社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
PRMU-2005-141, HIP2005-111(2006-01)

多様な携帯・可搬型機器に対応可能なモバイル複合現実感システム(4)
－重量クライアントの実現と機能検証－

古野 光紀　柴田 史久　木村 朝子　田村 秀行

立命館大学大学院理工学研究科 〒525-8577 滋賀県草津市野路東 1-1-1
E-mail: furuno@mclab.ics.ritsumei.ac.jp

あらまし 我々はこれまで多様な携帯・可搬型機器に対応可能な複合現実感システム開発のための共通フレームワークの設計・実装を進めてきた．本フレームワークでは，クライアント・サーバ方式を採用し，クライアントを端末の処理能力に応じて，それぞれ「軽量クライアント」,「中量クライアント」,「重量クライアント」と呼ぶ 3 種類に分類している．これまでは軽量クライアント及び中量クライアントを，携帯電話，PDA，ウエアラブル PC 及びノート PC を用いて実装してきたが，本稿では，当該フレームワークにおいて高性能モバイル機器を対象とした実装形態である「重量クライアント」を実現したので，これの設計及び実装について報告する．実験による検証の結果，約 30fps での複合現実表示が実現できることが確認された．

キーワード 複合現実感，モバイル端末，共通フレームワーク，重量クライアント，ウエアラブル PC

A Variety of Mobile Mixed Reality Systems with Common Architectural Framework (4)
－Implementation and Performance Test of Heavy-load Client－

Koki FURUNO　Fumihisa SHIBATA　Asako KIMURA　and　Hideyuki TAMURA

Graduate School of Science and Engineering, Ritsumeikan University

1-1-1 Nojihigashi, Kusatsu, 525-8577 Shiga, Japan
E-mail: furuno@mclab.ics.ritsumei.ac.jp

Abstract We are now aiming at building of a general framework of mixed reality (MR) systems in which MR functions can be installed onto various kinds of mobile computers such as cellular phones, personal digital assistants (PDAs) and wearable computers. Our framework employs a server-client model, in which clients are classified into three types. In this paper, we describe a design and implementation of Heavy-load Client. As an experimental result, we confirmed that the system worked in 30 fps.

Keyword Mixed Reality, Mobile Device, General Framework, Heavy-load Client, Wearable Computer

1. はじめに

現実の光景に電子的な付加情報を重畳描画する複合現実感（Mixed Reality; MR）[1-3]は，屋内据置型は今や実用化の段階に達しており，これを屋外において利用する携帯・可搬型の MR システムに関する研究が活発化してきている[4-6]．いつでも，どこでも，手軽に利用したいという要求は，携帯電話の爆発的な普及からも明らかであり，今後は様々な分野において，屋外や広い範囲で手軽に利用可能なモバイル MR システムへの期待がより一層高まってくると想定される．

モバイル MR システムの実現には，ノート PC，ウエアラブル PC，携帯情報端末（PDA），携帯電話などが利用されるが，これらの携帯・可搬型情報機器は，日々性能や機能が高度化し，その種類も多岐にわたる．加えて，モバイル MR システムの使途は据置型システムでは実現不可能な範囲を含め，広範な分野に及ぶことが考えられる．故に，モバイル MR システムを実現する際に，使用機器・目的ごとに個別に設計・実装を行うことは時間的，金銭的に多大なコストを伴うと推測できる．すなわち，携帯・可搬型情報機器のように急速な発展が見込まれる分野における開発では，その機能・性能の違いを吸収可能な柔軟性のある枠組を導入することで開発の効率化が期待できると考える．

