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－

A Distributed Framework for Mobile Mixed Reality System (6)
- Improvement of Content Control Mechanism Based on Physics Engine -

中満匠，柴田史久，木村朝子，田村秀行
Takumi Nakamitsu, 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 have developed a distributed framework for mobile mixed reality system. In this paper, we propose a novel mechanism for our framework to control the content in order to realize easy control of "physical" motion in classical dynamics. Our framework controls the motion of the virtual objects based on the original script language. However, it is difficult and messy to express "physical" motion using such a language because developers have to formulate the laws of physics. Therefore, we employ a physics engine that provides a simulation of physical systems, such as rigid body dynamics and collision detection to control the virtual objects. In addition, we refine the interpolation algorithm of position and orientation of the virtual objects to cope with cusps in "physical" motion.

Key Words: *Mixed Reality, Mobile System, Physics Engine*

1. はじめに

我々はこれまで，複数の端末が同一の複合現実（Mixed Reality; MR）空間を共有可能なモバイル MR システム構築のためのフレームワークについて検討してきた[1]. 本フレームワークは，サーバ・クライアント型のアーキテクチャを採用し，仮想オブジェクトをサーバが一元管理することで複数のクライアントで同一の MR 空間の共有を可能にする．また，MR 提示する仮想オブジェクトの動作を独自設計のスクリプト言語により制御する機構を開発し，複雑な動きの表現やインタラクション処理などが容易に可能となった[2]．しかしながら，実世界との融合という視点から考えた場合に重要となる物理法則に従った動作の制御が煩雑であるという問題を引き続き抱えていた．

そこで本研究では，我々が提案するフレームワークにおいて，仮想オブジェクトに対して質量・速度・摩擦などの古典力学的な法則を容易に適用可能な機構の実現を目指す．具体的には，物理演算エンジンを利用し，既存の MR 空間と整合性のある空間管理を行うことで，仮想オブジェクトを物理法則に則って動作させる機構を実現する．

2. コンテンツ制御機構

2.1 仮想オブジェクトの制御方法

本フレームワークにおける仮想オブジェクトの制御機構は，サーバに集約されており，スクリプトを常時実行することで，仮想オブジェクトの位置姿勢を制御している．クライアントで MR を実現するには，理想的には，フレーム更新に同期して仮想オブジェクトの位置姿勢の情報をサーバから取得する必要があるが，通信するデータ量と速度を勘案すると実現は困難である．そこで本フレームワークでは，サーバで生成した仮想オブジェクト情報をクライアントに一定間隔で送信し，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（**図 1**）.

補間処理には，3 次ベジエ曲線のアルゴリズムを応用し，クライアントの取得した 2 つの端点の位置情報に対し，速度ベクトル情報から制御点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[3].

2.2 独自スクリプト言語による仮想オブジェクト制御の問題

本言語は，MR 空間中に存在する CG などの仮想オブジェクトに対して，位置姿勢情報を制御するプログラムを記述することでその動作を制御する．プログラムでの動作記述により，無限に続く動作など自由な仮想オブジェクト動作の制御が可能である．

しかし，重力などの物理量を考慮した動作を本言語で記述することは，アプリケーション開発者であるオーサが全ての仮想オブジェクトに対し物理モデルを適用する必要

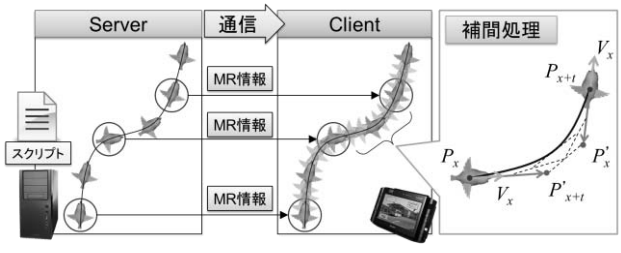

図1 コンテンツ制御機構の概略図

があり，煩雑であるという問題がある．そのため本研究では，記述が煩雑になりがちな仮想オブジェクトの物理的動作のために物理演算エンジンを導入し，仮想オブジェクトの物理演算を容易に制御可能な機構を実現する．

3. 物理演算エンジン制御機構

3.1 設計方針

物理演算を利用できることは便利ではあるが，必ずしもあらゆるアプリケーションで必要なわけではない．その一方で，利用する限りにおいてはできるだけ簡便な手段で利用できることが望ましい．そこで本研究では，以下の 2 つを要件として物理演算エンジンの導入方法を検討した．

