芸術科学会論文誌 Vol. 9 No. 2 pp. 38-48

分散型衝突検出手法に基づく局所領域の力覚提示システム

高橋哲也*, 今野晃市*, 曽根順治**, 徳山喜政**, 原美 オサマ*
*岩手大学 工学研究科, **東京工芸大学 工学部

Haptic Interaction with Complex Objects Based on Local-domain Forces and Distributed Collision Detection Approach

Tetsuya Takahashi, Kouichi Konno*, Junji Sone**, Yoshimasa Tokuyama**, Osama Halabi**
*Graduate School of Engineering, Iwate University
**Faculty of Engineering, Tokyo Polytechnic University
E-mail: konno@cis.iwate-u.ac.jp

概要

ハードウェアの低価格化，高性能化により，リアリティのある Virtual Reality(VR)環境構築は容易に実現可能になってきている．VR 技術は医学，工学，教育，ゲーム，グラフィック関連のアプリケーションなどでの利用が期待されている．従来，VR 環境構築に関する要素技術は数多く研究されている．要素技術の一つである衝突検出法は，計算コストが高くインタラクティブ性を損なう一因である．一つの考え方として，PC クラスタなどによる分散処理によって，高速化することで問題点を解決できる．本論文では，PC クラスタで動作する衝突検出法を用いたバーチャルタッチングシステムを提案する．本システムは，大量のポリゴンを有する複雑な対象物の衝突を高速に検出し，衝突部分の局所領域に関して力覚を提示するシステムである．本手法は，大量のポリゴンを有する複雑な対象物の衝突を PC クラスタを用いて高速に計算し，レスポンスを向上する．また，計算量が少なく処理が高速な力覚提示スレッドで擬似的に力覚を生成することによって，スムーズな力覚提示を行うことが可能である．

キーワード: 仮想現実，力覚提示，衝突検出，擬似的な力覚，ＰＣクラスタ

Abstract

This paper presents a virtual haptic interaction system that employs a distributed collision detection algorithm using PC-cluster. The system achieved very fast collision detection with complex objects that have a large number of polygons and forces were applied in the local domain of the collision area. Using PC-cluster to calculate collision improves the response significantly. In addition, a pseudo force generation approach is proposed to smooth the force feedback produced in the haptic loop.

Keyword: Virtual Reality, Force Feedback Presentation, Collision Detection, Artificial Force, PC Cluster

1. はじめに

ハードウェアの低価格化，性能向上により，リアリティのある Virtual Reality(VR)環境構築は容易に実現可能になってきている. VR 技術は医学, 工学, 教育, ゲーム，グラフィック関連のアプリケーションなどでの利用が期待されている．従来，VR 環境構築に関する要素技術は数多く研究されている．例えば，VR 環境構築のための要素技術として，空間分割法などを利用した衝突検出法[1]があげられる．衝突検出法は，力覚提示の際に，どの部分が接触しているのかを判定するために用いられる．一般に，衝突検出は計算コストが高く，インタラクティブ性を損なう一因となる．

前述した問題点を解決するための考え方として，PC クラスタを用いて衝突を検出することで，全体のレスポンスを向上する方法がある[1]．具体的には，莫大な計算を PC クラスタへ分散化し計算結果を統合しながら参照する VR システムを構築する．また，クライアント PC では，スレッドプログラミングにより複数のスレッドを生成し，力覚提示スレッドと衝突計算を行う分散衝突検出スレッドの 2 つのスレッドに処理ループを分けることでレスポンスの問題は解決できる[2]．

本論文では, PC クラスタを利用した衝突検出法を用いたバーチャルタッチングシステムを提案する．本システムは，大量のポリゴンを有する複雑な対象物の衝突を高速に検出し，衝突部分の局所領域に関して力覚を提示するシステムである．本手法は，大量のポ

リゴンを有する複雑な対象物の衝突を PC クラスタを用いて高速に計算し，衝突箇所を特定する．また，計算量が少なく処理が高速な力覚提示スレッドで力覚を生成することによって，スムーズな力覚提示を行うシステムを構築する．具体的には，ハプティックデバイスと連動して，手のモデルを自由に動かし，対象物に触れた位置での力覚を提示する．このとき，手のモデルと対象物の衝突計算を PC クラスタで分散処理することでレスポンス時間を短縮し，精確な力覚提示を実現する．

