像を提示するスレッドである．残る 2 つは，クライアント情報を送信するスレッドと，定期的に MR 情報を受信するスレッドである．

メディエータとシンクライアントを組み合わせた処理の流れを**図 3** に示す．この図は，図 2 のクライアント部分（右半分）に該当し，凡例は図 2 と同様である．通信内容から分かるように，MR サーバからは，メディエータとクライアントを同一のものとみなすことができる．メディエータは，シンクライアントからカメラ画像を取得すると，それを元にシンクライアントの位置姿勢を推定する．その後，位置姿勢に基づいて MR 画像を生成し，シンクライアントへ送信する．シンクライアントでは，受け取った MR 画像をユーザに対して提示する．

3.4 コンテンツ制御機構

一般に，MR を実現するためには，1)仮想世界にコンテンツを配置し，2)現実世界でのカメラ位置と姿勢を推定した上で，3)推定結果に基づいて現実世界の光景に仮想世界を重畳描画するという処理が必要となる．このうち，2)については，どのような位置合わせ手法を利用するかに依存し，3)はどのようなアプリケーションでも基本的には同じとなる．したがって，MR アプリケーションの開発を容易にするには，1)を如何にして簡略化するかが重要となる．換言すれば，仮想世界中におけるコンテンツの位置や姿勢などを簡単に制御できるような仕組みが必要となる．本研究では，これをコンテンツ制御機構と呼び，以下の 3 点に重点を置いて設計した．

- コンテンツの複雑な動きを表現可能
- ユーザからのインタラクションなどによってコンテンツの動きに変化をつけることが可能
- コンテンツを制御するための記述が容易

図 1 で示した MR サーバでは，コンテンツ管理，クライアント管理，スクリプトエンジン，物理演算エンジンという 4 つのモジュールを組み合わせたものがコンテンツ制御機構に該当する．

提案フレームワークで扱うコンテンツは，ユーザが持つクライアントと CG などを表す仮想オブジェクトから成る．前者はクライアント管理，後者はコンテンツ管理の各モジュールによって管理される．仮想オブジェクトは，1 つ以上のコンポーネントの集合として構成される．コンポーネントとしては，3 次元モデル，サウンド[18]，ボーンアニメーション，パーティクル[19]を扱うことができ，これらがルート・コンポーネントを起点とする木構造を成すことで 1 つの仮想オブジェクトになる．また，現実物体と同等の形状を有する透明な仮想オブジェクトを配置することで，仮想オブジェクトを描画する際の陰面消去や衝突判定などが実現できる．

仮想オブジェクトには，**表 1** に示すプロパティが存在し，これらのプロパティを，次節で述べるスクリプト言語を用いて更新することで，位置や姿勢などを制御する．またクライアントには，**表 2** に示すプロパティがあり，スクリプトから読み出して利用できる．スクリプトエンジンは，スクリプトの解析・実行を担うモジュールである．

3.5 スクリプト言語

本フレームワークで設計したスクリプト言語は，オブジェクト指向言語である Java や C++と類似した記述法を採用した．一般的なプログラミング言語をベースに文法を設計することで，アプリケーション開発者は初めて本言語を使用する際でも，本言語独自の概念を理解するだけで容易に開発が可能となる．しかし，本言語は通常のプログラミング言語と

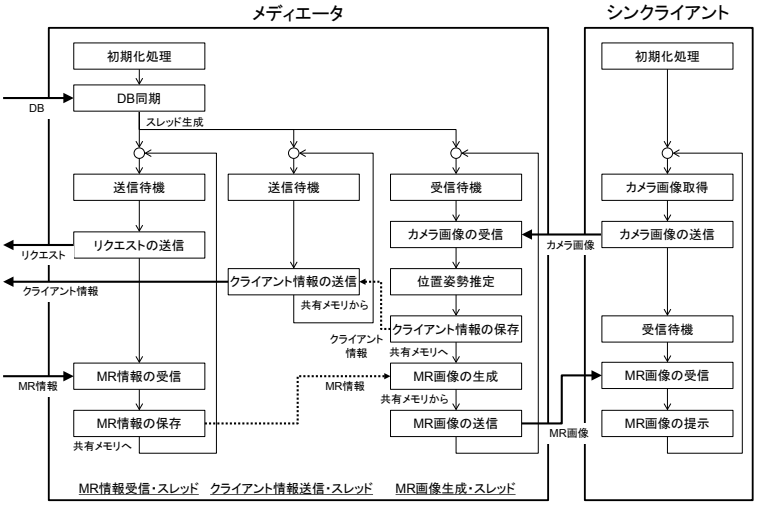


図 3 メディエータとシンクライアントの処理の流れ
Fig.3 Flowchart of the mediator and the thin-client

表 1 仮想オブジェクトのプロパティ
Table 1 Properties of a virtual object

型とプロパティ名	意味
`int id`	ID
`int classId`	クラスの ID
`Position3D pos`	位置
`Orientation3D ori`	姿勢（各軸）
`Scale3D scale`	スケール（各軸）
`int presen`	可視(0)・半透明(1)・非表示(2)・不可視(3)

