基礎論文

局所的な高速物理シミュレーションによる高解像度力覚提示の実現

須佐 育弥[*1]　大内 政義[*2]　岩下 克[*3]
佐藤 誠[*2]　長谷川 晶一[*1]

High Update Rate Local Dynamics Simulation for High Definition Haptic Display

Ikumi Susa[*1], Masayoshi Ohuchi[*2], Masaru Iwashita[*3],
Makoto Sato[*2]and Shoichi Hasegawa[*1]

Abstract – **In this paper, we present a technique of high quality haptic display working with low update rate rigid body simulator. The proposal method uses two dynamics simulators. One is low update dynamics simulator to simulate dynamics of whole virtual world and the other is high update rate dynamics simulator to simulate dynamics of objects nearby a haptic pointer which user controls (neighbor objects). Additionally, we calculate a mobility matrix of neighbor objects to consider contact force adding to neighbor objects. We did a simulation and experiment to check the effectiveness of the proposal method.**

Keywords : **haptic rendering, force feedback, rigid body simulation, multirate**

1　はじめに

ユーザがバーチャル世界とリアルタイムにインタラクションする方法として，力覚を使用する力覚提示システムがある．力覚提示システムの一つとして力覚インタフェースと物理シミュレータを組み合わせ，ユーザにバーチャル世界の物体を現実の物体を扱うかのように感じさせるシステムがある．このようなシステムではユーザは直感的な操作を実現できることから，設計，教育，訓練，エンタテイメント等への応用が期待されている．

力覚提示システムを使用して，人に力覚を提示する際，ユーザにバーチャル物体の形状を安定に力覚として提示するためには，1kHz 以上の更新周期で力覚インタフェースを制御する必要がある [1]．これにより，動的なバーチャル世界に対して力覚インタラクションをする場合はユーザの入力を物理シミュレーションに反映させなければならないので，物理シミュレータの更新周期も力覚インタフェースの制御周期に合わせなければならない．

例えば，長谷川ら [2] はペナルティ法に基づいた剛体物理シミュレーション手法を提案し，力覚インタフェースの制御とシミュレーションの更新を 300Hz として，力覚インタラクションを実現している．この手法はシミュレーション 1 ステップ当たりの計算量を抑えることができる．しかし，シミュレーションの更新速度が速いため，物体数が増加すると設定した更新速度で 1 ステップの計算を終えることができなくなるため，この手法では大規模で動的なバーチャル世界を構築することが難しい．

1.1　従来の研究

前述の問題に対して，Adachi ら [3] は力覚提示システムの処理をバーチャル世界の物理シミュレーションを行う物理プロセスと力覚レンダリングを行う力覚プロセスに分割し，プロセス間で通信を行う手法を提案している．この手法では，物理プロセスは力覚ポインタ近傍の物体の形状特徴を中間表現に変換し，力覚プロセスに送信する．力覚プロセスは力覚ポインタが中間表現に侵入した侵入量に応じてペナルティベースの力覚レンダリングを高速に行い，力覚インタフェースに提示力を送信する．

長谷川ら [4] は，物理プロセスの更新周期の間にユーザが入力した力積を力覚プロセスで計算し，その力積をもとに物理プロセスでバーチャル世界の運動を更新する手法を提案している．

これらの手法により，大規模なバーチャル世界を表現可能な力覚提示システムを構築することができた．しかし，これらの手法ではユーザの入力を物理シミュ

[*1]電気通信大学 知能機械工学科
[*2]東京工業大学 精密工学研究所
[*3]日立製作所 ソフトウェア事業部
[*1]University of Electro-Communications Department of Mechanical Engineering and Intelligent Systems
[*2]Tokyo Institute of Technology Precision and Intelligence Laboratory
[*3]Hitachi Ltd. Software Division

レーションに反映させる時に，プロセスの更新速度の違いからプロセス間の通信遅延が生じる．通信遅延により，ユーザの入力が物理シミュレーションに反映されてから，力がユーザにフィードバックされるまでに時間がかかる．例えば静止しているバーチャル物体をユーザが押すような状況では，ユーザが加えた力による物体の動きだしが遅れてユーザにフィードバックされる．そのため，ユーザが物体を押し始める時には物体が動かず，想定よりも大きな力をユーザに提示することになり正確な力覚を提示しているとはいえない．

1.2　研究の目的

前節で挙げた問題はプロセスの更新周期の違いに起因する．このことから，ユーザの入力をプロセスの同期待ちなしに物理シミュレーションに反映させることができれば，ユーザに正確に力覚を提示できると考えられる．そこで，本研究では動的で大規模なバーチャル世界を表現するための物理プロセスと力覚プロセスに分割された力覚提示システムにおいて，ユーザの入力をプロセスの同期待ちなしに物理シミュレーションに反映させ，正確に力覚を提示させることを目的とする．

2　提案手法

2.1　提案手法の概要

ユーザの入力をプロセスの同期待ちなしに物理シミュレーションに反映させるためには，1 章で説明したように，物理シミュレーションの更新周期を力覚インタフェースの制御周期に合わせなければならない．しかし，バーチャル世界の規模が大きくなるとシミュレーションの刻み幅内で計算が終わらない．

我々は大規模なバーチャル世界上での力覚提示を実現するために，プロセス分割したシステムの力覚プロセスにおいて局所的な物理シミュレーション (以下，局所シミュレーションと呼ぶ) を行う手法を提案してきた [5, 6, 7]．提案手法では，バーチャル世界の一部を力覚インタフェースの制御周期と同様の周期でシミュレーションさせることで，ユーザの入力をプロセスの同期待ちなしに物理シミュレーションに反映させることができる．これにより，正確に力覚を提示することが可能である．

本論文では，大規模なバーチャル世界において正確な力覚提示を実現する手法として，高速ループの力覚プロセスで実行させる局所シミュレーションと低速ループの物理プロセスで実行させるバーチャル世界全体の物理シミュレーション (以下，大域シミュレーションと呼ぶ) とで同期を取る手法を提案する．本手法は剛体の運動を適用範囲とし，大規模なバーチャル世界は多数の剛体をシミュレーション可能な世界とする．

2.2　局所シミュレーションを適用する範囲

力覚プロセスで行う物理シミュレーションは，シミュレーション対象をバーチャル世界の一部に限ることで計算量の削減を図る．局所シミュレーションの対象範囲として，例えば，

- 力覚ポインタに接触する可能性のある物体 (近傍物体)(図 1.(a)).
- 力覚ポインタから N 個の物体まで (図 1.(b)).