2. 関連研究

従来の分散型衝突検出法については，バウンディングボックスや Octree などの方法を用いて，よりリアルタイム処理に近づけるための衝突検出法の研究が行われている[3]．しかし，精確な衝突検出を行おうとすると安価な PC では時間がかかってしまうという問題がある．また，Graphics Processing Unit(GPU)を用いて衝突検出を高速化させる方法[4]も提案されている．しかし文献[4]の手法では，GPU に搭載されるメモリ量の制限により，一定数以上のポリゴンモデルを扱うことができない，という問題がある．そこで，安価な PC をネットワークで結合した PC クラスタを利用して，膨大なポリゴン数を持つポリゴンモデルに対するパフォーマンスを改善する方法が提案されている[5]．このように，PC クラスタは広く活用されているが，衝突検出や力覚提示を統合化した実用的な VR システムは少ない．B. Raffin らは， PC クラスタを用いて，衝突検出のような計算コストの高い処理に関する計算時間を軽減することが可能になると述べている[6]．彼らの手法は UNIX 環境で実現されており，分散処理に MPI[7]を想定しているので，専用ライブラリが必要である．

一方，Windows 環境では，分散処理に Windows OS で提供されている DCOM (Distributed Component Object Model)[8]を用いることができる． DCOM とは，オブジェクト間でデータの交換や処理依頼のやり取りなどをネットワークを通じて行うモデルである．DCOM は，Windows OS 上では特別なソフトウェアを必要としないため，容易に計算機環境を構築することが可能である． 本研究では，クラスタ化する PC はネットワーク上にある遊休 PC を活用することを目指しているので DCOM は最適と考える．

力覚提示手法については，バーチャル世界全体から接触部分近傍の物体形状を抽出する処理を低速更新で行い，抽出した形状を高速に力覚提示する手法が提案されている[9,10]．これらの手法では，物体を突付く動作を基準にしており，複雑な物理計算が必要である．また，抽出した近傍の面積に対するポリゴン数が多い場合，すなわち，ポリゴンがより複雑な形状になる場合，力覚提示のレスポンスに影響が出ることが考えられる．本手法では，PC クラスタを利用した分散型衝突検出手法を用いて，接触部分を高速に検出して幾何情報に基づいて力覚を提示する．

マルチスレッドを利用して，力覚提示をリアルタイムに近づける方法[11]が提案されているが，具体的なシステム構成については述べられていない．

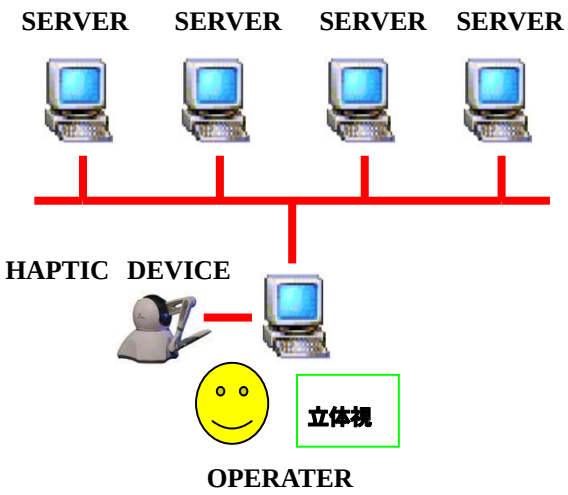
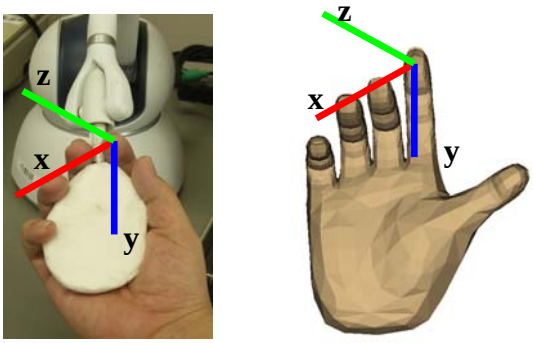


