日本バーチャルリアリティ学会第 10 回大会論文集(2005 年 9 月)

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―

A Variety of Mobile Mixed Reality Systems with Common Architectural Framework (3):
Experiment on High-Performance Computers

平岡貴志，橋本崇，柴田史久，木村朝子，田村秀行
Takashi HIRAOKA, Takashi HASHIMOTO, Fumihisa SHIBATA, Asako KIMURA, and Hideyuki TAMURA

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)

Abstract: We aim at constructing a common architectural framework for a variety of mobile mixed reality systems. The characteristic of our architecture is to adapt the server-client architecture and the distribution of necessary functions for mixed reality presentation between the server and the client according to a processing capacity of mobile computers. We have tested the effectiveness of our architecture by some kinds of mobile computers, such as cellular phones and PDAs. In this paper, we report the result of experiment on high-performance computers. As a result, we confirmed that mixed reality information was presented in real time with high-performance computers implemented as a client based on our architecture.
Key Words: *Mixed Reality, Mobile System, Common Architectural Framework, High-performance Computers*

1．はじめに

現実世界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）に関する研究が活発化している[1][2]．屋内利用向けの据置型システムが実用化される一方で，屋外利用を想定した可搬型システムも盛んに研究されている[3]. MR をいつでもどこでも手軽に利用できる可搬型システムの利点を考えると，今後も疑いなく可搬型システム実用化の流れは加速していくはずである．

こうした背景を踏まえ，我々は様々な携帯・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構築を目指している[4][5]．多様な可搬型機器に対応させるため，まず処理能力や記憶容量に制限が厳しい携帯電話・PDA を複数種類選択し，これまで機能検証を進めてきた．結果として，本フレームワークに基づく実装で複合現実型情報提示が可能であることが分かった[6][7]．本稿では，本フレームワークの更なる有効性を確認すべく高性能な端末での機能再検証を行い，性能評価した結果について報告する．

2．共通フレームワークの構成

2.1 システム・アーキテクチャ

本フレームワークの設計は，多様な携帯・可搬型機器に対応可能であること，複数端末で MR コンテンツを共有可能であること，を前提に進めてきた．これらの条件を満たすため，本フレームワークを図 1 に示すシステム・アーキテクチャによって実現した．軽量クライアントは，低性能な端末を対象とし，画像を取得する機能と生成した MR 画像を提示する機能のみを有し，残る処理は全てサーバに委ねる．中量クライアントは，自身で位置姿勢を検出し，サーバから自己の周辺における少量の MR コンテンツを取得し，それを基にリアルタイムな MR 画像を生成し提示する．重量クライアントは，高性能な端末を対象とし，MR 画像を提示する上で必要な機能を全て有した自己完結型クライアントである．サーバと重量クライアントが管理する MR コンテンツは，何らかの変化が生じた際その差分情報を互いに共有し同期を図る．