などを考えることができるが，次のような問題が生じる．図 1.(a) は近傍物体について局所シミュレーションを行うため，例えば近傍物体の周囲に接触する物体からの接触力を考慮していない．接触力を考慮しないことにより，近傍物体が重力によって落下してしまうなどの問題が起きる．図 1.(b) は力覚ポインタから N 個の物体までを局所シミュレーションの適用範囲としているが，間接的に接触している物体数が N 個を超えてしまうと，超えた物体の接触力を力覚ポインタに伝えることができず，正確に力覚を提示できない．

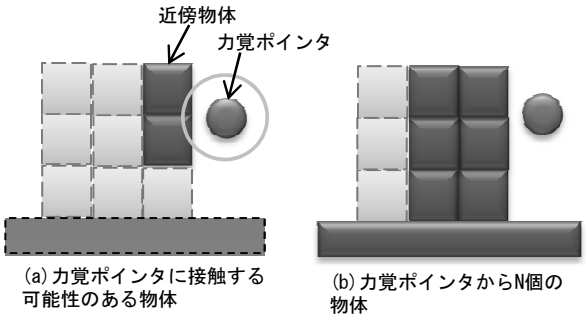

図 1　局所シミュレーションを適用する範囲
Fig. 1　Area of applying local dynamics calculation

以上のような問題に対して，我々はまず局所シミュレーションを適用する範囲を図 1.(a) とする．そして，物理プロセスにおいて通常の物理シミュレーションの更新からさらに 1 ステップ更新した時の近傍物体のモビリティを計算する．最後に計算した近傍物体のモビリティとユーザが実際に加えた力に基づいて，近傍物体を局所シミュレーションさせる．これにより，近傍物体に加わる接触力を全て考慮することができるので，正確に力覚が提示できると考えられる．

2.3　システム構成と処理の流れ

上述の提案手法を実現するためのシステム構成と処理の流れを図 2 に示す．また，以下に各プロセスの処理について説明する．

- 物理プロセス

1. 大域シミュレーション
   バーチャル世界全体の物理シミュレーションを行う．

2. 近傍物体の探索
   力覚ポインタに対して近傍の物体を探索する．

3. 近傍物体のモビリティの決定 (3 章)
   テストシミュレーションを行い， 近傍物体のモビリティを計算する．

4. 局所シミュレーションとの同期 (5.1 節)
   局所シミュレーションの結果を大域シミュレーションへ反映させる．

- 力覚プロセス

a. 大域シミュレーションとの同期 (5.2 節)
   近傍物体が受ける接触力以外の力を局所シミュレーションに反映させる．

b. 力覚ポインタの状態の更新
   力覚インタフェースから力覚ポインタの位置，姿勢を受信する．

c. 力覚レンダリング (4.1 節)
   提示力の計算をし，力覚インタフェースに提示力を送信する．

d. 局所シミュレーション (4.2 節)
   力覚ポインタが近傍物体に加えた提示力と，物理プロセスで求めたモビリティに基づいて，近傍物体の位置，姿勢，速度，角速度を更新する．

e. 物理プロセスが終わるまで b〜d の処理を繰り返す．

大域，局所シミュレーションの同期は物理プロセスのステップが終わる時点で行う．物理プロセスの更新周期に対する力覚プロセスの更新周期の比率を n とすると，力覚プロセスステップを n 回行っている間に物