表 2 クライアントのプロパティ
Table 2 Properties of a client

型とプロパティ名	意味
`int id`	ID
`int classId`	クライアントの種類
`Position3D pos`	位置
`Orientation3D ori`	姿勢（各軸）

表 3 変数の型
Table 3 Variable Types

型名	種類	サイズ
int	整数型	4 byte
float	浮動小数点型	4 byte
boolean	論理型	1 byte
string	文字列型	可変長

表 4 演算子
Table 4 Operators

種類	記号
代入演算子	=, +=, -=, *=, /=, %=
算術演算子	+, -, *, /, %
比較演算子	>, <, >=, <=, ==, !=
論理演算子	&&, ||
その他	new, isInstanceOf

表 5 制御構文
Table 5 Statements

種類	記法
選択文	if 文，if～else 文，switch 文
繰返し文	for 文，while 文
分岐文	break 文，continue 文，return 文

異なり汎用性を実現することは目的としない．本言語が持つ機能は，MR においてコンテンツを制御することに特化し，それらの機能を容易に利用できる設計とした．**表 3～5** に，変数の型，演算子，制御構文など言語仕様の一部を示す．

本言語では，以下に示す 3 つの基本クラスを用意し，アプリケーション開発者はこれらのクラスを継承したサブクラスに必要なコードを追加してアプリケーションを作成する．各クラスのフィールドやメソッド内で必要に応じて変数などを宣言することもできる．

【MRWorld クラス】

オブジェクトやクライアントが存在する MR 空間全体の情報を格納するクラスである．クラスメソッドによって MR 空間に対するオブジェクトの追加や削除などができる．また，クライアントの接続状態が変化した場合やオブジェクト同士の衝突などのイベントが発生した際には，コールバック用メソッドが呼び出される．**表 6** にメソッドの一部を示す．

【MRObject クラス】

オブジェクトの情報を格納するクラスである．位置や姿勢などのプロパティを格納するフィールド変数が予め定義されており，これらの値を変更することでオブジェクトを移動することができる．MR 空間の更新間隔毎にコールバック用メソッドが呼び出されるため，オブジェクトの移動などの処理はこのメソッド内に記述する．**表 7** にメソッドの一部を示す．

表 6 MRWorld クラスのメソッド
Table 6 Methods of MRWorld class

クラスメソッド	
メソッド定義	意味
void addObject(MRObject obj)	オブジェクトの追加
void removeObject (MRObject obj)	オブジェクトの削除
コールバック用インスタンスメソッド	
メソッド定義	意味
void connectedClientCallback (int clientClassId)	新しいクライアントの接続時に呼出し
void onCollisionCallback (MRObject obj1, MRObject obj2)	オブジェクト同士の衝突発生時に呼出し

表 7 MRObject クラスのメソッド
Table 7 Methods of MRObject class

インスタンスメソッド	
メソッド定義	意味
Position3D getPosition()	位置を取得
void setPosition (float x, float y, float z)	位置を設定
setAsBox (float h, float w, float l)	衝突形状を直方体に設定
コールバック用インスタンスメソッド	
メソッド定義	意味
void updateCallback(float dt)	MR 空間更新時に呼出し．dt は前回呼出しからの経過時間.

表 8 Client クラスのメソッド
Table 8 Methods of Client class

コールバック用インスタンスメソッド	
メソッド定義	意味
void updateCallback(float dt)	MR 空間更新時に呼出し．dt は前回呼出しからの経過時間.
void onInteractionCallback (int interaction_type)	インタラクション発生時に呼出し

【Client クラス】

クライアントの情報を格納するクラスである．キー入力などのインタラクションを検出すると，コールバック用メソッドが呼び出されるため，インタラクションに対する処理はこのメソッド内に記述する．**表 8** にメソッドの一部を示す．

これらの基本クラスに加えて，3 次元の位置情報を格納するための Position3D, 姿勢情報を格納する Orientation3D，三角関数などの数値処理に必要なメソッドを定義した Math クラスなどがある．

3.6 通信機構

ここまで述べてきたコンテンツ制御機構は，MR サーバ上で実行される．ユーザに対して MR 空間を提示するのはクライアント側になるため，単純に考えるとクライアントの画面更新間隔に合わせて MR

情報をMRサーバからクライアントへと送信する必要がある．しかしながら，送信するMR情報のデータ量と通信帯域を考慮すると，これを複数のクライアントに対して常に実現するのは難しい．そこで本研究では，MRサーバ上ではスクリプトエンジンによって常時，例えば秒間30回などのペースでMR空間の更新を行う一方，クライアントはMRサーバからMR空間の情報を一定間隔 ΔT で取得する定期通信方式を採用する．その上で，クライアントは次の通信までの間，前回サーバから取得した情報と新たにサーバから取得した情報を補間することで，仮想オブジェクトを提示する．