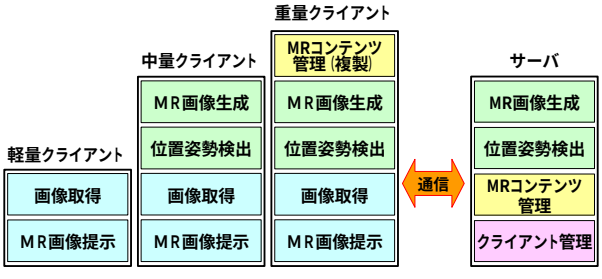

図 1　システム･アーキテクチャ

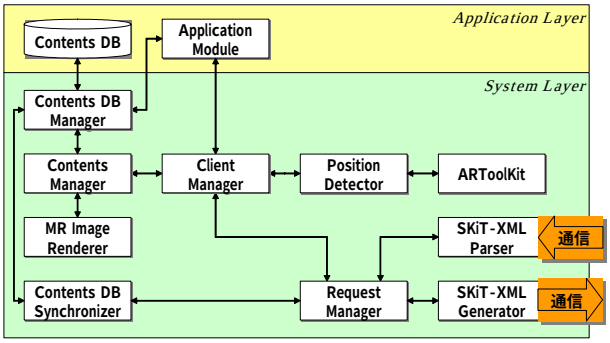

図 2 サーバのモジュール構成

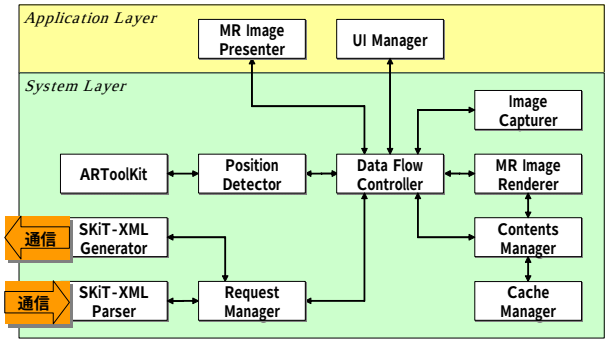

図 3 中量クライアントのモジュール構成

2.2 モジュール構成と処理の流れ

ここでは，サーバ及び中量クライアントのモジュール構成と処理の流れを説明する．本研究で実装したサーバのモジュール構成を図 2，中量クライアントのモジュール構成を図 3 に示す．図 2 及び図 3 に示すモジュール群は，大きく *Application Layer* と *System Layer* に分けられる．*System Layer* では，アプリケーションに依存しない MR を実現する上で基本となるモジュールを我々が提供し，*Application Layer* ではアプリケーション開発者が *System Layer* のモジュールと連携する形で，個々のアプリケーションを実現するためのモジュールを作成する．

中量クライアントが MR 画像を提示するまでの一連の処理の流れを以下に示す（図 4 参照）．

① Image Capturer で現実の光景の撮影画像を取得する．
② Position Detector によりクライアントの位置姿勢を検出する．
③ Request Manager が検出した位置姿勢情報を SKiT-XML Generator へ渡し，その情報を含んだ SKiT-XML Request を生成しそれをサーバへ送信する．
④ サーバでは，受信した SKiT-XML Request を Request Manager が SKiT-XML Parser を用いて解析し，位置姿勢情報を取り出す．
⑤ Client Manager は得られたクライアントの位置姿勢情報を基に Application Module で必要な MR コンテンツを決定し，Contents Manager を介して MR コンテンツを取得する．
⑥ Request Manager は，取得した MR コンテンツを SKiT-XML Generator へ渡し，それらの情報を含んだ SKiT-XML Response を生成し，中量クライアントへ送信する．
⑦ クライアント側の Request Manager は，受信した SKiT-XML Response を解析し，得られた MR コンテンツを MR Image Renderer に渡す．
⑧ MR Image Renderer は，別途 Position Detector より位置姿勢情報を受け取り，先ほどの MR コンテンツを使って MR 画像を生成する．
⑨ 生成した MR 画像を MR Image Presenter が提示する．

一度上記の流れに従って MR コンテンツを取得すれば，クライアントの位置が大きく変わらない限りにおいては，以降 MR コンテンツを取得する過程は必要なくなる．その場合，MR 情報を提示する処理の流れは，①→②→⑧→⑨となる．実装においては，このことを考慮した上で，MR 情報を提示する処理と，MR コンテンツを取得する処理を別スレッドに分けて，処理速度の向上を図っている．

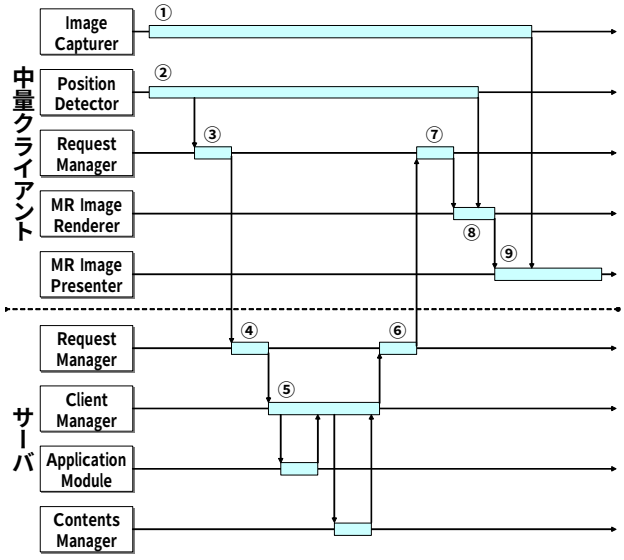

図 4 処理の流れ

3. 実験

3.1 床下配線サポートシステムの概要

提案アーキテクチャの有効性を確認するために，実験用アプリケーションとして『床下配線サポートシステム』を構築した．本システムは，予め位置情報が既知であるフリーアクセスフロア下の配線を，MR 機能を利用して重畳表示し，配線変更作業の煩雑さを軽減するシステムである．