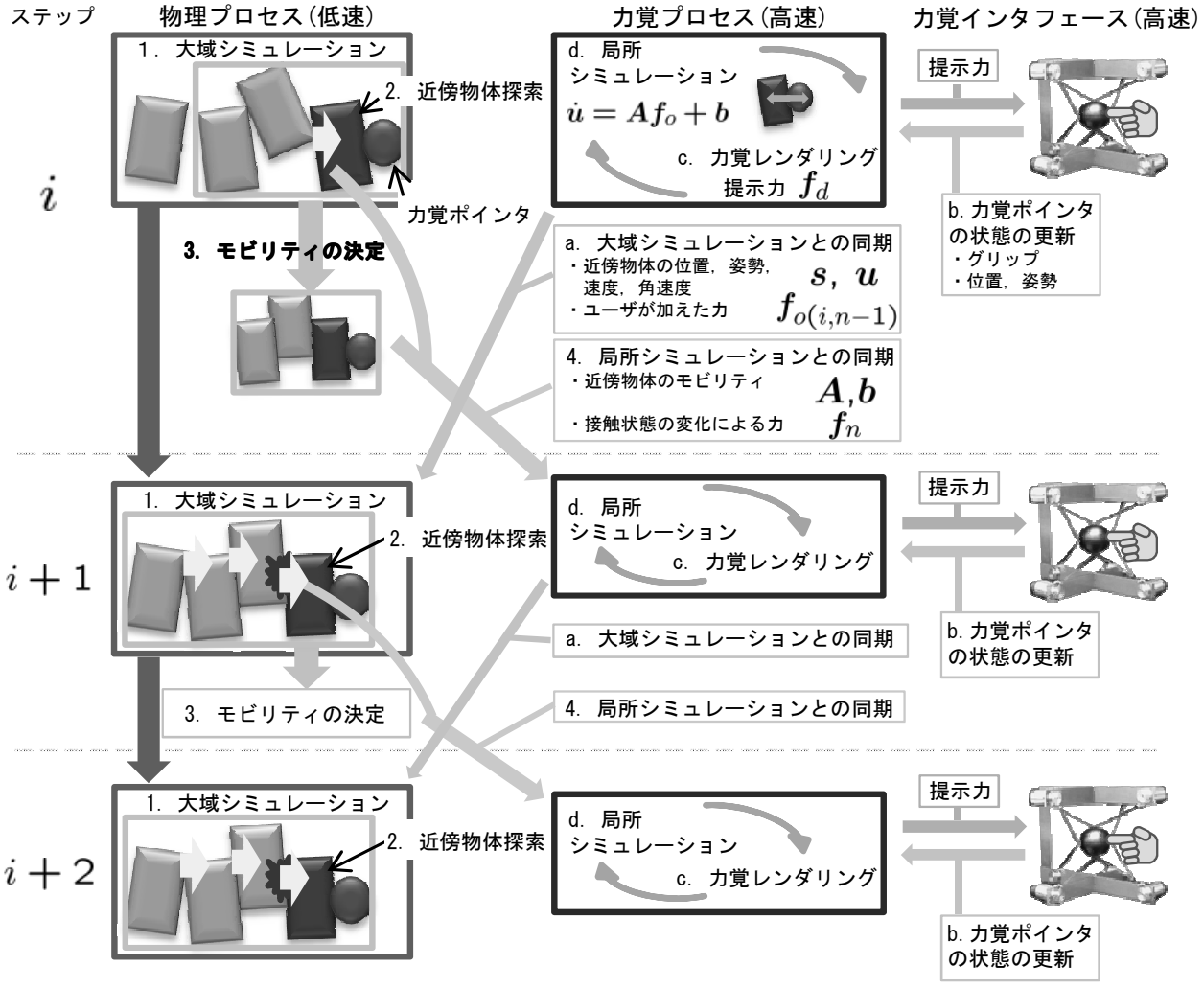

図 2　システム構成と処理の流れ
Fig. 2　System configuration and simulation flow

理プロセスステップを 1 回行い，シミュレーションを同期させる．

2.4　記号の表記

本論文では各プロセスステップで使用する記号を以下のように定義する．

- G: 物理プロセスで扱う変数の添え字
- L: 力覚プロセスで扱う変数の添え字
- i: 第 i 回目の物理プロセスステップ
- i,j: 第 (i,j) 回目の力覚プロセスステップ (第 i 回目の物理プロセスステップを行っている間の第 j 回目の力覚プロセスステップ．ただし，$0 \leq j < n$ で物理プロセスと同期後に 0 となる．)

例えば，第 i 回目の物理プロセスステップで大域シミュレーションを行い，ある剛体の位置，姿勢 $\boldsymbol{s}^G_{(i-1)}$ を更新したとすると $\boldsymbol{s}^G_{(i)}$ となる．また，第 (i,j) 回目の力覚プロセスステップで局所シミュレーションを行い，近傍物体の位置，姿勢 $\boldsymbol{s}^L_{(i,j-1)}$ を更新したとすると $\boldsymbol{s}^L_{(i,j)}$ となる．

次章からは各プロセスの処理の詳細について述べる．

3　物理プロセスの処理

2.2 節で述べたように，近傍物体を正確に局所シミュレーションさせるには近傍物体に接触する物体からの接触力を考慮する必要がある．本研究では近傍物体に加わる接触力を考慮するために，通常の大域シミュレーションからシミュレーションをもう 1 ステップ進め (テストシミュレーション)，近傍物体の速度，角速度変化から近傍物体のモビリティを計算する．そして，得られた近傍物体のモビリティを局所シミュレーションで使用することで，近傍物体に接触する物体からの接触力を考慮する．本章ではテストシミュレーションによる近傍物体のモビリティの決定方法について説明する．

3.1　近傍物体のモビリティ

力覚ポインタが近傍物体に加える力 $\boldsymbol{f}_o$ は，ユーザの操作に依存するため定まらない．そこで，近傍物体の運動と $\boldsymbol{f}_o$ の関係を考える．$\boldsymbol{f}_o$ と近傍物体の運動に線形性を仮定すると，近傍物体の運動はメカニカルインピーダンスを用いて，

$$\begin{aligned} \boldsymbol{M}\dot{\boldsymbol{u}} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{K}\int \boldsymbol{u}dt + \boldsymbol{f}_e &= \begin{pmatrix} \boldsymbol{f}_o \\ \boldsymbol{r} \times \boldsymbol{f}_o \end{pmatrix} \\ &= \boldsymbol{J}_h \boldsymbol{f}_o \end{aligned} \tag{1}$$

と書ける．ただし，

- $\boldsymbol{M}, \boldsymbol{B}, \boldsymbol{K}$: メカニカルインピーダンス行列 ($6 \times 6$ 行列)
- $\boldsymbol{u}$: 近傍物体の速度，角速度 (6 次元ベクトル)
- $\boldsymbol{f}_o$: ユーザが加える力 (3 次元ベクトル)
- $\boldsymbol{f}_e$: $\boldsymbol{f}_o$ 以外の外力 (6 次元ベクトル)
- $\boldsymbol{r}$: 近傍物体重心からの $\boldsymbol{f}_o$ の作用点
- $\boldsymbol{J}_h$: $\boldsymbol{f}_o$ を剛体に加える力とトルクに変換する行列 (6×3 行列)

である．

$\boldsymbol{f}_o$ が変化しても $\boldsymbol{u}$, $\int \boldsymbol{u}dt$ は変化しない．そこで式を変形させ，これらの定数をまとめると，

$$\begin{aligned} \dot{\boldsymbol{u}} &= \boldsymbol{M}^{-1}\boldsymbol{J}_h\boldsymbol{f}_o - \boldsymbol{M}^{-1}(\boldsymbol{B}\boldsymbol{u} + \boldsymbol{K}\int \boldsymbol{u}dt + \boldsymbol{f}_e) \\ &= \boldsymbol{A}\boldsymbol{f}_o + \boldsymbol{b} \end{aligned} \tag{2}$$

となる．$\boldsymbol{A}$ は近傍物体のモビリティ (メカニカルアドミタンス) で質量慣性成分を表す 6×3 行列，$\boldsymbol{b}$ は重力等の外力 (ユーザが加える力によって変化しない力) から由来する加速度項 (6 次元ベクトル) である．