図 1．クライアント･サーバモデル

図 2．PC クラスタ

3. 提案するバーチャルタッチングシステム

3.1 ハードウェア構成



(a) (b)

図 3．PHANToM Omni に器具を装着した様子と Hand モデル

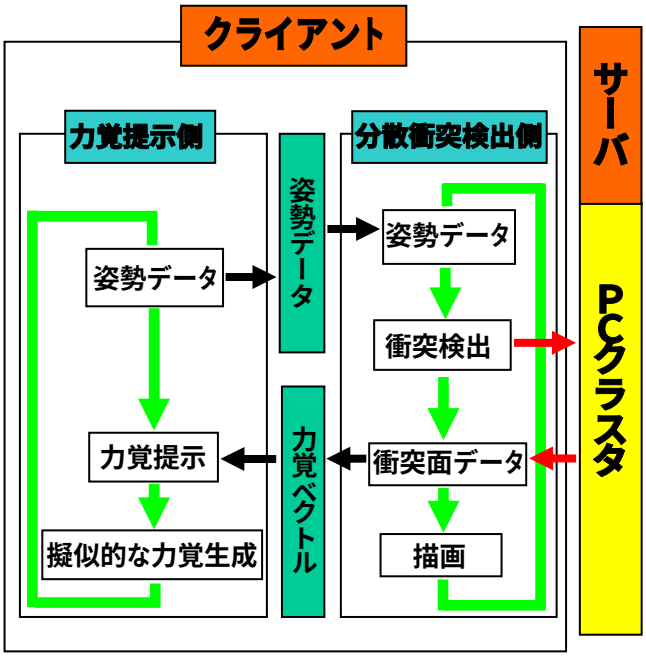


図 4．ソフトウェア構成

本論文で提案するバーチャルタッチングシステムは，図 1 に示すようなクライアント・サーバモデルで構成する．莫大な計算コストをリアルタイムで扱うための分散処理には，複数台の PC をギガビットネットワークで連結したクラスタシステムを用いる．さらに 1 台のクライアント PC を同じギガビットネットワークに接続する．クライアント PC には力覚を提示するためのハプティックデバイス（PHANToM Omni）を接続しておく．なお本研究では，図 2 にあるように 8 台の計算機を連結した PC クラスタを用いる．各計算機は，CPU Pentium4 3.0GHz，メモリ 1.0GByte であり，ギガビットネットワークで連結されている．本研究では，図 3 のように Hand モデルの指先を PHANToM のペン先とリンクさせて，PHANToM のペン先の動きと仮想空間内での Hand モデルが同期して動くシステムを構築する．オペレーターが PHANToM を操作し，Hand モデルを動かしているとき，クライアントとサーバが連携しながら Hand モデルと対象物との衝突検出を行う．

3.2 ソフトウェア構成

PHANToM Omni は入出力制御更新レートが 1KHz と高速処理を行うことができる．本研究では，力覚提示のレスポンスを保つために，スレッドプログラミングにより複数のスレッドを生成し，処理ループを分けたソフトウェア構成とする．クライアント PC の中で力覚提示側と分散衝突検出側の 2 つのスレッドを立ち上げ，衝突検出側は独立した計算処理を行う構造にする．図 4 はクライアント内でのスレッド処理についての概要図である．各スレッドにおける処理については 4.2 節で詳しく述べる．

3.3 階層化境界球群を用いた衝突検出法

本研究では，リアルタイムシステム構築のために，藤原らが提案した階層化境界球群を用いた衝突検出法[5]を用いる．

階層化境界球群を用いた衝突検出法は，Octreeを拡張した概念を用いて，最下層の球に含まれる面の数が，ほぼ一定となる階層化境界球群を生成する．例えば，3 次元空間内に物体Aと物体Bが存在するとき，図 5 のような物体Aを覆うｎ階層の球群と，物体Bを覆うｍ階層の球群を生成する．このような階層化境界球群を生成し，ブロックサイクリックマッピング[12]により分散化したとき，各計算機のロードバランスを保ちながら衝突を検出することができる．また，Octreeの部分空間を最適化することによって，冗長な干渉計算を行わずに衝突を検出することができるので，計算コストを削減することができる．図 5 に示すように，衝突を検出するには，まず，物体Aの 1 階層目の球群と物体Bの 1 階層目の球群を総当りで干渉チェックする．図 6 の(a)では緑線のc1 が 1 階層目，青線のb1，b2，b3 が 2 階層目，赤線のa1～a9 が 3 階層目の領域を示す．次に，1 階層目で干渉している球の子球群である，2 階層目の球群同士を総当りで干渉チェックする．さらに，干渉している球の子球群を総当りで干渉チェックしていき，物体Aのｎ階層目の球群と，物体Bのｍ階層目の球群まで衝突部分を絞り込む．最後に，干渉してい