補間処理には，3次ベジエ曲線のアルゴリズムを応用する（**図4**参照）．MRサーバから送られた今回と前回の位置情報 P_t 及び $P_{t\text{-}3dt}$（ただし，ΔT=3dt）対し，各点の速度ベクトル情報 V_t 及び $V_{t\text{-}3dt}$ から制御点 $P_{t\text{-}dt}$ と $P_{t\text{-}2dt}$ を生成することでサーバにおける仮想オブジェクトの動作を滑らかに再現する．この際，例えばボールが跳躍する際に発生する速度ベクトルが急激に変化する点を尖点と定義し，このような点が発生した場合には，尖点を含む2つの曲線区間を補間するアルゴリズムを考案した[17]．

また，コンテンツ制御をMRサーバで行うにあたって考慮すべき事項にユーザからのインタラクションの処理がある．インタラクションに対する即応性という意味では，クライアントで処理することが望ましいが，これを行うとクライアント間でコンテンツの不整合が発生する．そのため，インタラクションの処理をMRサーバで実行しつつ，できるだけ応答が早くなる方式を検討する必要がある．本研究では以下の3通りの方式について検討した（**図5**参照）．

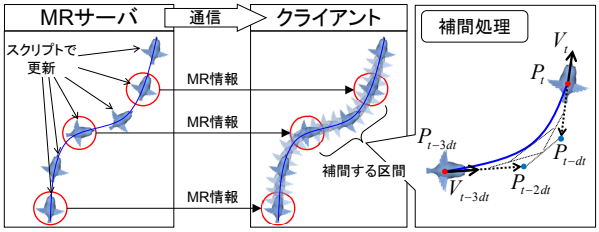


図4 補間処理の概要
Fig.4 The outline of interpolation

【通信方式(A)】

文献[16]にて提案した方式で，クライアントにMRサーバと定期的に通信するスレッドを1つ用意し，クライアントからMRサーバに送る通信にインタラクションの情報を乗せる．仕組みが簡単であり，複数のクライアントに対しても安定して動作するが，インタラクションが反映されるまでの遅延時間が長くなる．遅延時間を短くするために定期通信間隔を短くすると，MR情報の受信回数も増えるため，通信量が増加する．

【通信方式(B)】

文献[20]にて提案した方式で，MRサーバからMR情報を受信する定期通信スレッドに加えて，インタラクションが発生するたびに，インタラクションの情報送信専用のスレッドを立ち上げる方式である．通信方式(A)に比べてインタラクションの情報が直ちにMRサーバに送信されるため，インタラクション結果を反映するまでの遅延時間が短縮される一方で，連続するインタラクションが発生した場合などに通信回数が増加するため効率が悪くなる．

【通信方式(C)】

通信方式(B)の利点と欠点を踏まえて新たに考案した方式で，MRサーバへインタラクション情報を定期的に送信するスレッドと，MRサーバからMR情報を定期的に受信するスレッドを設ける方式である．クライアントで発生したインタラクションはキューに蓄積された上で，定期的に送信される．また，別のスレッドで定期的にMR情報を受信する．インタラクション情報の送信は通信量が多くないため，

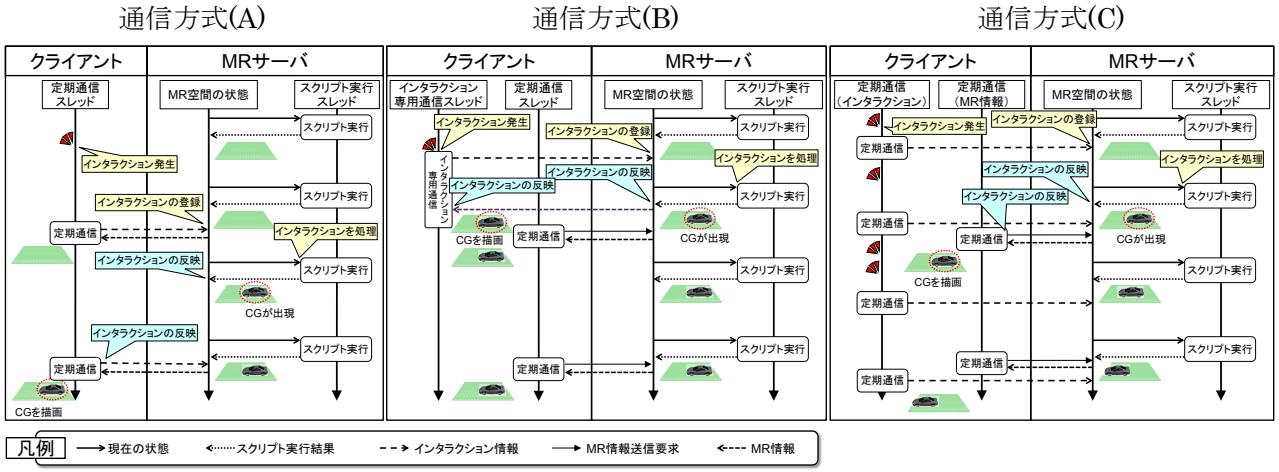


図5 通信方式の比較
Fig.5 Comparison of communication methods

インタラクション送信用スレッドの定期通信間隔を短く設定すると，通信方式(A)に比べてインタラクション反映までの遅延時間を短縮できる．また，インタラクションをキューに蓄積するため，通信方式(B)に比べて連続するインタラクションが発生しても通信量は一定の範囲内に収めることができる．