本システムでは以下に示す機能を実現している．

- 任意の床下の配線状況の MR 表示
- 対象範囲全体の床下配線の全体図表示

本稿では，この『床下配線サポートシステム』を基に高性能端末の機能検証のための実験を行う．

3.2 実験機器と実験環境

前節の床下配線サポートシステムを複数の可搬型機器で実装した．用いた端末の種類と実装形態を表 1 に示す．

サーバは Redhat Linux 9 を OS とするワークステーション Dell Precision 450 に Java Servlet 2.4 により実装した．主な仕様は，CPU が Intel Xeon 2.4GHz，メインメモリが 1GB，グラフィックスカードが nVIDIA Quadro FX 500 である．Java Servlet の動作環境として Apache 2.0.52 と Tomcat 5.5.4

表 1 仕様端末と実装形態

端末分類	使用端末	実装形態
ワークステーション	Dell Precision 450	サーバ
携帯電話	NTT DoCoMo SH901iC	軽量クライアント
PDA	SHARP Zaurus SL-6000W	軽量クライアント
PDA	SHARP Zaurus SL-6000W	中量クライアント
PDA	HP iPAQ h5550	中量クライアント
ウエアラブルPC	Xybernaut MA-V	中量クライアント
ノートPC	Dell Precision M60	中量クライアント

を利用している. MR 画像のレンダリングには Java 3D 1.3.1 を用いた.

各クライアントのハードウェア構成を表 2 に示す. 携帯電話を用いたクライアントは, NTT DoCoMo の i アプリ DoJa4.0 プロファイル上で実装した. PDA を用いたクライアントは SHARP Zaurus SL-6000W と HP iPAQ h5550 の 2 機種である. SL-6000W は Embedded Linux (OpenPDA 1.0) を OS とし, 開発言語は C++で, Cygwin 環境にクロスコンパイラをインストールし開発環境を構築した. 一方, h5550 は Windows Mobile 2003 software for Pocket PC を OS とし, 開発環境は Microsoft eMbedded Visual C++ 4.0 である. ウエアラブル PC とノート PC は, Windows XP Professional を OS とし, 開発環境は Visual C++ .NET 2003 である.

実験対象は, 図 5 に示すフリーアクセスフロアのタイル 4×6 枚の範囲である. タイル 1 枚は, 50cm 四方の大きさであり, タイル下 10cm には, 実線で示す電源ケーブルと点線で示す LAN ケーブルが通っている. また, サーバと中量クライアントの位置姿勢検出機構に ARToolKit を利用した[8]. これにより, 各タイルの中央に 12cm 四方の正方形マーカを貼付している.

3.3 実験結果と考察

初めに中量クライアントにおける各 MR 機能に要する時間について, 『床下配線サポートシステム』で実験をした. 各々のモジュールの平均実行時間を 10 回の試行から計算した. 結果を表 3 に示す. 想定した通りの明確な端末性能差が得られた.

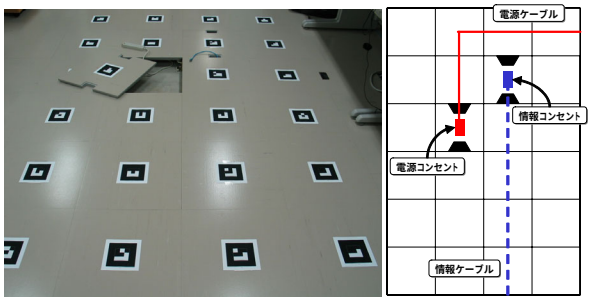

図 5 実験環境と見取り図

表 3 MR 機能に要する時間(単位:ms)

端末	画像取得	位置姿勢検出	MR情報生成	MR情報提示
SL-6000W	249	971	206	62
h5550	233	175	236	18
MA-V	24	35	76	4
Precision M60	30	9	16	2

次にレンダリング対象となる仮想物体との距離を変更してフレームレートを計測してみた. カメラから対象までの距離は, 座った状態 (30cm), 立った状態 (100cm) で真上からタイルを見下ろすように撮影する. 結果として, 全クライアントで座った状態での撮影で多少フレームレートが落ちたが, 極端な低下は見られなかった. 以前の実装では, MR 情報生成においてカメラと対象の仮想物体との距離が近いほど極端にレンダリングに時間がかかってしまうという問題があったが[6], 設計及び実装の見直しによりこれを改善した.