近傍物体が他の物体と接触していたり，リンクやバネ，ダンパで接続されている場合でも，接触状態や接続が変化しない限り線形性の仮定は成り立つ．静止摩擦と動摩擦の切り替わりや接触の増減がある場合は線形性は成り立たないが，そのような変化は大域シミュレーションのステップでのみ起こると考え，局所シミュレーションでは扱わない．そのため，大域シミュレーションと局所シミュレーション間で整合性を保つ方法として接触状態の変化や静止摩擦と動摩擦の切り替わりにより発生する力を大域シミュレーションにて計算を行い，その力を局所シミュレーションに反映させる．詳しくは 5.2 節で説明する．

3.2　テストシミュレーションによる近傍物体のモビリティの決定

本研究では大域シミュレーション後にもう 1 ステップシミュレーションを進めるテストシミュレーションを 4 回行うことで近傍物体のモビリティ $\boldsymbol{A}$ と重力等の外力に由来する加速度項 $\boldsymbol{b}$ を決定する．ここでは第 i 回目の物理プロセスステップでのモビリティの決定について説明する．

力覚ポインタが近傍物体に加える力を $\boldsymbol{f}^G_{o(i+1)} = (0,0,0)^t$ と設定してテストシミュレーションを実行すると $\boldsymbol{u}^G_{0(i+1)}$ が得られる．ここで，大域シミュレーションの刻み時間を Δt^G として，式 (2) を差分形にすると，

$$\boldsymbol{u}^G_{(i+1)} = \boldsymbol{u}^G_{(i)} + (\boldsymbol{A}_{(i+1)}\boldsymbol{f}^G_{o(i+1)} + \boldsymbol{b}_{(i+1)})\Delta t^G \tag{3}$$

となる．$\boldsymbol{u}^G_{0(i+1)}, \boldsymbol{u}^G_{(i)}, \boldsymbol{f}^G_{o(i+1)}, \Delta t^G$ は既知なので式 (3) に代入すると，$\boldsymbol{b}_{(i+1)}$ を求めることができる．次に第 $(i-1, n-1)$ 回目の力覚プロセスステップから受信したユーザが実際に入力した力 $\boldsymbol{f}^L_{(i-1,n-1)}$ のノルムを使

用して $\boldsymbol{f}^G_{o(i+1)}$ を

$$\boldsymbol{f}^G_{o(i+1)} = \begin{cases} (||\boldsymbol{f}^L_{o(i-1,n-1)}||, 0, 0)^t (= \boldsymbol{f}^G_{o1}) \\ (0, ||\boldsymbol{f}^L_{o(i-1,n-1)}||, 0)^t (= \boldsymbol{f}^G_{o2}) \\ (0, 0, ||\boldsymbol{f}^L_{o(i-1,n-1)}||)^t (= \boldsymbol{f}^G_{o3}) \end{cases} \quad (4)$$

と設定し，それぞれについてテストシミュレーションを行い，1 ステップ先の近傍物体の速度，角速度を求める．ユーザが実際に入力した力を使用する理由は，より正確な近傍物体のモビリティを求めるためである．$\boldsymbol{f}^G_{o(i+1)}$ のノルムが小さい場合には静止摩擦力により，近傍物体は動かず，大きい場合には動摩擦により近傍物体は動く．ここでテストシミュレーションにより得られた近傍物体の速度，角速度を $\boldsymbol{u}^G_{1(i+1)}, \boldsymbol{u}^G_{2(i+1)}, \boldsymbol{u}^G_{3(i+1)}$ とすると，これらはそれぞれ式 (3) より，

$$\begin{cases} \boldsymbol{A}_{(i+1)}\boldsymbol{f}^G_{o1} &= \frac{\boldsymbol{u}^G_{1(i+1)} - \boldsymbol{u}^G_{0(i)}}{\Delta t^G} + \boldsymbol{b}_{(i+1)} \quad (= \boldsymbol{y}_1) \\ \boldsymbol{A}_{(i+1)}\boldsymbol{f}^G_{o2} &= \frac{\boldsymbol{u}^G_{2(i+1)} - \boldsymbol{u}^G_{0(i)}}{\Delta t^G} + \boldsymbol{b}_{(i+1)} \quad (= \boldsymbol{y}_2) \\ \boldsymbol{A}_{(i+1)}\boldsymbol{f}^G_{o3} &= \frac{\boldsymbol{u}^G_{3(i+1)} - \boldsymbol{u}^G_{0(i)}}{\Delta t^G} + \boldsymbol{b}_{(i+1)} \quad (= \boldsymbol{y}_3) \end{cases} \quad (5)$$

と書くことができる．さらに，式 (5) をまとめると

$$\boldsymbol{A}_{(i+1)} \begin{bmatrix} \boldsymbol{f}^G_{o1} & \boldsymbol{f}^G_{o2} & \boldsymbol{f}^G_{o3} \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_1 & \boldsymbol{y}_2 & \boldsymbol{y}_3 \end{bmatrix} \quad (6)$$

となり，両辺右側から $[\boldsymbol{f}^G_{o1}\boldsymbol{f}^G_{o2}\boldsymbol{f}^G_{o3}]^{-1}$ をかけると，

$$\boldsymbol{A}_{(i+1)} = \begin{bmatrix} \boldsymbol{y}_1 & \boldsymbol{y}_2 & \boldsymbol{y}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{f}^G_{o1} & \boldsymbol{f}^G_{o2} & \boldsymbol{f}^G_{o3} \end{bmatrix}^{-1} \quad (7)$$

となることから，モビリティ $\boldsymbol{A}_{(i+1)}$ が得られる．得られたモビリティ $\boldsymbol{A}_{(i+1)}$ と重力等の外力に由来する加速度項 $\boldsymbol{b}_{(i+1)}$ を第 $(i+1, 0)$ 回目の力覚プロセスステップに送信し，4.2 節で説明する局所シミュレーションにて使用する．

4 力覚プロセスの処理

4.1 力覚レンダリング

力覚ポインタの位置と近傍物体の情報からユーザへの提示力の計算を行う．提示力は力覚ポインタが近傍物体に侵入した場合に，力覚ポインタと近傍物体を繋ぐ仮想のばねダンパモデルを用い，力覚ポインタが物体に侵入した量とその微分に比例した値を提示力とする．ここで，