本研究で設計したフレームワークでは，通信方式(C)を採用しており，インタラクション情報と MR 情報を分離した非同期である 2 つの定期通信と仮想オブジェクトの補間手法によって複数のモバイル機器間における MR 空間の共有を実現している．

3.7 ボーンアニメーションと同期

3.4 節にて述べたように提案フレームワークで扱うことのできる仮想オブジェクトには，ボーンアニメーションがある[19]．これは，既存のソフトウエアで作成された 3 次元モデルとモーションのデータから成り，これをスクリプトで制御することによってアニメーションを表現する．

仮想オブジェクトにおいてボーンアニメーションを扱う場合，3 次元モデルを親コンポーネントとし，モーションデータを子コンポーネントとして仮想オブジェクトを構成するものとする．

ボーンアニメーションを制御する上で最低限必要な操作は「再生」「一時停止」「停止」が考えられる．これに加えて連続して再生するための「繰り返し回数」やモーションの速度を制御する「再生速度」というものが考えられる．これらを，モーションデータを保持するコンポーネントのプロパティとして**表 9** のように設定し，プロパティの値をスクリプトから変更することでボーンアニメーションを制御する．

提案フレームワークにおいてサウンドやボーンアニメーションといった時間軸を有するコンテンツを扱う場合，MR サーバとクライアント間における通信遅延が問題となる．通信遅延は，クライアント毎に異なるため，各クライアントが MR サーバから情報を受け取った時点で再生を開始するとクライアント間にずれが生じる．そこで，MR サーバとすべてのクライアントで予め時刻同期を行った上で，MR サーバから送信する MR 情報に時刻情報を付加し，クライアント側では受け取った時刻情報を元に前述の補間処理やボーンアニメーションおよびサウンドの再生タイミングを同期する機構を設計した．

表 9 モーションデータのプロパティ
Table 9 Properties of motion data

型とプロパティ名	意味
`int state`	再生(0)・一時停止(1)・停止(2)
`int loop`	繰返し再生回数
`float speed`	再生速度

MR サーバとクライアントは，予めクライアント起動時に時刻を同期する．時刻同期は NTP と同様に往復の通信時間を計測することで行う．

前節で述べたように，クライアントは MR サーバから定期的に受信した MR 情報に基づいて補間を行いながら仮想オブジェクトを描画する．このとき通信遅延に起因するクライアント間のずれをなくすために，サーバに対して遅延する時間 *delay*(ms)を全クライアントで統一された値として設定する．その上で，クライアントは現在の時刻から *delay* 分遡った MR 空間を，補間処理を用いて描画する（**図 6** 参照）．同じ空間を共有するという前提からクライアント間で通信経路に大きな差異はないと考えられるため，*delay* の値はその空間における通信状況を予め確認した上で設定する．

クライアントでサウンドやボーンアニメーションの再生時刻を統一するために，サーバからクライアントへと送信する情報には，再生開始時刻を付加して送信する．クライアントは，前述の *delay* 分を考慮の上で，サーバから受信した再生開始時刻になれば再生を開始する．情報を受信した時刻が再生開始時刻を過ぎていれば，現在の時刻との差分を計算し，途中から再生を行う．この仕組みにより，通信にかかる時間の差に関係なく，全てのクライアントで同一時刻に再生することが可能となる．

4 性能評価実験

4.1 実験機器

ここまでに述べてきた設計方針に沿って提案フレームワークを実装し，性能評価を試みた．まずは MR サーバとクライアントを実装し，基本性能について実験を試みた．

MR サーバは，Windows 7 Professional (x64)を OS とする PC 上に C++で実装した．主な仕様は，CPU が Intel Core i7 3960X Extreme Edition，搭載メモリが 16GB である．クライアントは，iOS 7 を OS とする端末上に Objective-C と C++で実装した．実験に用いたのは，iPad Air および iPhone5s である．また一部の実験では，クライアントとして

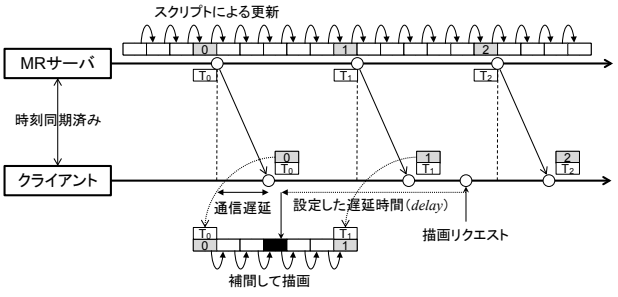


図 6 クライアントの同期処理
Fig.6 Synchronization process of the client

Windows 7 Professional (x64)を OS とする PC も用いた．ネットワーク構成としては，Wi-Fi ルータ NEC PA-WR9500N-HP に，MR サーバはギガビットイーサによる有線で接続し，クライアントは iOS, Windows ともに IEEE802.11n（5GHz 帯）による無線で接続する．位置合わせには ARToolKit[21]を用いる．