次にレンダリングする仮想物体が複雑になった場合の各端末の動作について調べた. 実験では, 簡単のため複雑な仮想物体を複数の三角形の組み合わせで近似する. 各三角形は合同で同一平面上に重ならないように規則的に配置し, その個数を変化させ各端末のフレームレートを計測した. 計測結果を表 4 に示す. 各端末とも三角形の個数が増えるにつれて, 徐々にフレームレートが低下しているが, それほど大きな変化はない.

そこで上記の実験よりもさらに負荷をかけた場合の各端末の動作を調べてみた. 今度は表示する仮想物体を底面の分割数 16 の円錐とし, 表示する円錐の個数を変化させ, 端末毎のフレームレートを計測した. 計測結果を表 5 に示す. 前述の三角形表示の実験では見られなかったが, この実験では SL-6000W の軽量・中量クライアントの動作結果から分かるように, 中量クライアントはレンダリングする仮想物体のポリゴン数が一定以上増えると, 負荷のかかる処理をサーバに任せる軽量クライアントとして振舞う方がパフォーマンス向上を図れることが分かる.

表 4 仮想物体数とフレームレートの関係

仮想物体数	1	5	10	20	50	100	200	500
SH901iC	0.069	0.074	0.075	0.076	0.077	0.075	0.071	0.065
SL-6000W(軽量)	0.283	0.212	0.210	0.195	0.300	0.259	0.238	0.161
SL-6000W(中量)	0.835	0.830	0.847	0.799	0.691	0.674	0.536	0.423
h5550	2.140	2.124	2.114	1.997	1.763	1.458	1.120	0.661
MA-V	14.15	14.12	13.33	11.34	11.11	10.87	10.30	8.815
Precision M60	20.96	19.96	19.63	20.00	19.00	18.88	17.79	17.63

表 2 各クライアントのハードウェア構成

使用端末	CPU	メモリ	表示解像度	グラフィックスカード	カメラデバイス	通信方式
SH901iC	――	――	240 x 240	――	内蔵カメラ	携帯電話網
SL-6000W	Intel Xscale PXA255 400 [MHz]	64 [MB]	480 x 640	――	SHARP CE-AG06	無線LAN IEEE802.11b
h5550	Intel Xscale PXA255 400 [MHz]	128 [MB]	240 x 320	――	Life View FlyCAM-CF1.3	
MA-V	Intel Celeron 500 [MHz]	256 [MB]	800 x 600	ATI RAGE Mobility-M	OrangeMicro iBot	無線LAN IEEE802.11g
Precision M60	Intel Pentium M 2.1 [GHz]	2 [GB]	800 x 600	nVIDIA Quadro FX Go 1000	小型カメラ(NTSC信号出力)	

表 5 仮想物体数とフレームレートの関係

仮想物体数	1	5	10	20	50	100	200	500
SH901iC	0.077	0.075	0.081	0.074	0.072	0.071	0.064	0.057
SL-6000W(軽量)	0.314	0.309	0.288	0.288	0.280	0.257	0.225	0.203
SL-6000W(中量)	0.664	0.641	0.773	0.676	0.508	0.404	0.222	0.101
h5550	2.139	1.985	1.774	1.498	1.045	0.694	0.414	0.207
MA-V	5.763	4.869	4.661	3.966	2.842	1.930	1.200	0.531
Precision M60	14.92	14.58	14.39	13.09	10.96	8.684	6.186	3.329

次にサーバ側のデータベースに含まれる MR コンテンツを更新した際の中量クライアントの動作について実験した．ここでは，中量クライアントが MR コンテンツ更新の有無をサーバに確認する間隔を 0 秒，2 秒，4 秒，8 秒に設定した．尚，更新間隔 0 秒とは，常にサーバに MR コンテンツ更新の問い合わせをする状態である．計測結果を表 6 に示す．MA-V，Precision M60 では更新間隔 0 秒において更新が瞬時に反映された．更新間隔 2 秒，4 秒，8 秒において，SL-6000W のみがデータベース更新の反映時間が更新間隔よりも長かった．またこのとき各端末が受信した SKiT-XML の容量は 7KB であった．