る物体Aのｎ階層目の球群に含まれている面と，物体Bのm階層目の球群に含まれている面との，干渉計算で正確な衝突面を求める．しかし，この方法では計算コストが非常に高くなる．そこで藤原らの手法では，衝突に関わる計算時間を短縮するためにPCクラスタを利用する．階層化境界球群を用いた衝突検出法を，複数の計算機に分散して処理することで，計算機１台あたりの計算コストを削減し，高速化することができる．階層化境界球群をPCクラスタへ分散化する手法については，3.4 節で説明する．

3.4 分散化手法概要

ブロックサイクリックマッピングという分散化手法を用いることで図５にある物体Aのn階層目の球群を各計算機に，均等かつ離散的に割り当てることができる．ブロックサイクリックマッピングでは，形状の存在する部分空間を複数の計算機へ割り当てるため，計算負荷が特定の計算機へ偏らない．

すべての衝突を検出するには，各計算機が，割り当てられた球群に対して衝突面を検出し，最後にクライアントがすべての計算機の結果を統合する．

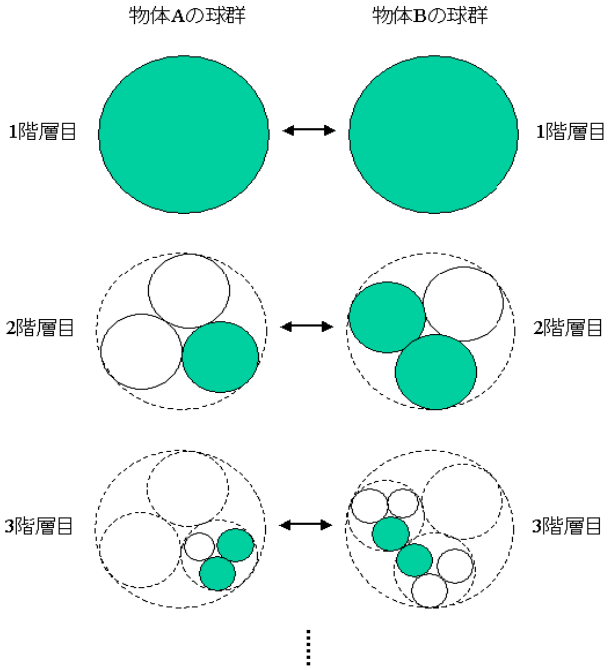
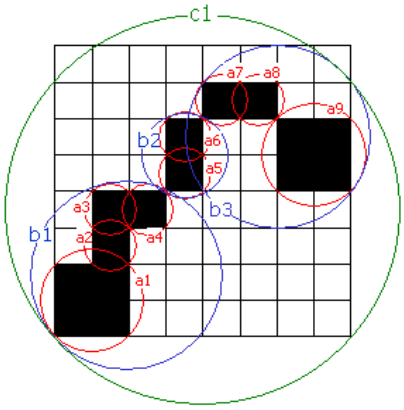


図５．衝突検出の手順

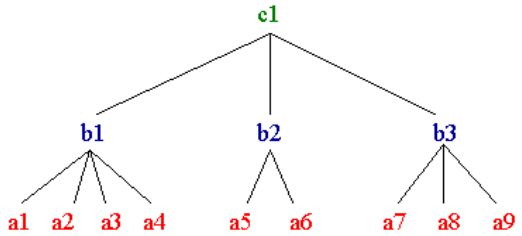
以下に 3．3 節で述べた衝突検出法の分散化に関する手順を説明する．この手順は，図４に示す，クライアント内の２つスレッド処理のうちの分散衝突検出側で行われる処理を表している．なお本研究では，すべての計算機が同一の物体データを所持しているものとする．