- MR 空間内のすべての仮想オブジェクトが必ずしも物理法則に従う必要はない
- 簡単なメソッドによって容易に物理演算の適用・非適用が選択可能

これらの要件を満たすように以下の設計方針を定めた．

(1) 個々の仮想オブジェクト毎に，物理演算の適用・非適用を切り替えるためのメソッドを導入する．従前通り，すべての仮想オブジェクトを一元管理した上で，物理法則の影響を受ける仮想オブジェクトを別途管理する機構を導入する．

(2) 物理演算を行う上での標準的なパラメータを事前に設定し，できるだけ簡単なパラメータ設定で，物理演算の利用を開始できるようにする．

なお，本研究における物理演算は剛体を対象とし，重力や反発係数をはじめとする属性，オブジェクト同士が接続された際の影響を考慮したものとする．

3.2 物理演算エンジン制御機構の概要

本機構における空間構成を**図 2**に示す．本機構において，仮想オブジェクトは MR 空間を制御する MRWorld と物理演算エンジンによる物理演算が可能な空間を制御する PhysicsWorld の 2 つの空間で同時に管理する．MRWorld の仮想オブジェクトは，物理演算を行いたいときに限り PhysicsWorld で物理演算を行い，演算結果を MRWorld の仮想オブジェクトに適用することで，すべての仮想オブジェクトを MRWorld 内に混在可能とした．また，独自スクリプト言語により物理演算エンジンの利用を制御可能とすることで，従来との高い互換性を実現した．

しかし，物理的動作をクライアント上で再現する場合，これらの動作は従来以上に厳密な動作表現を必要とするため，補間処理による再現結果が不十分となる場合がある．そこで，従来の補間処理アルゴリズムを改良し，クライアントにおける仮想オブジェクトの物理的動作の再現をより正確に行えるよう変更した．詳細は 3.4 節で述べる．

3.3 物理演算エンジンを用いた仮想オブジェクト動作の制御

3.3.1 仮想オブジェクト情報の管理

本機構では MRWorld および PhysicsWorld の 2 つの空間において，それぞれに属する仮想オブジェクトを管理する．仮想オブジェクトは物理演算エンジンの利用が開始され

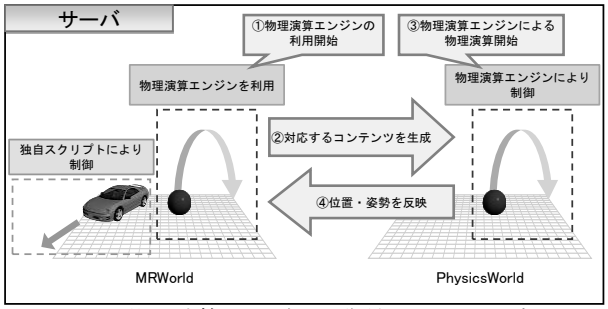

図2 物理演算エンジン制御機構の空間構成

た際に，同時に対応する仮想オブジェクトが PhysicsWorld 内に生成される．物理演算は，PhysicsWorld に生成された仮想オブジェクトに対して行われ，結果として算出された位置姿勢情報が常時 MRWorld 内の対応する仮想オブジェクトに反映される．このような仕組みにより，MRWorld における仮想オブジェクトの物理的動作を実現する．MRWorld 内の物理演算が適用されないオブジェクトは従前と同様にスクリプトによって制御されるため，MRWorld においては，物理演算エンジンを適用したものと非適用のものを混在させることが可能となる．

3.3.2 独自スクリプト言語仕様

本言語は，MR において仮想オブジェクト動作を制御するために必要な機能のみに特化することによって容易に利用できる設計となっている．

物理演算エンジンの利用方法として，独自スクリプト言語によって扱えるメソッドを**表 1**のように用意することで，従来では煩雑となっていた物理的動作を容易に制御可能とする．単一の仮想オブジェクトの移動・回転減衰等は仮想オブジェクトのインスタンスに対し物理的に考慮したい事項について記述することで，その属性を決定する．複数のオブジェクトに影響する設定項目である重力やジョイントは PhysicsWorldMethod という PhysicsWorld のインスタンスに対し記述を行う．ジョイントとは，複数のオブジェクトを接続した制限のある動作を可能とするもの