上記の場合，仮想物体はシンプルな形状の配線 2 本で，また全体のポリゴン数や受信する SKiT-XML の容量も少ない．そこでポリゴン数を増やし，そのときの更新反映時間を計測してみた．実験では前述の円錐を選択し，円錐の個数を 1 個，100 個，500 個に変化させ反映時間を計測した．結果を表 7，表 8，表 9 に示す．それぞれで受信した SKiT-XML の容量は 3KB，144KB，709KB であった．表 7 でも表 6 と同様に SL-6000W のみ更新間隔より更新の反映時間の方が長い．しかし，表 9 では高性能端末でも同様のことが起きている．これは仮想物体の増加に伴い，受信する SKiT-XML の容量も増え，それのダウンロードや解析に要する時間も増えてしまうからである．故に更新タイミングにもよるが更新反映時間は設定した更新間隔より長くなることがある．高性能な端末ほど，コンテンツ取得後の処理は高速で行われるため，更新の反映時間が更新間隔に設定した時間より長くなりにくいが，低性能な端末ではそれを考慮して更新間隔を設定する必要がある．

表 6 データベース更新の反映時間(単位：s)

更新間隔	0 [s]	2 [s]	4 [s]	8 [s]
SL-6000W	4.001	5.262	5.014	9.154
MA-V	―――	1.707	2.752	7.163
Precision M60	―――	1.931	2.795	4.283

表 7 仮想物体数 1 個での更新反映時間(単位：s)

更新間隔	0 [s]	2 [s]	4 [s]	8 [s]
SL-6000W	3.240	5.504	9.615	14.61
MA-V	―――	1.187	2.466	4.876
Precision M60	―――	1.804	2.384	5.016

表 8 仮想物体数 100 個での更新反映時間(単位：s)

更新間隔	0 [s]	2 [s]	4 [s]	8 [s]
SL-6000W	15.24	16.36	21.92	22.50
MA-V	2.691	3.143	4.948	8.328
Precision M60	1.088	2.287	2.930	4.916

表 9 仮想物体数 500 個での更新反映時間(単位：s)

更新間隔	0 [s]	2 [s]	4 [s]	8 [s]
SL-6000W	72.43	74.87	78.87	84.00
MA-V	11.89	10.81	10.94	15.23
Precision M60	3.828	4.854	5.036	7.347

4．むすび

これまで性能面で制限の厳しい携帯電話・PDA での提案フレームワークの機能検証は完了していた．本研究では，高性能な端末での機能再検証を試みた．想定していた通り高性能な端末で本フレームワークに基づく中量クライアントを実装することができた．また，高性能端末の性能評価の結果より，高いパフォーマンスが確認され，本フレームワークに基づく実装で，リアルタイム複合現実型情報提示が可能であることが実証された．

今後は，高い性能を生かしたアプリケーションを考案・実装し，本フレームワークの有用性を向上させていく予定である．

謝辞

本研究を進めるにあたり，モバイル複合現実感システム及び床下配線サポートシステムの設計・実装，端末の性能評価実験等に御協力頂いた関係者各位に深く感謝申し上げる．

参考文献

[1]「特集：複合現実感」日本 VR 学会論文誌，Vol.4，No.4，1999

[2]「特集：複合現実感 2」 同上，Vol.7，No.2，2002

[3]R.Azuma et al., Recent Advances in Augmented Reality, IEEE Computer Graphics and Applications, Vol.21, No.6, pp.34-47, 2001

[4]柴田他： 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(1)―基本アーキテクチャとコンテンツ記述方式―，日本 VR 学会第 9 回大会論文集，pp.281-284，2004

[5]橋本他： 多様なモバイル機器に対応可能な複合現実感システムの開発(1)―基本アーキテクチャ第 2 版―，信学総大講演論文集，pp.289，2005

[6]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(2)―携帯電話・PDA による試作例―，日本 VR 学会第 9 回大会論文集，pp.285-288，2004

[7]古野他： 多様なモバイル機器に対応可能な複合現実感システムの開発(4)―改良アーキテクチャでの機能再検証―，信学総大講演論文集，pp.292，2005

[8]H.Kato et al., Virtual object manipulation on a table-top AR environment, Proc. Of Int. Symp. on Augmented Reality(ISAR2000), pp.111-119, 2000