1. クライアントはすべてのサーバに対して物体データの読み込みを命令する．命令を受けたサーバは新たにスレッドを生成し，処理を開始したことをクライアントに通知する．
2. クライアントおよびサーバは物体データを読み込む．
3. クライアントおよびサーバはそれぞれ形状表面に着目したOctree の生成，部分空間の最適化，階層化境界球群を生成する．
4. 階層化境界球群の最下層の球群を，クライアント，およびサーバに，分散して割り当てる．
5. クライアントおよびサーバは割り当てられた球群に対して衝突を検出する．
6. サーバは衝突検出の結果を保存し，スレッドを破棄する．
7. クライアントはすべてのサーバに対して衝突検出結果を要求する．サーバはクライアントの要求通り衝突検出結果をクライアントに渡す．
8. クライアントはサーバの結果とクライアントの結果を統合する．
9. クライアントは衝突検出に基づいた力覚ベクトルを計算し，衝突箇所を表示する．
10. ハプティックデバイスのプロキシの姿勢を取得し，手のモデルの姿勢を座標変換する．同様の変換行列をサーバに渡し，座標変換を命令する．命令を受けたサーバは新たにスレッドを生成し，処理が開始したことをクライアントに通知する．
11. クライアント，およびサーバは，物体の座標と，球の中心の座標を変換する．

12. 手順 5 に戻り，衝突を検出する．



(a)



(b)

図 6．階層化球群の生成例

4．力覚提示手法とスレッド処理

4.1 力覚ベクトルの生成

一般に，VR 環境で対象物に接触するためには，接触している部分を高速に特定し，接触部分にかかる力の大きさと方向を算出する必要がある．そこで，まず本研究では，Hand モデルと対象物の接触部分を 3.3 節と 3.4 節で述べた手順で獲得する．その後，接触部分近傍の幾何情報に従って，PHANToM Omni で力覚を提示するための力覚ベクトルを生成する．

本研究では，Hand モデルの動作速度に比例する減衰力を考慮した，自然な力覚提示を行うため，図 7 に示すような線形バネと粘性減衰を並列に連結したバネダンパモデル[13]で力覚ベクトルを生成する．

バネダンパモデルを利用した力覚ベクトル $\mathbf{F}$ は，式(1)で生成する．

$$\mathbf{F} = -K_s \mathbf{X} - K_d \mathbf{v} \qquad (1)$$

ただし $\mathbf{X}$ は，対象物の変位ベクトルである．また $\mathbf{v}$ は，Hand モデルの速度ベクトル，K_s は剛性，K_d はダンパ定数である．本研究では，経験的に K_d =0.01 を設定することで安定して力覚ベクトルを発生しながら，剛体を表現できる．

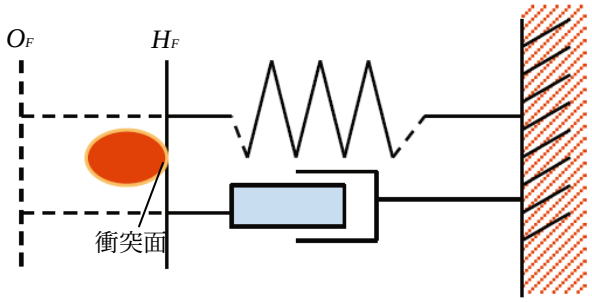
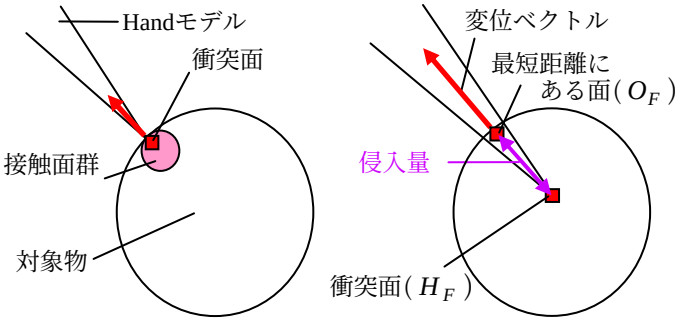