そこで我々は，多様な携帯・可搬型機器に対応可能なモバイル複合現実感システムの共通フレームワークの構築を目指している[7]．共通フレームワークのコン

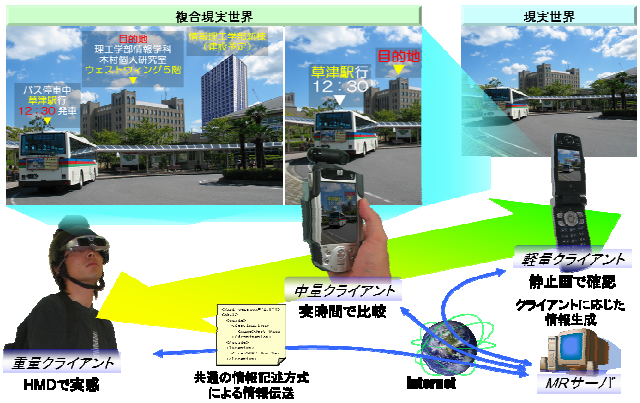


図 1 共通フレームワークのコンセプト

セプトを図 1 に示す．

我々が構築を進めている共通フレームワークでは，サーバ・クライアント形式でモバイル複合現実感システムを構成する．クライアントは自身が行う処理の負荷に応じて軽量クライアント，中量クライアント，重量クライアントに分類される．これまで我々は，複数種類のモバイル端末を用いて軽量および中量クライアントを実装し，機能検証を進めてきた．本稿では，本フレームワークにおける重量クライアント（Heavy-load Client）の実現と，機能検証を行なった結果について報告する．

2. 重量クライアントの設計

2.1. システム・アーキテクチャ

我々が構築を進めている共通フレームワークの設計方針は以下の 2 点である．

- 特定の端末やアプリケーションに依存しないこと
- 複数ユーザで MR コンテンツを共有できること

上記 2 点の設計方針を踏まえて，MR の実現に必要な機能をサーバ・クライアントに振り分けた．そのシステム・アーキテクチャを図 2 に示す．本節では，各クライアントの概要を述べる．

軽量クライアント（Light-load Client）は，現実の光景を撮像する機能と付加情報が重畳された MR 画像を表示する機能のみを有する．その他の機能はサーバに処理を依頼することで MR 表示を実現するため，1 枚の MR 画像を表示する度にサーバと画像のやりとりを

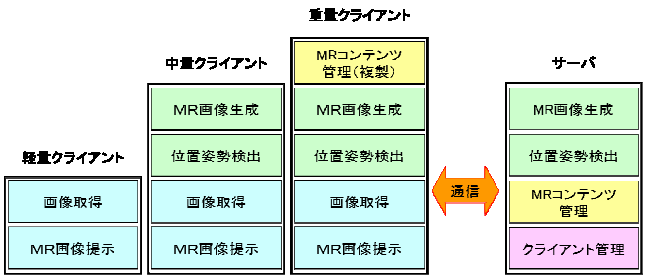


図 2 システム・アーキテクチャ

行う必要があり，連続的な MR 表示は不得手である．しかし，搭載される機能が少ないため実装は比較的容易であり，モバイル端末の中でも特に計算資源の制約が厳しい携帯電話での実現も可能であった．

中量クライアント（Middle-load Client）は，軽量クライアントに位置姿勢検出機能，MR 画像生成機能が付加された実装形態である．中量クライアントは，自身で検出した位置姿勢をサーバに送信し，現在提示すべき自身の周辺の MR コンテンツを受け取る．それをもとに MR 画像を生成することで，連続的に MR 表示を行うことが可能となる．軽量クライアントに比べて，通信によるオーバヘッドが軽減され，リアルタイムな MR 表示が可能となる．しかし，これまでの実験で，性能の低い端末を用いた場合に，提示する MR コンテンツがある程度複雑になると軽量クライアントとして動作させた方がスループットを向上させられるという結果が得られている[8]．