4.2 実験と考察

【実験 1】

クライアント 1 台を MR サーバに接続した際のクライアント情報および MR 情報の伝送時間を測定した．MR 空間に配置するオブジェクト数は，10 個刻みで 20～200 個まで条件を変化させて測定した．計測は，100 回行い，平均値，最大値，最小値をグラフ化した．クライアント情報の送信間隔は 15ms，MR 情報の送信間隔は 100ms とした．

【実験 1 の結果と考察】

実験 1 の結果を**図 7** に示す．MR 情報については，(a)に示したようにオブジェクトの数が増えると，それに比例して伝送時間が増加する．平均値を見る限りではオブジェクト数が 200 個でも 20ms 程度であり，十分な速度が出ている．ただし，最大値としては 45ms 程度の値が出ているため，クライアントにおける補間処理を省くことは難しい．一方，クライアント情報の伝送時間については，(b)に示したように最大値がばらつくものの平均値は 6ms 未満であり，送信間隔を短くすることでインタラクションの遅延を抑えることも可能である．

【実験 2】

実験 1 の条件に対して，クライアント数を 2～4 台および 10 台に変更して伝送時間を測定した．2～4 台の実験ではすべて iOS のクライアントを，10 台の実験では iOS を 4 台と Windows のクライアントを 6 台用いた．

【実験 2 の結果と考察】

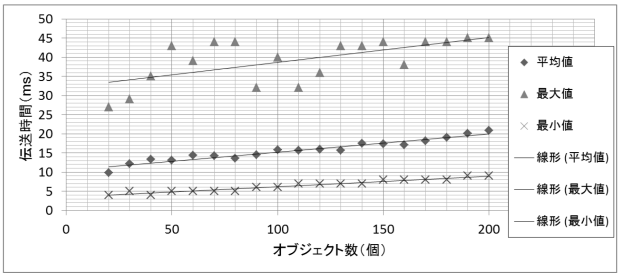
実験 2 の結果を**図 8** に示す．MR 情報については，(a)に示したようにオブジェクトの数が増えると，それに比例して増加する．クライアントが 10 台の場合は，1～4 台の場合と比較すると伝送時間は長くなっている．原因は，クライアントの台数に比例して通信量が増加しているためだが，台数に比例して伝送時間が長くなっているわけではない．また，クライアント情報については，台数が 10 台の場合に若干伝送時間が長くなっているが，十分に送信間隔の中に収まっている．

【実験 3】

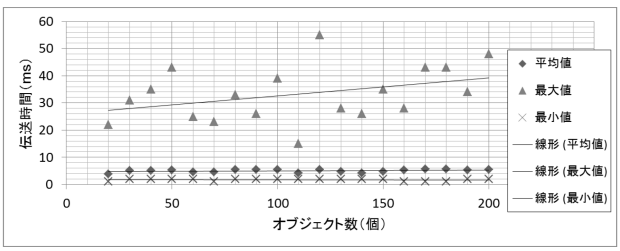
クライアントにおけるインタラクションの応答性能について確認する．具体的には，クライアントにおいて時刻 t_0 にタッチインタラクションを発生させてから，MR サーバでスクリプトが実行されインタラクションが反映される時刻 t_1，およびクライアントにおいてインタラクションの結果が描画される時刻 t_2 を調べた．クライアントは 1～4 台，オブジェクトの数は 20 個，クライアント情報の送信間隔は 15ms，MR 情報の送信間隔は 100ms とした．

【実験 3 の結果と考察】

結果を**表 10** にまとめた．t_1 - t_0 が MR サーバへのインタラクションの反映にかかる時間，t_2 - t_0 がクライアントにおいてインタラクション結果が画面に描画されるまでの時間である．MR サーバに接続する