表1 物理演算エンジン利用のための独自スクリプト言語におけるメソッド例

メソッド名	概要
setGravity(double Xgravity, double Ygravity, double Zgravity);	重力値を設定
OnPhysics(Position3D pos,Orientation ori, double mass, double scale, int shapeType);	物理演算エンジンの利用開始
OffPhysics();	物理演算エンジンの利用停止
setVelocity(double Xvelocity, double Yvelocity, double Zvelocity);	速度を設定
setDamping(double linDamping, double angDamping);	移動・回転減衰を設定
setRestitution(double restitution);	反発係数を設定
setFriction(double friction);	摩擦係数を設定
PointJoint(Object obj1, Object obj2, Position3D pos1, Position3D pos2);	ポイントジョイントを設定
SliderJoint(Object obj1, Object obj2, Axis axis1, Axis axis2);	スライダジョイントを設定
HingeJoint(Object obj1, Object obj2, Position3D pos1, Position3D pos2, Rotation rot);	ヒンジジョイントを設定
setSliderLimit(Object obj1, Object obj2, double min, double max);	対象ジョイントに移動制限を設定
setHingeLimit(Object obj1, Object obj2, double min, double max);	対象ジョイントに回転制限を設定

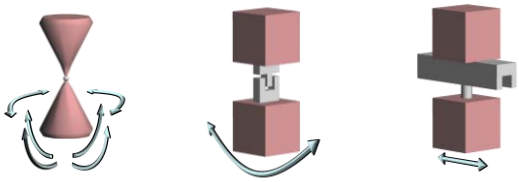

図3 ジョイントによるオブジェクト動作イメージ

である．各ジョイントは**図 3**に示すような移動あるいは回転の制限が与えられる．具体的にはポイントジョイントはふり子運動，スライダジョイントは 1 軸に対する並行運動のみ，ヒンジジョイントは 1 軸に対する回転運動のみが可能である．これらの記述により，単一の仮想オブジェクトの移動や衝突，落下，仮想オブジェクト同士が接続された状態での運動等を行うことができる．

3.4 補間処理アルゴリズムの拡張

3.4.1 クライアントで再現困難な仮想オブジェクト動作

2.1節において述べたように，クライアントはサーバからの仮想オブジェクト情報を元に 3 次ベジエ曲線を用いて仮想オブジェクトの動作を再現する．しかし，**図 4**に示すボールの踊躍などの特徴的な動作を再現する場合，3 次ベジエ曲線を用いた補間手法では，サーバから一定間隔で取得した 2 点間を滑らかな曲線により再現するため，鋭角な角度変化を含む動きを十分に再現できず，クライアントにおいて地平面に接しない軌跡を描く場合がある．この急速な角度変化を起こす境界の点をここでは尖点と定義する．物理演算エンジンを利用する場合，反発係数が正の値を持つ仮想オブジェクトを利用すると，仮想オブジェクト同士の衝突や，地平面に対する反発等，尖点を含む動きが頻出する可能性が高くなる．そこで尖点を含む仮想オブジェクト動作を，クライアントにおいて十分に再現可能とするため，尖点を考慮した補間処理アルゴリズムを検討する．

3.4.2 尖点を考慮した補間処理アルゴリズム

従来の補間処理アルゴリズムでは 3 次ベジエ曲線を用いて動作を再現してため，尖点の付近では丸みを帯びた動作となってしまう．尖点を含む仮想オブジェクト動作は，尖点を境とした 2 つの曲線の組み合わせとして考えられるため，尖点を算出し，これらの 2 曲線を，それぞれ 3 次ベジエ曲線を用いて時系列に算出することで尖点を含む仮想オブジェクト動作を再現する．尖点は仮想オブジェクト踊躍等の動作において速度ベクトルが急速な角度変化を起こす境界の点である．