重量クライアント（Heavy-load Client）は，比較的高性能なモバイル端末を対象としており，MR の実現に必要な機能の全てを有する自己完結型のクライアントである．ユーザが HMD を装着し，高品質の MR をリアルタイムに体験する，という利用形態を想定している．重量クライアントは，サーバが保持する MR コンテンツ DB（Database）の完全な複製を取得し，自身で管理する．MR コンテンツ DB には，アプリケーションで使用する全ての MR コンテンツが格納されており，重量クライアントはその中から必要な MR コンテンツを自ら選択し，表示することができる．サーバの MR コンテンツ DB の複製を保持することによって，滑らかな MR 表示の妨げとなるサーバとの通信を大幅に削減でき，リアルタイムに MR を体験することが出来ると考える．中量クライアントにおいても，サーバから受け取った MR コンテンツを保持しておくことでリアルタイムな MR 表示が可能であるが，中量クライアントが取得できるのはクライアントの周辺に存在する MR コンテンツのみであり，表示する MR コンテンツの選択はサーバに委ねられる．

我々は，共通フレームワークの設計方針として複数ユーザが MR コンテンツを共有可能であることを挙げている．MR コンテンツを共有するため，重量クライアントの設計に関しては，自身とサーバの MR コンテンツ DB の同期方法の検討が必須課題となる．これについては 2.3 節で詳細に述べることとする．

2.2. モジュール構成と処理の流れ

本節では，共通フレームワークにおける重量クライアントのモジュール構成と，実際の処理手順について述べる．まず，重量クライアントとサーバのモジュール構成を示し，それを踏まえた上で MR 表示における

一連の処理の流れを説明する．

図 3 に重量クライアント，**図 4** にサーバのモジュール構成図を示す．各モジュールの機能は以下のとおりである．

- Data Flow Controller
  各モジュール間のデータの流れを制御する．
- Image Capturer
  カメラを用いて現実の光景の撮影する．
- Position Detector
  カメラ画像や各種センサを用いて，クライアントの位置姿勢を検出する．
- MR Contents DB Manager
  MR Contents DB に格納されている情報の取得及び更新作業を行う．
- MR Contents Manager
  MR Contents DB Manager より MR コンテンツを取得する．また，アニメーションを行う仮想オブジェクトの状態（位置姿勢）を決定する．オブジェクトの状態決定には，キーフレームでの状態をもとにした内挿法を用いる．
- MR Image Renderer
  Position Detector により求められた位置姿勢情報と，MR Contents Manager から取得する MR コンテンツをもとに MR 画像を生成する．
- Request Manager
  サーバ・クライアント間の情報伝達に用いるコンテンツ記述言語 SKiT-XML[9]を使用してサーバと通信を行う．
- MR Contents DB
  サーバが持つ MR Contents DB の複製．
- MR Contents DB Synchronizer
  サーバと自身の MR Contents DB の同期を図る．
- UI(User Interaction) Manager
  ユーザの仮想世界に対するインタラクションを検知し，Data Flow Controller に通知する．
- MR Image Presenter
  MR Image Renderer が生成した MR 画像を表示する．
- Application Module
  アプリケーションに特有の処理を行う．具体的には，MR Contents DB に格納されている MR コンテンツから提示すべきものを選択する．そして，必要に応じてクライアントの位置姿勢やインタラクションをもとに MR コンテンツに変更を加える．

上記以外で，サーバ側にのみ存在するモジュールの機能は以下のとおりである．

- Client Manager
  クライアントの状態を管理する．また，クライアントの Data Flow Controller と同様，モジュール間のデータの流れを制御する．

重量クライアントで MR 表示を行う際は Data Flow Controller を中心として以下のように処理が進められる．処理の流れを**図 5** に示す．尚，以下に示す各項目の数字は図 5 中の数字に対応する．