図7．バネダンパモデル

図 8 は，変位ベクトル $\mathbf{X}$ を決定するための模式図である．図 8(a)ピンクの円は Hand モデルと対象物の接触面群を示す．Hand モデルに属する全ての接触面群の頂点と，対象物に属する全ての接触面群の重心間の距離を総当りで計算し，最短距離にある Hand モデルの衝突面 H_F と対象物モデルの面 O_F を得る．図 8(b)に示すように H_F と O_F の距離を侵入量として，変位ベクトルの大きさとする．図 8 では，最短距離を表す面を赤色の四角いマーカーで示した．本手法では面 O_F の単位法線ベクトルを変位ベクトルの方向とする．



(a) (b)

図 8.変位ベクトルの決定

バネダンパモデルでは，物体の剛性は定数 K_s で決まるので，固い物体を表現するためには K_s を大きくしなければならない．K_s が大きくなると，少しの侵入量の変化で力の大きさが大きく変化する．

例えば図 9 は，速度ベクトル $\mathbf{v}$，剛性 K_s を 4.0 に固定したときの，侵入量と力覚ベクトル $\mathbf{F}$ の関係を示したグラフである．横軸は侵入量，縦軸は力覚ベクトル $\mathbf{F}$ の大きさを示している．例えば，侵入量が 0.5mm から 1.0mm に変化したとき，力の大きさが 2N から 4N に

なり，2 倍となっている．このことから，剛性を大きくすると少しの侵入量の変化によって，力覚が敏感に変化することが分かる．

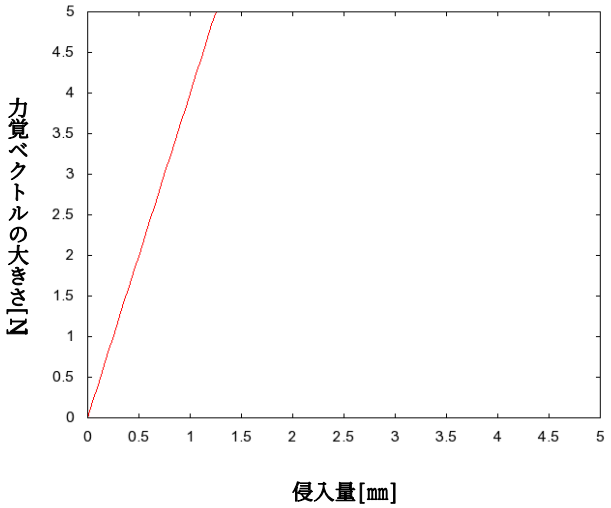


図9. 従来の力覚ベクトルの大きさと侵入量の関係

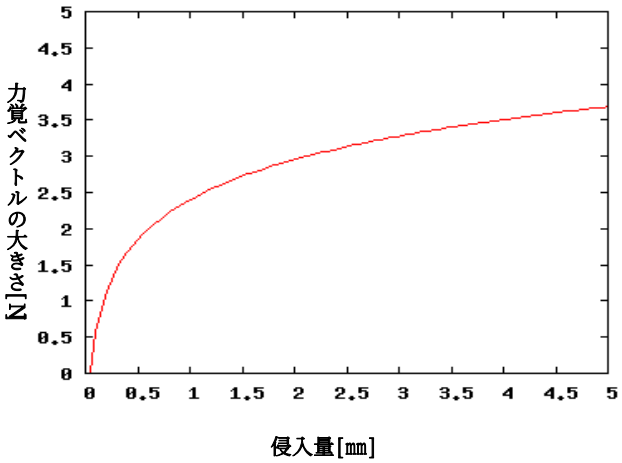


図10. 本手法の力覚ベクトルの大きさと侵入量の関係

本手法では，対象物に侵入した直後には強い力覚を提示し，その後の侵入量の変化に対する力覚の変化を緩やかにするため，バネダンパモデルを拡張し，対数関数を用いることで剛体に触れた感覚を提示する．具体的には，剛性 K_s は式(2)により算出する．

$$K_s = ((\log(P) + a) * b) \qquad (2)$$

ただし，P は侵入量を表す．またa,bは，実験による経験値である定数となっており，本研究では，a=3.0, b=0.8を設定している．

図10は，式(2)をグラフで表したものである．グラフに示されるように，本手法では侵入量が0.5㎜以下では，図9のグラフに示す力覚ベクトルに近い力が提示され，その後は力の大きさが緩やかに変化する．