$\boldsymbol{f}_d$: ユーザへの提示力
k: ばね係数
d: ダンパ係数
$\boldsymbol{x}_p$: 力覚ポインタの位置
$\dot{\boldsymbol{x}}_p$: 力覚ポインタの速度
$\boldsymbol{x}^L_o$: 近傍物体と力覚ポインタとの接触点
$\dot{\boldsymbol{x}}^L_o$: 近傍物体の速度

とすると，ユーザへの提示力は，

$$\boldsymbol{f}_d = k(\boldsymbol{x}^L_o - \boldsymbol{x}_p) + d(\dot{\boldsymbol{x}}^L_o - \dot{\boldsymbol{x}}_p) \quad (8)$$

となる．また，近傍物体に加わる力 $\boldsymbol{f}^L_o$ は作用反作用の法則より，

$$\boldsymbol{f}^L_o = -\boldsymbol{f}_d \quad (9)$$

となる．

4.2 局所シミュレーション

4.1 章で計算した力 $\boldsymbol{f}^L_o$ を用い，近傍物体について局所シミュレーションを行う．局所シミュレーションで使用する近傍物体の運動を表す式は 3 章で導出した近傍物体のモビリティ $\boldsymbol{A}$，重力等の外力から由来する加速度項 $\boldsymbol{b}$ を使用して

$$\dot{\boldsymbol{u}}^L = \boldsymbol{A}\boldsymbol{f}^L_o + \boldsymbol{b} \quad (10)$$

となる．実際，第 $(i+1, j)$ 回目の力覚プロセスステップでの近傍物体の更新後の速度，角速度は式 (10) の差分形

$$\boldsymbol{u}^L_{(i+1,j)} = \boldsymbol{u}^L_{(i+1,j-1)} + (\boldsymbol{A}_{(i+1)}\boldsymbol{f}^L_{o(i+1,j)} + \boldsymbol{b}_{(i+1)})\Delta t^L \quad (11)$$

として表すことができる．ただし，Δt^L は局所シミュレーションの刻み時間である．また，更新後の近傍物体の位置，姿勢 $\boldsymbol{s}^L$ は

$$\boldsymbol{s}^L_{(i+1,j)} = \boldsymbol{s}^L_{(i+1,j-1)} + \boldsymbol{S}\boldsymbol{u}^L_{(i+1,j)}\Delta t^L \quad (12)$$

となる．ここで $\boldsymbol{s}$ は位置表現を 3 次元デカルト座標 $\boldsymbol{x}$，姿勢表現を四元数 $\boldsymbol{q} = \begin{pmatrix} q_w & q_x & q_y & q_z \end{pmatrix}^T$ として $\boldsymbol{s} = \begin{pmatrix} \boldsymbol{x} & \boldsymbol{q} \end{pmatrix}^T$ で，$\boldsymbol{S}$ は角速度を四元数に変換する行列

$$\boldsymbol{S} = \begin{pmatrix} E & 0 \\ 0 & Q \end{pmatrix} \quad (13)$$

で，$\boldsymbol{E}$ は 3×3 の単位行列，$\boldsymbol{Q}$ は

$$\boldsymbol{Q} = \frac{1}{2} \begin{pmatrix} -q_x & -q_y & -q_z \\ q_w & -q_z & q_y \\ q_z & q_w & -q_x \\ -q_y & q_x & q_w \end{pmatrix} \quad (14)$$

である．

力覚プロセスステップを n 回行った後，近傍物体の位置，姿勢，速度，角速度とテストシミュレーションのためのテスト力 $f^L_{o(i+1,n-1)}$ を第 $i+2$ 回目の物理プロセスステップへ送信する．

5　大域，局所シミュレーションの同期処理

5.1　大域シミュレーションへ局所シミュレーション結果を反映

近傍物体の局所シミュレーション結果を大域シミュレーションに反映させる．反映は大域シミュレーションの近傍物体の位置，姿勢，速度，角速度を局所シミュレーションの近傍物体の位置，姿勢，速度，角速度に置き換える．実際，第 $i+2$ 回目の物理プロセスにおいて，第 $(i+1)$ 回目の力覚プロセスの局所シミュレーション結果を反映させると，

$$s^G_{(i+2)} = s^L_{(i+1,n-1)} \tag{15}$$

$$u^G_{(i+2)} = u^L_{(i+1,n-1)} \tag{16}$$

となる．これにより，ユーザの入力を正確に物理シミュレーションに反映できる．

5.2　局所シミュレーションへ接触力以外の力の反映

近傍物体の局所シミュレーションは近傍物体に接触している物体からの接触力をモビリティとして考慮しているが，近傍物体と他物体の接触状況や静止摩擦，動摩擦の切り替わり等，線形的に変化しない力は考慮していない．これらの力を考慮しないと近傍物体に接触している物体が近傍物体に侵入する，近傍物体が受けた衝突力がユーザの手に伝わらない等の問題が起こる．そのため，大域，局所シミュレーション間での整合性を取る必要がある．

提案手法では大域シミュレーションで近傍物体に加わる非線形に変化する接触力以外の力 $\boldsymbol{f}^G_n$ を計算し，力覚プロセスを n 回行ううちの 1 回目に反映させ，大域，局所シミュレーションで扱う近傍物体の速度，角速度の整合性をとる．例えば，第 $i+1$ 回目の物理プロセスステップでの大域シミュレーションで近傍物体が衝突や摩擦状態の変化で力 $\boldsymbol{f}^G_{n(i+1)}$ を受けた場合，第 $(i+2,0)$ 回目の力覚プロセスステップの局所シミュレーションへ力 $\boldsymbol{f}^G_{n(i+1)}$ を反映させる．このとき第 $(i+2,0)$ 回目の力覚プロセスステップでの局所シミュレーションによる近傍物体の運動の更新は以下の式に従って行う．

$$\begin{aligned}\boldsymbol{u}^L_{(i+2,0)} &= \boldsymbol{u}^L_{(i+1,n-1)} \\ &\quad +(\boldsymbol{A}_{(i+1)}\boldsymbol{f}^L_{o(i+2,0)} + \boldsymbol{b}_{(i+1)})\Delta t^L \\ &\quad +\boldsymbol{M}^{-1}\boldsymbol{J}^G_{n(i+1)}\boldsymbol{f}^G_{n(i+1)}\Delta t^G \end{aligned} \tag{17}$$