① Image Capturer を用いてカメラ画像を取得する．
② ①で得た画像や，センサなどの出力をもとに Position Detector によって位置姿勢を検出する．
③ UI Manager からユーザが行ったインタラクションを取得する．
④ ①，②，③で得られた情報を Application Module に渡し，提示すべき MR コンテンツの ID リストを受け取る．
⑤ ④で得られた ID リストを MR Image Renderer に渡し，MR 画像を生成する．
⑥ MR Contents Manager は提示する MR コンテンツの ID リストを受け取り，対応する MR コンテンツの詳細な情報を MR Contents DB Manager から取得する．提示する MR コンテンツがアニメーションを行う場合は，該当する仮想オブジェクトの MR 空間での位置姿勢を決定する．
⑦ MR Contents DB Manager は要求された ID に対

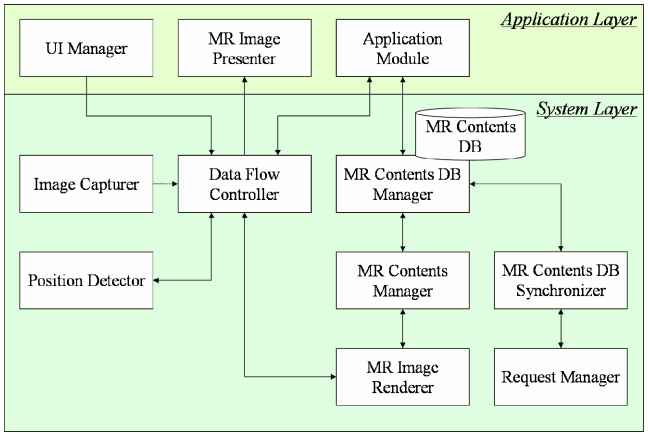


図 3 モジュール構成図（重量クライアント）

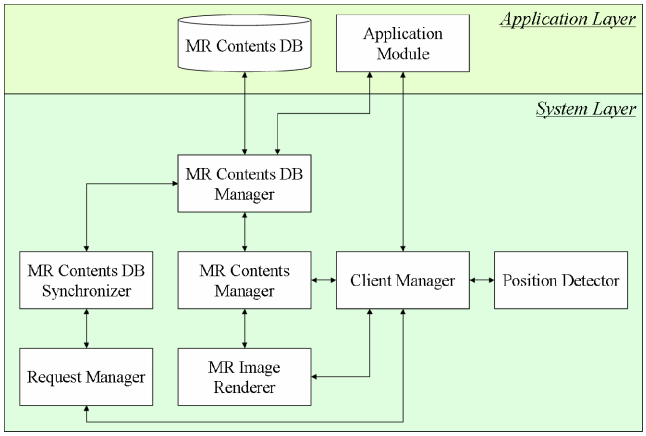


図 4 モジュール構成図（サーバ）

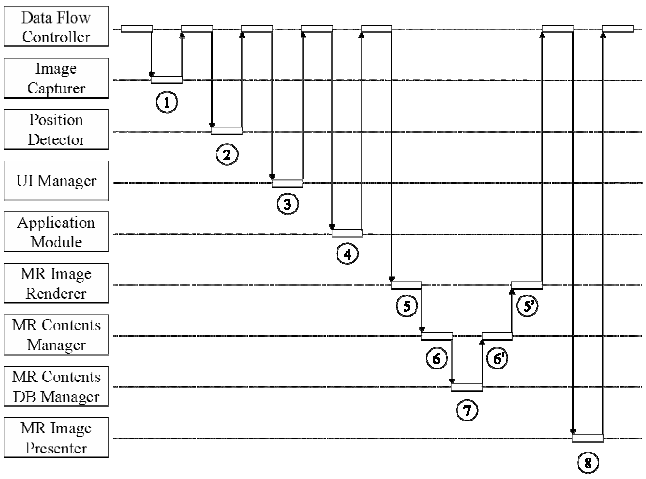


図 5 MR 表示処理の流れ

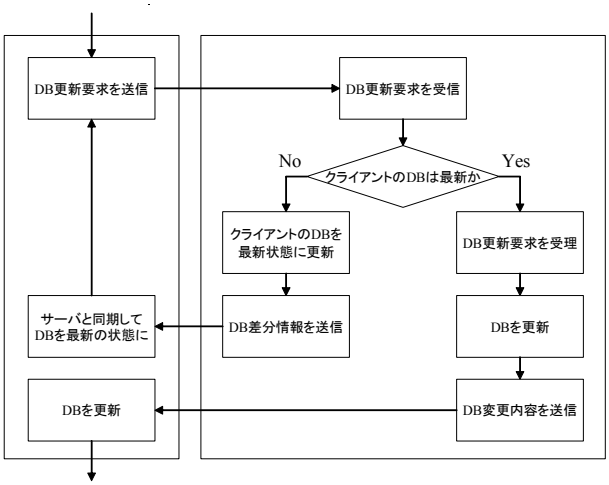


図 6 クライアント側からの DB 更新の流れ

応する MR コンテンツを DB から取得する.

⑧ MR Image Presenter を用い，⑤で生成された MR 画像をユーザに提示する.

2.3. MR コンテンツ DB 同期方法の検討

重量クライアントはシステム起動時にサーバの MR コンテンツ DB の複製を取得し，自身で管理する．しかし，MR コンテンツ DB は Application Module によって変更される場合があり，共通フレームワークの設計方針である複数ユーザによる MR コンテンツの共有を満足させるためには，重量クライアントとサーバの間で MR コンテンツ DB の同期を図る必要がある.

MR コンテンツ DB を同期させるため，MR Contents DB Synchronizer を重量クライアントとサーバに配置する.重量クライアントの立場から MR コンテンツ DB の同期を考えた場合，MR Contents DB Synchronizer には以下の 2 通りの状況に対応する処理が必要となる.