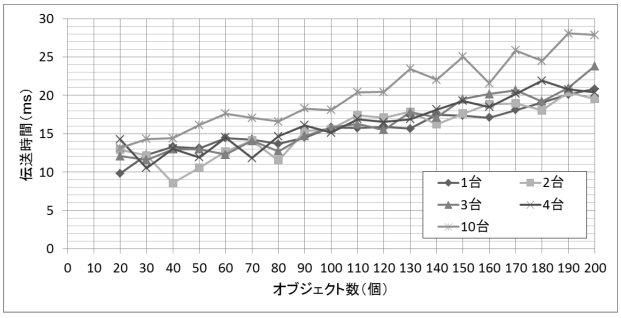
(a) MR 情報の伝送時間



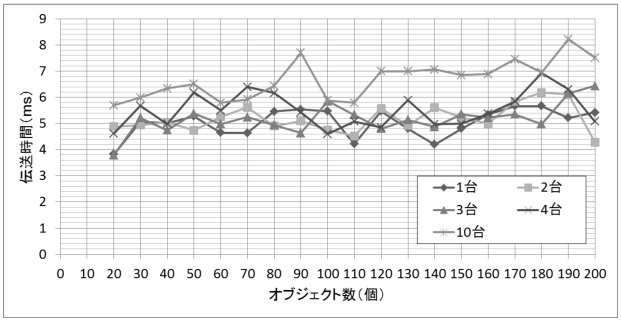
(b) クライアント情報の伝送時間

図 7 実験 1: クライアント 1 台の場合の伝送時間

Fig.7 Exp 1: Transmission time for a single client



(a) MR 情報の伝送時間



(b) クライアント情報の伝送時間

図 8 実験 2: クライアントが複数の場合の伝送時間

Fig.8 Exp 2: Transmission time for multiple clients

台数が増えると，インタラクションを反映するまでの時間が長くなる傾向が見られた．この理由は，他のクライアントがネットワークを占有しているため，通信がしづらい状況になるためと推測できる．

【実験 4】

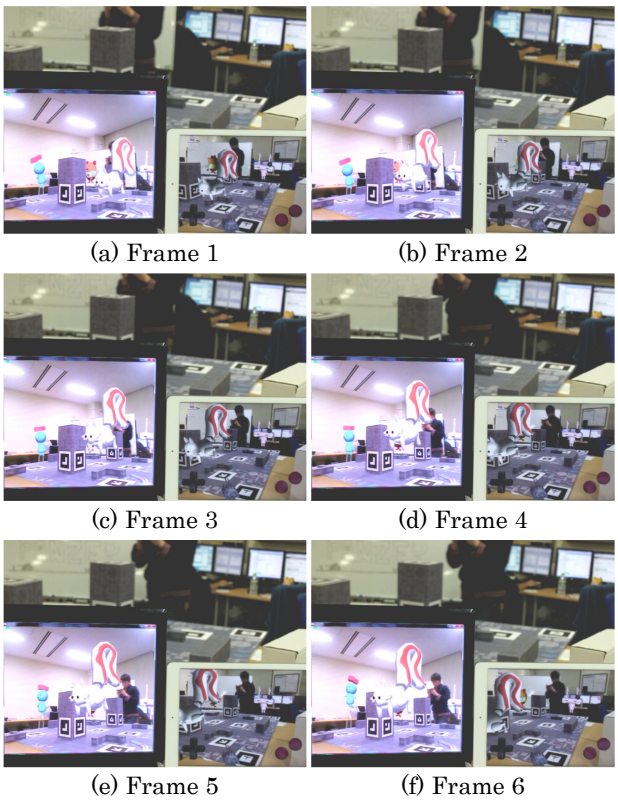
アニメーションの同期が意図した通りに動作するのかを iOS と Windows クライアントで確認した．ボーンアニメーションによって動作する CG キャラクタ「よむりす」を作成し，複数のクライアントで同期されているかを確認した．

【実験 4 の結果と考察】

図 9 に 2 台のクライアントの画面を並べて撮影した連続する静止画像を示す．これは，「よむりす」がボーンアニメーションによって奥から手前へと飛び跳ねている様子となっている．画像奥にいる人物の動きと各クライアントの画面に描画されている CG キャラクタの位置関係からわかるように，OS の異なる複数のクライアント間でアニメーションを含むコンテンツが共有できていることが確認できた．

表 10 実験 3: インタラクションの平均反映時間(ms)
Table 10 Exp 3: Average time of reflection of user interaction

クライアント	1 台	2 台	3 台	4 台
$t_1 - t_0$	23.9	50.8	44.5	63.3
$t_2 - t_0$	369.6	399.0	411.7	411.0



(a) Frame 1　(b) Frame 2
(c) Frame 3　(d) Frame 4
(e) Frame 5　(f) Frame 6

図 9 実験 4: アニメーションの同期結果
Fig.9 Exp 4: Synchronization of animation

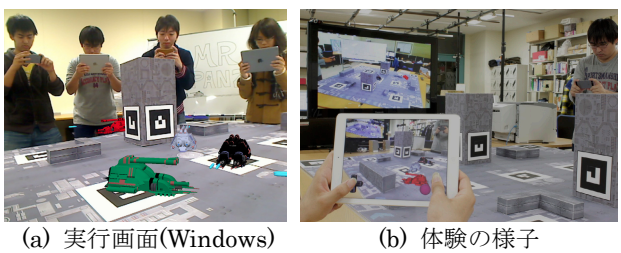
4.3 アプリケーションの試作例

実装したフレームワークを用いて MR Panzer という最大 4 名のユーザが参加できる複数人対戦型のゲームを試作した．各ユーザは，パネル上に配置された前後左右ボタンとミサイル発射ボタンで自機をコントロールして他ユーザの戦車を攻撃する．フィールド上には，実物体で作成されたビル(Building)や壁(Wall)などの障害物が存在し，実物体と同形状のマスクオブジェクトを配置することで，戦車やミサイルとの衝突判定を実現している．ゲーム中では BGM が流れており，ミサイルが戦車や障害物に衝突すると効果音が再生される．また，戦車には耐久度を設定しており，ミサイルによる攻撃を 5 回受けると自軍の陣地に送還されるといった最低限のゲーム要素などが組み込まれている．