ただし，M^{-1} は近傍物体の質量慣性行列 $(6\times6$ 行列$)$，$\boldsymbol{J}^G_n$ は 力 $\boldsymbol{f}^G_n$ を力とトルクに変換するヤコビアン $(6\times3$ 行列$)$ である．式 (17) の右辺第 1 項と第 2 項は 4.2 節で説明した式 (12) と相当し，第 3 項で近傍物体に接触力以外の力 $\boldsymbol{f}^G_{n(i+1)}$ を加え速度，角速度を変化させる．このようにすることで，大域，局所シミュレーション間で整合性を取ることができる．

6　評価

提案手法の有効性を確認するために，計算機シミュレーションによる力覚レンダリングの評価とシミュレーションに必要な計算時間を計測した．

6.1　実験システム

以下に実験を行うにあたって実装したシステムの詳細を示す．

- 計算機

  Intel(R) Core(TM)2 Duo CPU 2.33GHz

- 力覚インタフェース　SPIDAR-G6[8]
- 物理シミュレータ　Springhead2[9]

また，力覚提示の手法として下表 1 の 3 種類をシステムに実装した．シミュレーションの刻み時間は各手法の物理プロセスのの更新周期と同値である．

表 1　システムに実装した手法
Table 1　Techniques mounted on the system

手法	物理プロセス更新周期	力覚プロセス更新周期
シミュレータを高速更新する手法	1ms	1ms
提案手法 (局所シミュレーションを行う手法)	50ms	1ms
従来手法 (シミュレータに力積を送信する手法 [4]	50ms	1ms

6.2　計算機シミュレーションによる力覚レンダリングの評価

提案手法が従来手法と比較して正確に力覚レンダリングができているのかを評価するための計算機シミュレーションを行った．

6.2.1　シミュレーション内容

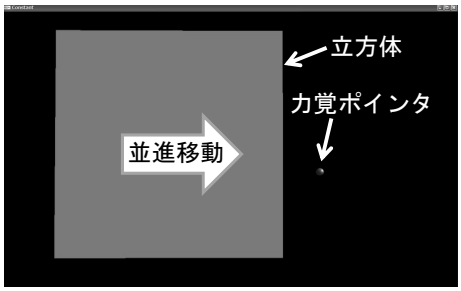

図 3　構築したバーチャル世界
Fig. 3　The virtual world for experiment

図 3 のような重力のない 3 次元バーチャル世界上で立方体を並進移動させる．この立方体を球の形をした力覚ポインタに衝突させ，はね返る間に力覚ポインタが立方体に加える力を計測する．力覚レンダリングに

よる力を全て立方体に加えるために，力覚ポインタの位置更新，速度更新はせず，その場に留まるように設定した．また，立方体の回転運動が力覚レンダリングに影響することを考慮し，立方体は回転運動をさせず，力が立方体の面に対して垂直に加わるようにした．

上記のシミュレーションを実装した 3 手法 (シミュレータを高速更新する手法，提案手法，従来手法) について立方体の質量 60kg，速度 0.017m/s で行った．

6.2.2 シミュレーション結果

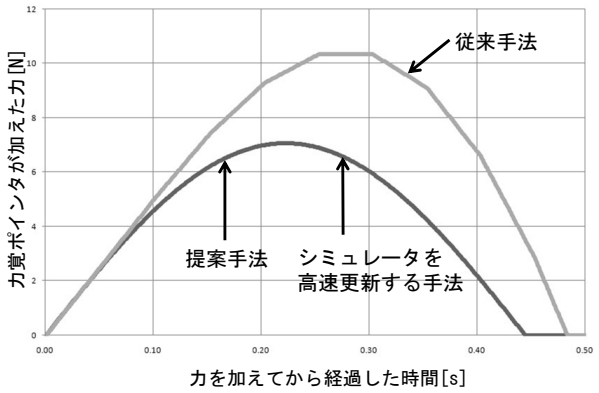

図 4 シミュレーション結果
Fig. 4 Result of the simulation

図 4 にシミュレーション結果を示す．図 4 から従来手法はシミュレータを高速に更新する手法に比べて，立方体と力覚ポインタとの接触時間が長く，立方体へ加える力が大きくなっている．一方，提案手法は立方体と力覚ポインタとの接触時間と力がシミュレータを高速に更新する手法と同様の軌跡が得られている．

次に立方体と力覚ポインタが接触している間に力覚ポインタが立方体に加えた力積を表 2 に示す．力積の値から，従来手法は他の手法に比べて大きな力積を立方体に加えており，提案手法はシミュレータを高速更新する手法と同値であった．このことから提案手法は従来に比べてより正確に力覚レンダリングができていると考えられる．

また，参考として力覚インタフェースを用いてバーチャル世界内で移動してくる立方体を被験者が受け止める実験を行い，提案手法と従来手法に有意差があるかを調べた．実験の結果，提案手法と従来手法で有意差は得られなかったが，従来手法のほうが立方体を受け止めた時のはね返りが強いという感想が被験者から得られた．

6.3 計算時間の評価

提案手法では通常の物理シミュレーションに加え，テストシミュレーションを行っている．そこで，提案手法のテストシミュレーションがどの程度計算時間を増加させているか調査するために，シミュレーション

表 2 力覚ポインタが立方体に加えた力積
Table 2 Momentum added by haptic pointer

	力積 [Ns]
シミュレータを高速更新する手法	2.0
提案手法	2.0
従来手法	3.2

の 1 ステップ当たりの計算時間を測定した．

6.3.1 実験内容

図 5 近傍物体に接触する物体の影響
Fig. 5 Affect of object contacting with neighbor object

図 5 のような床の上にある立方体を力覚ポインタで押している時の物理シミュレーション 1 ステップ当たりの計算時間を計算機の CPU クロック時間を利用して計測した．計測はシミュレータを高速更新する手法，提案手法，従来手法で行い，提案手法についてはテストシミュレーションの計算にかかる時間も合わせて計測した．また，物理シミュレーションされる物体の個数によって計算時間が変化するので，各手法について立方体の個数を 1，10 個にして計測した．個数が 10 個の場合は始めは立方体同士は接触しておらず，力覚ポインタで押していくうちに接触するようにした．