図 4において，P_i をクライアントへと前回送信された点

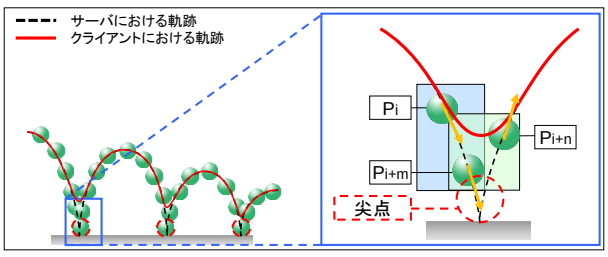

図4 サーバおよびクライアントにおける
仮想オブジェクト動作の軌跡と尖点

であるとし，次にクライアントへと送信するを P_{i+n} とする．P_{i+n} はサーバにおいて n フレーム後に算出された最新の点である．尖点の算出には，サーバにおいて毎フレーム算出される最新の仮想オブジェクトの速度ベクトル情報と，その前フレームで算出された速度ベクトル情報の成す角度を利用し，その角度が閾値を超えた場合にその 2 点間中に尖点が存在しているとする．しかし尖点は速度ベクトルを有さないため，3 次ベジエ曲線による補間処理ではサーバにおける動作の軌跡を十分に再現することができない．そのため，算出された尖点に最も近い速度ベクトルを有する点を尖点としてクライアントへと送信する．**図 4**においては，P_i から m フレーム後(0<m<n)の P_{i+m} が尖点として送信される．そして，尖点が仮想オブジェクト動作の軌跡において，いつ発生したのかを知るために，すべての仮想オブジェクト情報に対して時刻情報を付加する．クライアントは取得した尖点の情報および他の仮想オブジェクトにおける時刻情報をもとに，前回の点から尖点，尖点から最新の点と分割したうえで，3 次ベジエ曲線による補間処理を行うことで，尖点を含む仮想オブジェクト動作を再現可能となる．

4. 動作確認

4.1 実験目的

物理演算エンジン制御機構における物理演算エンジンの正しい動作と，独自スクリプトによる容易な制御を確認する．また，尖点を含む仮想オブジェクト動作がクライアントにおいて十分な再現を行えるのか動作確認を行う．

【実験 1】物理的動作の理論値との比較

YZ 平面において高さ 300[m]から地平面に対し並行に秒速 30[m]の速度を仮想オブジェクトに対し与え，Z 軸方向に 9.80665[m/s²]の重力のみを考慮する（**表 2**）．この仮想オブジェクト動作を独自スクリプト言語により記述し，加えて動作に対し同条件における物理学上の理論値を導き，それらの値を比較することで，物理演算エンジン制御機構の動作確認を行う．

【実験 2】物理的動作の容易な制御

複数の仮想オブジェクトが衝突し，反発する動作を行う．キー入力により仮想オブジェクトが生成され，初速度が与えられる．仮想オブジェクトは力が減衰することなく，衝突しあう．この動作を独自スクリプト言語によって記述し，物理的動作の制御における容易さを確認する．

【実験 3】尖点を含む仮想オブジェクト動作の再現

YZ 平面上において，仮想オブジェクトが Y 軸に対し踊躍する動作を行う．仮想オブジェクトは 250[m]を上限として上下し，水平方向の移動は等速度である．Z 軸の値が 0[m]の瞬間には確実に尖点が含まれるよう尖点算出の際の閾値を設定した．この動作の，クライアントおいて再現された軌跡を，従来の尖点を考慮していない補間処理アルゴリズムにより再現された軌跡と共に記録し，その結果から尖点を考慮した補間処理アルゴリズムの動作確認を行う．

表2 実験１における設定条件

設定項目	値
開始地点	XY 平面より 300[m]
重力	XY 平面に 9.80665[m/s²]
速度	30[ms]
移動・回転による減衰率	無し

リスト1 独自スクリプト言語による
物理演算エンジンの利用例

```
1  Sample extends MRWorld{
2  // Enter キーが押されたら，Ball を生成
3  if( client.getInteraction() ==System.INPUT.KEY_RETURN ){
4    OBJECT_LIST.add(new Ball(pos,ori));
5    Object obj = OBJECT_LIST.get( OBJECT_LIST .size()-1);
6    // 物理演算エンジンの利用開始
7    ball.OnPhysics(pos,ori,1.0,1.0,1);
8    // Ball の反発係数を 1.0 に設定
9    ball.setRestitution(1.0);
10   // Ball に対し X 軸方向に 30.0 の速度設定
11   ball.setVelocity(30.0,0.0,0.0);
12   // Ball の摩擦係数を 0.0 に設定
13   ball.setFriction(0.0); }}
```