a) サーバの MR コンテンツ DB に変更があった場合，それを自身の DB に反映させる

b) 自身の DB を変更する場合，同様の変更をサーバの DB に対して行う

a)に関しては，MR Contents DB Synchronizer が定期的にサーバに対して MR コンテンツ DB の変更の有無を問い合わせることで解決する．サーバ側で何らかの変更が行われていた場合，更新前からの差分情報が返信され，それをもとに自身の DB を更新する.

b)の場合は，自身の MR コンテンツ DB を更新する前にサーバに DB 更新要求を送信する.DB 更新要求には変更内容が含まれる．DB 更新要求を送信した重量クライアントの MR コンテンツ DB がサーバの DB と同期されていた場合に限り DB 更新が認められ，サーバから変更内容が返信される．DB の同期が取られていなかった場合は，現在のサーバ側の DB と同一の状態に更新するための DB 差分情報が返信され，その変更を行った上で再度 DB 更新要求を送信する．b)における処理の流れを**図 6** に示す.

これらの処理により，サーバの DB は最新の状態が保たれ，各クライアントがサーバと同期を取ることで全体として MR コンテンツの共有が可能となる.

3. 実験と考察

3.1. 重量クライアントの実装

2 章で述べた設計に沿って重量クライアントの実装を行った．**表 1** に使用したモバイル端末とその性能を示す．全ての端末において，表示デバイスには島津製作所 DataGlass2/A を採用し，カメラデバイスとして M60 と TypeU は USB PC カメラ CMS-V13 を，MA-V は IEEE1394 カメラ Orange Micro iBOT を使用する．ユーザの位置姿勢検出には ARToolKit[9]を用いる.

実装に使用する全ての端末は Microsoft Windows XP Professional を OS とする PC である．開発環境には Microsoft Visual C++ .NET を用い，DirectX 9.0c によって 3D グラフィックスの描画を行う.サーバは Pentium4 3GHz, 1GB RAM で OS は Redhat 9 という構成である.