6.3.2 計測結果

計測結果を図 6 に示す．図 6 のグラフの縦軸は物理シミュレーション 1 ステップの計算時間を刻み時間で除算した割合で，刻み時間当たりの計算占有率である．計算占有率は刻み時間内に計算が完了しているかどうかをみる指標で，計算占有率が 1 を下回る場合は刻み時間内に計算が完了することを意味し，1 を超える場合は刻み時間内に計算が完了しないことを意味する．

まず，図 6 左側から物体の数の違いによる計算占有率の変化に着目してみると，各手法について立方体の数が多いほうが物理シミュレーション 1 ステップの計算占有率が増加している．シミュレータを高速更新する手法では立方体の数が 1 個の場合は計算占有率は 0.15 を下回りリアルタイムに計算できているが，立方体の数が 10 個となると計算占有率が 1 を超えるよう

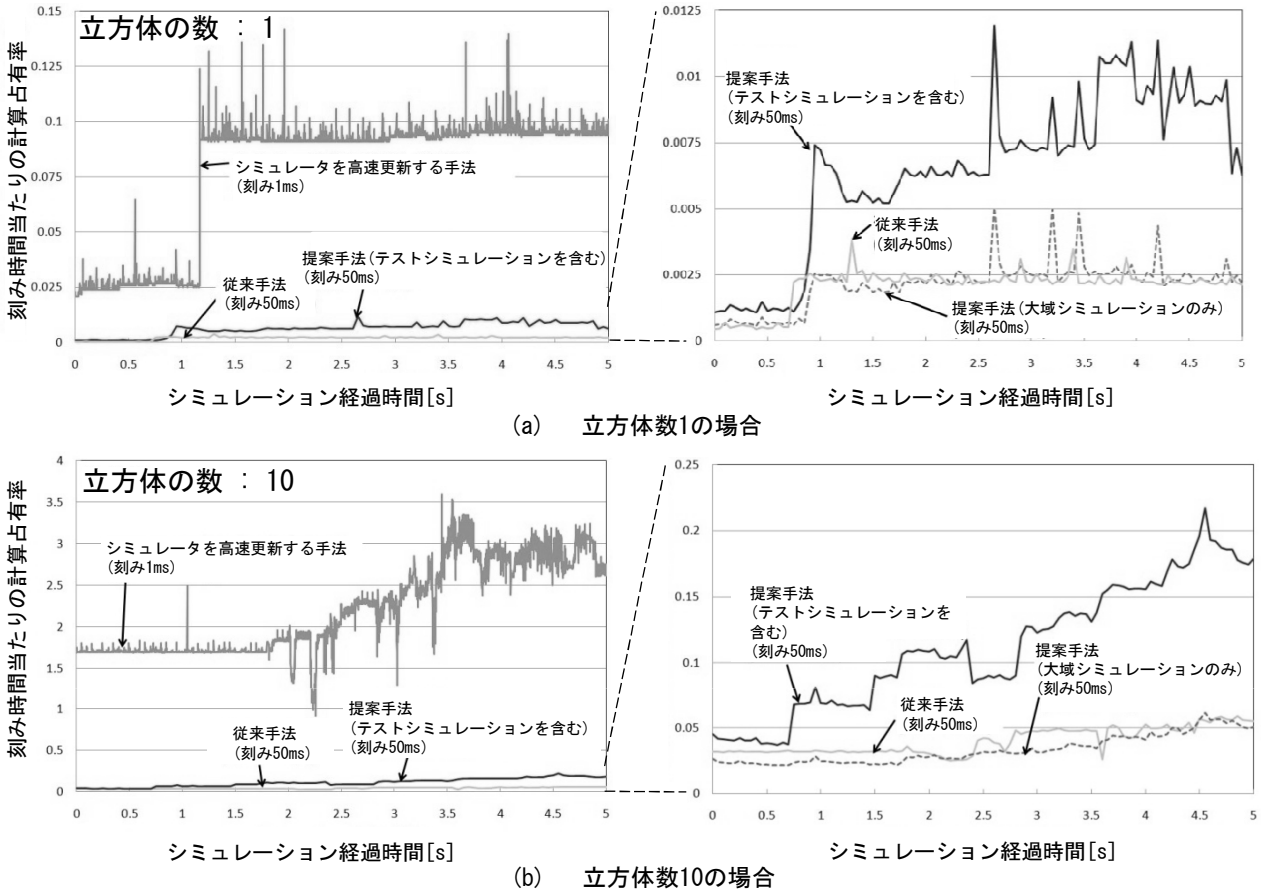

図 6　計算時間の測定結果
Fig. 6　Result of computation time

になり，設定した刻み時間 1ms 内で計算が終わらず，刻み時間の約 3 倍の計算時間を要している．実際に時間計測中の力覚インタフェースを用いた力覚提示も発振が起きるようになり不安定であった．提案手法，従来手法については立方体 10 個の場合についても計算占有率が 0.25 を下回り，リアルタイムに計算を終えている．力覚インタフェースを用いた力覚提示も発振は起きず，安定に力を提示できていた．

次に，提案手法と従来手法の結果を拡大したもの (図 6 右側) についてみる．参考のため，提案手法の結果の内テストシミュレーションを含まない大域シミュレーションのみを測定したものを破線で示す．提案手法 (大域シミュレーションのみ) と従来手法では立方体の数の変化で計算占有率に大きな差は見られなかった．提案手法 (テストシミュレーションを含む) と従来手法を比較すると提案手法 (テストシミュレーションを含む) は従来手法に比べ，約 3 倍から 4 倍となっている．これは，テストシミュレーションで力覚ポインタが力を加えない場合とそれぞれ 3 方向に力を加える場合の計 4 回のシミュレーションを行い，接触する立方体の個数に応じて拘束力の計算をしているため，計算時間が増加しているといえる．

以上のように，提案手法はシミュレータを高速更新する手法に比べて多くの物体をリアルタイムにシミュレーションすることができている．また，従来手法に比べて，6.2 節での実験の結果から正確に力覚提示ができているが，テストシミュレーションの分だけ計算時間が増加していることがわかった．