表 11 は MR Panzer のために作成したスクリプトのクラスとそのコード行数である．全体で 1000 行ほどのスクリプトで空間共有型のゲームが作成できることを確認した．MR Panzer の開発者は提案フレームワークの知識やスクリプトの使用経験がある学生で，コーディングに要した時間は 15 時間ほどである．スクリプトも主に単純な条件分岐や代入文から構成されており，特別なプログラミングスキルを要求するようなものではない．**図 10** に MR Panzer を実行している様子を示す．図 10(b)の左奥にある観客

表 11 MR Panzer のコード行数
Table 11 Lines of code for MR Panzer

クラス名	親クラス	行数
TankWorld	MRWorld	222
ClientIphone	Client	146
ClientWindows	Client	15
TankA	MRObject	128
TankB	MRObject	128
TankC	MRObject	128
TankD	MRObject	128
Rocket	MRObject	64
Explosion	MRObject	24
Building	MRObject	16
Wall	MRObject	16
BGM	MRObject	16



(a) 実行画面(Windows)　(b) 体験の様子

図 10 MR Panzer を実行している様子
Fig.10 Scene of MR Panzer

用モニタにはフィールドの全体が表示されており，左手前のユーザが把持する iPad の画面と同期がとれていることがわかる．

5 むすび

本論文では，コンテンツの動きが同期した MR 空間を同時に複数のモバイル機器で共有可能な MR システムを構築するための機能分散型フレームワークを提案し，その設計と実装について述べた．具体的には，提案フレームワークの要となるコンテンツ制御の仕組みや専用に設計したスクリプト言語，MR サーバとクライアント間の通信機構などについて詳述した．さらに，設計方針に沿って提案フレームワークを実装し，その性能について評価した．その結果，種類の異なる複数のクライアントが接続する状況においても，意図した通り MR 空間の共有が実現できることを確認した．

アプリケーションの試作例としては，対戦型のゲームを選択し，実装を試みた．この題材を選んだのは，あくまでも同期に関する機能検証を第一義と考え，複数人で同じ MR 空間を共有し，かつインタラクションの応答性が重要な要素となるものだからである．このゲームは，スクリプトのコード行数としては約 1000 行と小規模ながら，高校生や大学 1 回生を対象としたデモンストレーションにおいて好評を博しており，提案フレームワークによって MR アプリケーションが簡単に開発できることを示すことができた．

提案フレームワークは，位置合わせ手法に依存しない設計となっているが，現時点で利用できるのは ARToolKit のみとなっている．今後は，自然特徴点を用いる手法などとも組み合わせた上で，フレームワークを利用したアプリケーションの事例を増やし，さらなる改良を進めていく予定である．

謝辞

本研究の開発・実装作業に携わった荻野翼氏に感謝の意を表する．本研究の一部は，科学研究費補助金（基盤研究(B) No. 22300046，研究代表：柴田史久）の支援による．

参考文献

[1] D. Wagner, T. Langlotz, and D. Schmalstieg: Robust and unobtrusive marker tracking on mobile phones, Proc. 7th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2008), pp 121 - 124 (2008.9)

[2] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg: Pose tracking from natural features on mobile phones, Proc. 7th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2008), pp. 125 - 134 (2008.9)

[3] D. Wagner, D. Schmalstieg, and H. Bischof: Multiple target detection and tracking with guaranteed framerates on mobile phones: Proc. 8th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2009), pp. 57 - 64 (2009.10)

[4] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg: Wide area localization on mobile phones, Proc. 8th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2009), pp. 73 - 82 (2009.10)

[5] G. Klein, and D. Murray: Parallel tracking and mapping on a camera phone, Proc. 8th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2009), pp. 83 - 86 (2009.10)

[6] C. Arth, M. Klopschitz, G. Reitmayr, and D. Schmalstieg: Real-time self-localization from panoramic images on mobile devices, Proc. 10th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2011), pp. 37 - 46 (2011.10)

[7] J. Ventura, and T. Höllerer: Wide-area scene mapping for mobile visual tracking, Proc. 11th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2012), pp. 3 - 12 (2012.11)

[8] M. Bauer, B. Bruegge, G. Klinker, A. MacWilliams, T. Reicher, S. Riß, C. Sandor, and M. Wagner: Design of a component-based augmented reality framework, Proc. Int'l Symp. on Augmented Reality 2001 (ISAR 2001), pp. 45 - 54 (2001.10)

[9] D. Schmalstieg, and D. Wagner: Experiences with handheld augmented reality, Proc. 6th Int'l Symp. on Mixed and Augmented Reality (ISMAR 2007), pp.3 - 15 (2007.11)

[10] W. Piekarski, and B. H. Thomas: Tinmith-evo5 – An architecture for supporting mobile augmented reality environments, Proc. Int'l Symp. on Augmented Reality 2001 (ISAR 2001), pp. 177 - 178 (2001.10)