3.2. 性能評価

重量クライアントとして実装を行った全ての端末

表 1 使用端末のハードウェア構成

使用端末	CPU	RAM	GPU
Dell Precision M60	Intel Pentium M 2.1[GHz]	2[GB]	nVIDIA Quadro FX Go 1000
Sony VGN-U71P(VAIO TypeU)	Intel Pentium M 1.1[GHz]	512[MB]	Intel 855GM チップセット
Xybernaut MA-V	Intel Celeron 500[MHz]	256[MB]	ATI RAGE Mobility-M

表 2 各処理における実行時間 [ms]

端末	画像取得	位置姿勢検出	MR 画像生成
M60	10.251	14.507	14.373
TypeU	21.723	29.252	11.142
MA-V	49.993	74.946	56.718

で，簡単な仮想オブジェクト（球体）を描画し，処理時間を計測した．画像取得，位置姿勢検出，MR 画像生成それぞれに要した時間を**表 2** に示す．なお，この実験ではサーバと接続せず，クライアント単体で仮想物体の表示を行った．結果は 200 フレームを表示した平均値である．描画のフレームレートは M60 で 25.7fps, TypeU で 16.3fps，MA-V では 5.8fps であった．本実験では，カメラデバイスで撮影する画像サイズを VGA（640x480）に設定して時間計測を行ったが，画像サイズを QVGA（320x240）に変更した場合のフレームレートはそれぞれ 31.0, 31.2, 13.3fps に向上した．

3.3. 仮想オブジェクトのアニメーション

仮想オブジェクトのアニメーションは，今回初めて導入した機能である．オブジェクトの位置姿勢は時間によって区切られたキーフレームの補間によって決定する．8 秒周期でマーカを中心として半径 200mm の円運動を行う仮想オブジェクト（鳥）を表示した．表示した鳥は円運動を行うと共に翼を羽ばたかせている．**図 7** に表示結果を示し，**表 3** に処理時間と全体に占める割合を示す．本来は光学シースルーHMD を用いて情報提示行うため背景は描画されないが，実行結果には背景を付加した図を示す．表 3 に示す結果から分かるように，アニメーションを導入しても表示フレームレートはほとんど変化せず，滑らかに仮想オブジェクトを表示させることが出来た．仮想オブジェクトは，想定した通り円運動を行い，キーフレームの補間処理が適切に行われたことが確認できた．また，時刻をもとに補間を行っているので，端末の時計を合わせておけば仮想オブジェクトの動きの同期も可能であった．

3.4. MR コンテンツ DB の同期

MR Contents DB Synchronizer による MR コンテンツ DB の同期に関して機能検証を行った．実験の手順として，サーバ側および重量クライアント側で MR コンテンツ DB に変更を加え，それが他方の DB に反映されることを目視で確認する事とした．ここでは Precision M60 を用いて実験を行った．

まず，サーバ側の MR コンテンツ DB に対して仮想オブジェクトの形状変更を行った．具体的には，表示される仮想オブジェクトを立方体から球体に変更した．その結果，重量クライアントで提示されていた立方体が球に変更され，DB の同期処理が正しく行われたことがわかった．変更前後の表示結果を**図 8** の(a)，(b)に示す．次に，重量クライアント側で MR コンテンツ DB に変更を加えた．上記の逆で，球体を立方体に変更した．サーバ側の DB に変更が反映されたことを確認するため，軽量クライアントを用いて MR 表示を行った．軽量クライアントは画像取得と MR 画像提示