4.2 実験結果

【実験 1】物理演算的動作の理論値との比較

図 5に結果を示す．物理演算エンジンを用いた仮想オブジェクトは，仮想オブジェクトの動作開始地点より 234[m]の地点において，地平面より約 2.21[m]の地点を記録している．また，動作開始地点より約 235.32[m]地点においては，地平面への到達を記録している．この結果より，物理演算エンジン制御機構による動作は物理学上の近似値を得たことから正しいことが確認できる．

【実験 2】独自スクリプト言語による制御

リスト 1のように仮想オブジェクト動作を記述した結果の動作例を図 6に示す．図 6における(1)-(5)は時系列順であり，(6)は(1)-(5)を時系列順に透明度を下げ，重ねあわせたものである．物理演算エンジン制御機構により，仮想オブジェクトの物理的動作をリスト 1における 7・9・11・13 行目の記述により制御できたことで，その容易さを確認できた．

【実験 3】尖点を含む仮想オブジェクト動作の再現

結果を図 7に示す．尖点を考慮していない従来の補間処理アルゴリズムでは Z 軸における最も 0 に近しい値は 34.8461[m]を記録しているのに対し，尖点を考慮した補間処理アルゴリズムでは Z 軸における最も 0 に近しい値は 3.15573[m]を記録している．この結果より，尖点を考慮した補間処理アルゴリズムは，尖点を考慮していない補間処理アルゴリズムと比較し，仮想オブジェクト動作をより正しく再現できていることが確認できた．

4.3 考察

前節の実験結果から，実装した物理演算エンジン制御機構が正しく動作していることを確認した．しかし，尖点を考慮した補間処理の場合，尖点から次点への軌跡が若干異なる場合がある．これは，尖点とした，尖点に最も近しい点の速度ベクトルを用いたことによる誤差であると考えられる．

5. むすび

本稿では，モバイル MR システムのためのフレームワークにおける物理演算エンジン制御機構について述べた．本

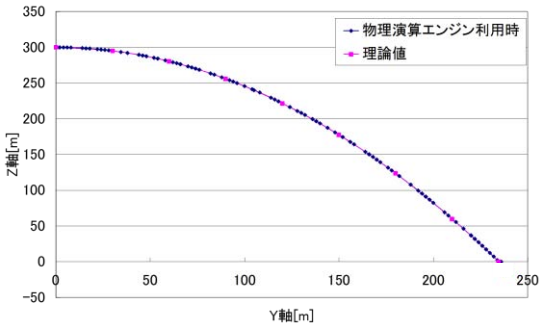

図5 独自スクリプト言語による
物理演算エンジン利用結果と理論値

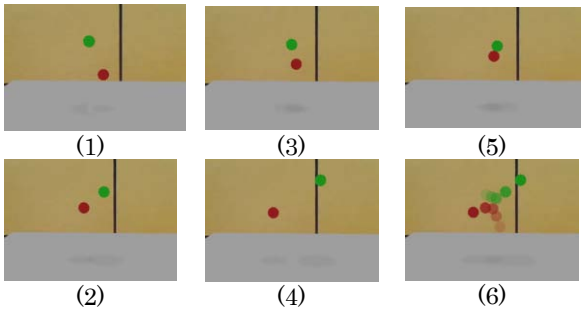

図6 仮想オブジェクトの衝突動作例

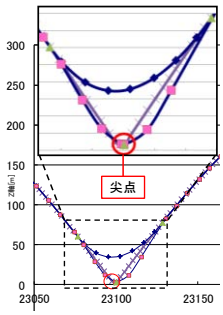

図7 尖点を考慮した補間処理および
従来の補間処理による補間結果

機構により，独自スクリプト言語による容易な記述により仮想オブジェクトの物理的動作を制御可能となる．今後は，端末性能による動作の再現差異を減少させ，仮想オブジェクトの物理的動作の再現を様々な端末において，より確かに表現する．

今後の課題として，尖点を考慮した補間処理アルゴリズムの更なる改良が考えられる．尖点に最も近しい点の速度ベクトルを用いて補間処理を行っているために，再現が不十分である場合がある．そのため，クライアントにおける尖点描画後の補間処理に用いる速度ベクトルを算出し，より正しい軌跡の再現を行う必要性がある．

参考文献

[1] 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 - 332,. 2005.

[2] 山下他：“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

[3] 縄谷他：“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（2）－コンテンツ制御機構の拡張－”，本大会，2010.