7　考察

7.1　近傍物体のモビリティの効果

近傍物体に接触している物体の接触力が近傍物体のモビリティで考慮されているかを確認するために，図 5 のような状況で力覚ポインタを使い立方体を押し，提示される力の変化を確かめた．結果，立方体同士が接触するごとに手にかかる力が大きくなっていることを感じ，近傍物体のモビリティが近傍物体に接触している物体からの接触力を考慮していることの確認がとれた．

また，6.2 節の実験では近傍物体の運動を並進移動に制限しており，近傍物体の回転運動について評価はしていない．しかし，力覚インタフェースを用いて立

方体の形状をした近傍物体の辺に近い部分に力を加えると，違和感なく近傍物体が回転していることを確認した．

7.2 提案手法の適用範囲

本論文では対象を剛体に絞って議論を進めた．有限要素解析が必要な弾塑性体については検討していない．提案手法を弾性体に適応させるには更に検討が必要だと考えられる．

8 まとめと今後の展望

従来ではプロセス分割された力覚提示システムにおいて，プロセス間での通信遅延のため正確に力覚提示を行うことができなかった．本論文では力覚プロセス内でバーチャル世界の一部を高速にシミュレーションする局所シミュレーションによる手法を提案した．そして，正確に力覚提示が実現できているか評価するためのシミュレーションを行い，提案手法が従来手法に比べて有効であることを確認した．また，各手法の計算時間を測定し．シミュレータを高速更新する手法に比べて提案手法は大規模なバーチャル世界上で力覚提示が可能であることを確認した．これらの評価から提案手法によりプロセス分割された力覚提示システムにおいても正確に力覚提示を実現できることがいえる．

今後の展望として，今回提案した手法の近傍物体のモビリティの決定をするテストシミュレーションは力覚ポインタと近傍物体の接点 1 点に対してテスト力を加え近傍物体の運動の変化をみるものである．そのため，力覚ポインタと近傍物体が面で接触している場合にはテストシミュレーションによる正しい結果が得られず，安定に力覚提示ができない可能性がある．現在，このような力覚ポインタと近傍物体が面接触している場合の近傍物体のモビリティの決定方法についての研究を進めている．将来，提案手法の適用範囲を剛体に限らず，弾塑性体，流体にまで拡張し，膨大な計算量を必要とするバーチャル世界への力覚インタラクションの応用が期待できる．

参考文献

[1] Lonnie Love, Wayne Book: Contact Stability Analysis of Virtual Walls; Proc. Of Dynamic Systems and Control Division ASME, pp.689-694 (1995)

[2] 長谷川晶一, 藤井伸旭, 赤羽克仁, 小池康晴, 佐藤誠: 力覚インタラクションのための多面体接触面積に基づくリアルタイム剛体シミュレーション; 計測自動制御学会論文集, Vol.40, No.2, pp122-131 (2004)

[3] Y. Adachi, T. Kumano, K. Ogino: Intermediate representation for stiff virtual objects; Proc. IEEE Virtual Reality Annual International Symposium, pp.203-210 (1995)

[4] 長谷川晶一, 石井雅博, 小池康晴, 佐藤誠: 力覚ディスプレイをもつ動的な仮想世界のための力積に基づくプロセス間通信; 電子情報通信学会論文誌 D-II, Vol.J82-D-II, No.10, pp.1758-1765 (1999)

[5] 岩下克, 赤羽克仁, 長谷川晶一, 小池康晴, 佐藤誠: 局所的な物理シミュレーションを行う力覚レンダリング; 電子情報通信学会技術研究報告. MVE, マルチメディア・仮想環境基礎, Vol.105, No.106, pp.73-78 (2005)

[6] 大内政義, 長谷川晶一, 小池康晴, 佐藤誠: 物理シミュレータ上での力覚提示のための局所的な物理シミュレーションを行う力覚レンダリング; 電子情報通信学会技術研究報告. MVE, マルチメディア・仮想環境基礎, Vol.106, No.611, pp.1-6 (2007)

[7] 須佐育弥, 長谷川晶一: 解析法の物理シミュレータのための局所的な物理シミュレーションを行う力覚レンダリング; ヒューマンインタフェース学会研究報告集, Vol.10, No.2, pp.57-62 (2008, 6)

[8] 佐藤誠, 平田幸広, 河原田弘: 空間インタフェース装置 SPIDAR の提案; 電子情報通信学会論文誌, Vol.J74-D-2, No.7, pp.887-894 (1991)

[9] Springhead2: http://springhead.info/wiki/

[10] 日本視覚学会: 視覚情報処理ハンドブック; 朝倉書店, pp.219-220 (2000)

(2009 年 2 月 5 日受付)

［著者紹介］

須佐 育弥 （学生会員）

2008 年電気通信大学電気通信学部知能機械工学科卒業，現在，同大学大学院電気通信学研究科知能機械工学専攻修士課程在学中．力覚レンダリングの研究に従事．

大内 政義

2007 年東京工業大学大学院総合理工学研究科修了．在学中は力覚レンダリングに関する研究に従事．工学修士．

岩下 克

2003 年東京工業大学工学部情報工学科卒業，2005 年同大学大学院知能システム科学専攻修士修了，同年，株式会社日立製作所入社，現在に至る．在学中はバーチャルリアリティの研究に従事．工学修士．

佐藤 誠 （正会員）

1973 年東京工業大学工学部電子物理工学科卒業，1978 年同大学院博士課程修了，同年，同大学工学部助手，現在，同大学精密工学研究所教授，現在に至る．パターン認識，画像処理，ヒューマンインタフェースの研究に従事．工学博士．

長谷川 晶一 （正会員）

1997 年東京工業大学工学部電気電子工学科卒業，1999 年同大学大学院知能システム科学専攻修士修了，同年ソニー株式会社入社，2000 年東京工業大学精密工学研究所助手，2007 年電気通信大学知能機械工学科准教授，現在に至る．バーチャルリアリティ，力覚インタフェース，ヒューマンインタフェースの研究に従事．工学博士．