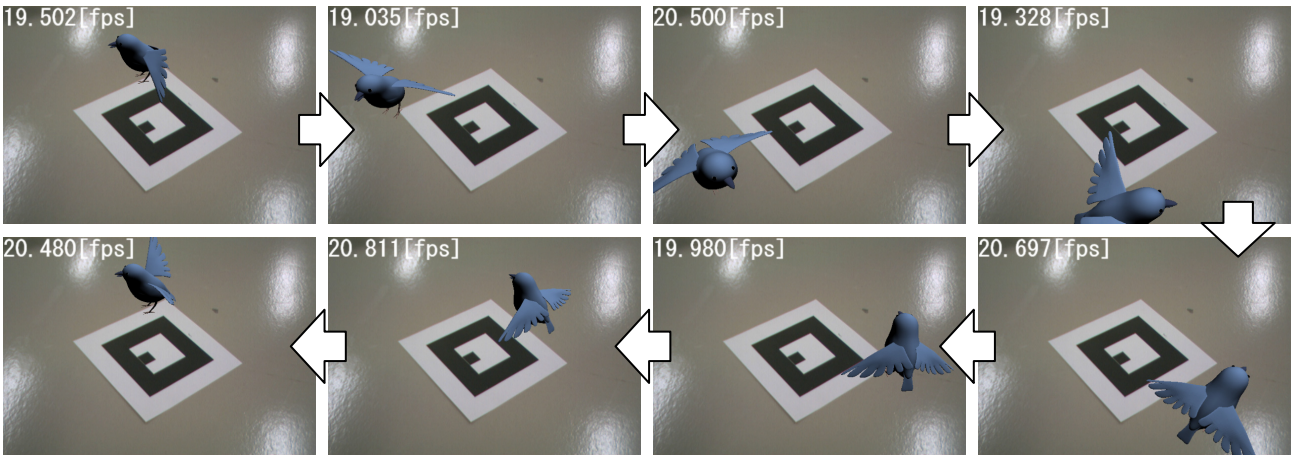
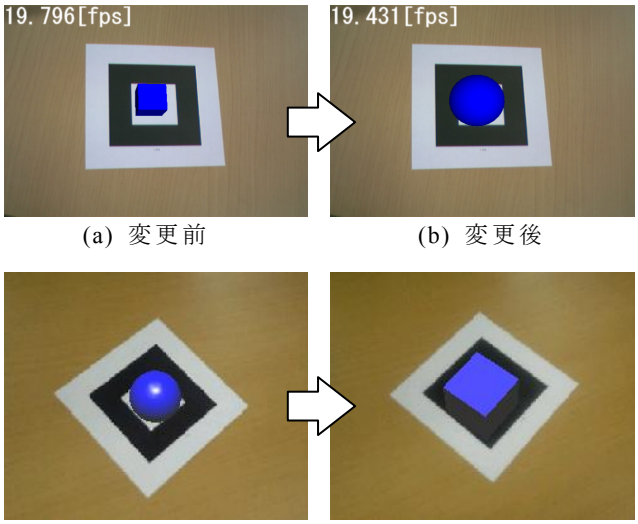


図 7 アニメーション表示結果

表 3 アニメーションを行う場合の処理時間

端末	画像取得(VGA)		位置姿勢検出		アニメーション処理		MR 画像生成		fps
	時間[ms]	割合	時間[ms]	割合	時間[ms]	割合	時間[ms]	割合	
M60	10.1	0.30	12.6	0.36	0.7	0.02	11.7	0.34	29.3
TypeU	22.0	0.34	26.4	0.41	0.8	0.01	17.1	0.27	15.6
MA-V	47.8	0.24	69.8	0.36	0.9	0.004	81.6	0.41	5.1



(a) 変更前　(b) 変更後

(c) 変更前　(d) 変更後

図 8 MR コンテンツ DB 変更の反映

以外の機能をサーバに委ねており，サーバの DB の変更を確認するのに最適なクライアントであると考えられる．軽量クライアントによる変更前後の表示結果を図 8 の(c), (d)に示す．これらの結果より，MR Contents DB Synchronizer による DB の同期処理が設計通りに機能し，それぞれの MR コンテンツ DB は同一の状態に保たれることが確認された．また，MR Contents DB Synchronizer の同期処理の間隔を 2 秒，4 秒，8 秒として，重量クライアント（M60）の DB を更新した際，サーバを介して他の重量クライアント（TypeU）の DB に反映されるまでの時間は，それぞれ 6.6 秒，7.6 秒，11.2 秒であった．