[11] W. Piekarski: 3D modeling with the tinmith mobile outdoor augmented reality system, IEEE Computer Graphics and Applications, Vol. 26, No.1, pp. 14 - 17 (2006.1)

[12] J. Gausemeier, J. Fruend, C. Matysczok, B. Bruederlin, and D. Beier: Development of a real time mage based object recognition method for mobile AR-devices, Proc. Int'l Conf. on Computer Graphics, Virtual Reality, Visualisation and interaction in Africa 2003 (AFRIGRAPH 2003), pp. 133 - 139 (2003.2)

[13] B. MacIntyre, A. Hill, H. Rouzati, M. Gandy, and B. Davidson: The Argon AR web browser and standards-based AR application environment, Proc. Int'l Symp. on 10th Mixed and Augmented Reality (ISMAR 2011), pp. 65 - 74 (2011.10)

[14] Unity: http://unity3d.com/

[15] D. Wagner, and D. Schmalstieg: ARToolKitPlus for pose tracking on mobile devices, Proc. 12th Computer Vision Winter Workshop (CVWW07), pp. 139 - 146 (2007.1)

[16] 山下智紀，荒川祥太朗，柴田史久，木村朝子，田村秀行: モバイル MR システム構築のための機能分散型フレームワーク-システムアーキテクチャとコンテンツ制御機構-, 第 14 回日本バーチャルリアリティ学会大会論文集，3A2-3 (2009.9)

[17] 中満匠，柴田史久，木村朝子，田村秀行: モバイル MR システム構築のための機能分散型フレームワー

ク(6)－物理演算エンジンによるコンテンツ制御機構の拡張－，第 16 回日本バーチャルリアリティ学会大会論文集，14D-5, pp. 332 - 335 (2011.9)
[18] 前田貴裕，柴田史久，木村朝子，田村秀行: モバイル MR システム構築のための機能分散型フレームワーク(5)－サウンド制御機構の設計と実装－，第 16 回日本バーチャルリアリティ学会大会論文集，14D-4, pp. 328 - 331 (2011.9)
[19] 川端大輔，木村朝子，柴田史久: モバイル MR システム構築のための機能分散型フレームワーク(8)－アニメーションの表現力向上－，第 18 回日本バーチャルリアリティ学会大会論文集，32D-4, pp. 525 - 528 (2013.9)
[20] 小泉全弘，柴田史久，木村朝子，田村秀行: モバイル MR システム構築のための機能分散型フレームワーク(7)－スマートフォンのためのインタラクション機構の拡張－，第 16 回日本バーチャルリアリティ学会大会論文集，14D-6, pp. 336 - 339 (2011.9)
[21] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana, Virtual object manipulation on a table-top AR environment, Proc. Int'l Symp. on Augmented Reality (ISAR2000), pp.111 - 119 (2000.10)

（2013 年 12 月 10 日受付）

[著者紹介]

柴田 史久（正会員）

1996年大阪大学大学院基礎工学研究科博士前期課程修了．1999年同研究科博士後期課程修了．大阪大学産業科学研究所助手を経て，2003年4月より立命館大学理工学部助教授．同大学情報理工学部准教授を経て，現在，同教授．博士（工学）．モバイルコンピューティング，複合現実感等の研究に従事．本学会複合現実感研究委員会幹事．IEEE，電子情報通信学会，情報処理学会，ヒューマンインタフェース学会等の会員．本学会学術奨励賞・論文賞，ヒューマンインタフェース学会論文賞を受賞．

松田 祐樹

2012 年立命館大学情報理工学部情報コミュニケーション学科卒．2014 年同大学院情報理工学研究科博士前期課程修了．同年，任天堂（株）入社．2011 年より 2014 年 3 月まで，モバイル端末を用いた複合現実感に関する研究に従事．

レ ヴァン ギア

2012 年立命館大学情報理工学部情報コミュニケーション学科卒．現在，同大学院博士前期課程在学中．モバイル端末を利用した複合現実感に関する研究に従事．

川端 大輔（学生会員）

2013 年立命館大学情報理工学部情報コミュニケーション学科卒．現在，同大学院博士前期課程在学中．モバイル端末を利用した複合現実感に関する研究に従事．

山﨑 賢人（学生会員）

2013 年立命館大学情報理工学部メディア情報学科卒．現在，同大学院博士前期課程在学中．複合現実感を用いた作業支援システムの研究に従事．

木村 朝子（正会員）

1996年大阪大学基礎工学部卒．1998年同大学院基礎工学研究科博士前期課程修了．同大学助手，立命館大学理工学部助教授，科学技術振興機構さきがけ研究員等を経て，2009年4月より立命館大学情報理工学部メディア情報学科准教授．現在，同教授．博士（工学）．実世界指向インタフェース，複合現実感，ハプテックインタフェースの研究に従事．電子情報通信学会，情報処理学会，ヒューマンインタフェース学会，ACM，IEEE各会員．本学会学術奨励賞・論文賞，ヒューマンインタフェース学会論文賞，情報処理学会山下記念研究賞等受賞．