3.5. 考察

今回の実装と機能検証実験によって，共通フレームワークにもとづく重量クライアントが実現可能であることが確認できた．処理速度の面から見ても，これまでに実装を行ってきた中量クライアントと同等以上の性能が発揮された．アニメーション処理に関しては処理の負荷が極めて小さく，共通フレームワークにとって有効な機能拡張であったと思われる．また，サーバ・重量クライアント間での MR コンテンツ DB の同期に関しても問題なく実現でき，結果としてシステム全体で MR コンテンツの共有が可能であると確認できた．

問題点として，位置姿勢検出に用いる画像の大きさで表示フレームレートが極端に変化してしまっていることが挙げられる．一般的に，画像ベースの位置姿勢検出手法は解像度が検出精度に大きな影響を与える事となり，処理速度とのトレードオフの見極めを行う必要がある．今後は，屋外利用も視野に入れ，各種センサ類を併用した位置姿勢検出手法を導入していく予定である．

現在，重量クライアントを利用したガイドアプリケーションの作成を進めており，当アプリケーションを用いて更なる機能検証を進める予定である．想定しているアプリケーションは，エージェントがユーザと共に移動しながら案内を行うもので，図 7 に示した鳥はそのエージェントの候補の 1 つである．

4. むすび

本稿では，モバイル複合現実感システムのための共通フレームワークにおける重量クライアントの設計と機能検証実験について述べた．重量クライアントが実現できたことで，共通フレームワークにおける全ての実装形態が実現可能であることが確認された．

今後は，共通フレームワークの更なる発展を目指すと共に，各クライアントの機能検証実験を重ねていく予定である．また，屋外利用可能な位置姿勢検出手法を提案し，実用的なアプリケーションと組み合わせて屋外での利用法も提案していきたいと考える．

謝　辞

本研究の設計・実装，及び実験にご協力頂いた橋本崇氏，吉田友祐氏，松田征洋氏をはじめとする研究グループの各位に感謝します．また，本研究は科学研究費補助金（基盤研究(B)No.17300039）及びハイテク・リサーチ・センター整備事業の一部の補助を受けて行われました．

文　献

[1] 「複合現実感」特集号，日本バーチャルリアリティ学会論文誌（TVRSJ），Vol.4, No.4, 1999.

[2] 「複合現実感 2」特集号，同上，Vol.7, No.2, 2002.

[3] 「複合現実感 3」特集号，同上，Vol.10, No.3, 2005.

[4] S.Feiner et al.: “A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment,” Proc. of Int. Symp. on Wearable Computers(ISWC’97), pp.74-81, 1997.

[5] 興梠他: “ウェアラブルカメラと慣性センサ群のデータ統合に基づくパーソナルポジショニング”, 信学技報, PRMU2002-180, pp.67-72, 2003.

[6] 神原他: “RTK-GPS と慣性航法装置を併用したハイブリッドセンサによる屋外型拡張現実感システム”, 画像の認識・理解シンポジウム（MIRU2004）講演論文集, pp.933-938, 2005.

[7] 柴田他: “多様な可搬型機器に対応可能な複合現実感システムの共通フレームワークの設計と実装”, 日本バーチャルリアリティ学会論文誌, Vol.10, No.3, pp.323-331, 2005.

[8] 平岡他: “多様な携帯・可搬型機器に対応可能なモバイル複合現実感システム(3) - 高性能端末での機能再検証 -“, 日本バーチャルリアリティ学会第 10 回大会論文集, pp.101-104, 2005.

[9] 橋本他: “モバイル複合現実感システムにおけるコンテンツ記述言語の設計と実装”, 信学技報, PRMU2005（本研究会）

[10] H.Kato et al.: “Virtual Object Manipulation on a Table-top AR Environment,” Proc. of Int. Symp. on Augmented Reality (ISAR 2000), pp.111-119, 2000.