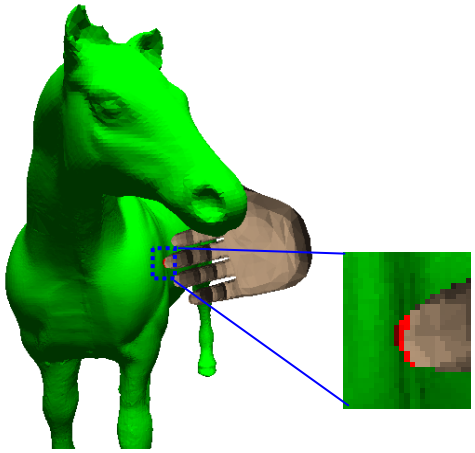


図 11． Hand モデルと馬モデルのタッチングシミュレーション例

4.2 力覚提示手法とスレッド処理

図 11 は Hand モデルと馬モデルとのタッチングシミュレーションの例である．Hand モデルと馬モデルの接触面が赤く表示されており，複数箇所で衝突が起こっていることが分かる． PHANToM Omni は入出力制御更新レートが 1KHz で動作するため，衝突検出で計算時間がかかってしまうと，衝突計算を行っている間，力覚生成が途切れてしまい，連続的かつ滑らかな力覚提示が困難になってしまうという問題点がある．

そこで，3.2 節で述べたように，クライアントが，力覚提示側と分散型衝突検出側の 2 つのスレッドを生成し，衝突検出が原因で力覚提示が途切れないようにする．すなわち，処理ループを独立させることで力覚提示の更新レートを維持し，力覚提示の途切れを防止する．3.2 節の図 4 はその構造の概要図である．力覚提示側のスレッドでは，PHANToM Omni で力覚を提示するための処理を行い，PHANToM Omni から得られる Proxy の姿勢データを，分散衝突検出側のスレッドに渡す．Proxy の姿勢データは，アフィン変換を表す 4×4 行列である．分散衝突検出側では，渡された姿勢データに基づいて，PC クラスタを利用した衝突検出を行う．そして，衝突検出によって得られた衝突面の単位法線ベクトルから力覚ベクトルを生成して，4.1 節で述べた方法で力覚提示側に渡す．力覚提示側では，式(2)基づいて力覚ベクトルを発生させる．ただし，

図4に示す姿勢データと力覚ベクトルに関する書き込みは，排他制御を行うため，書き込み中は取り出しが行えない．また，衝突検出側のスレッドの処理が力覚提示側のスレッドの処理の約25倍の時間を要するので，力覚の提示が不連続にならないようにするための，方策が必要である[14]．本研究では，衝突検出側のスレッドで，次の力覚ベクトルを書き込むまでの間に擬似的に力覚ベクトルを生成する．擬似的力覚ベクトル生成については，4.3節で説明する．

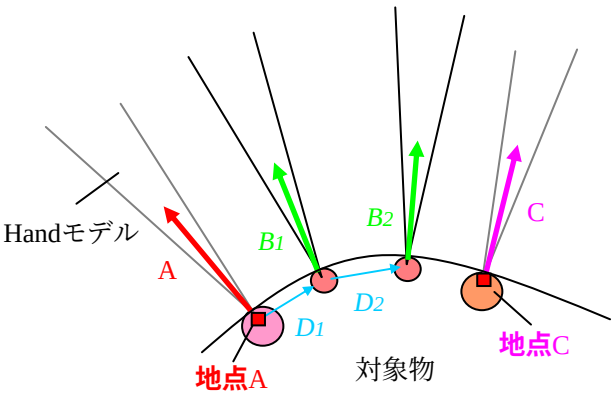


図12．擬似的な力覚の変化

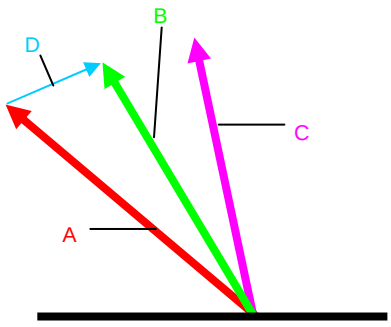


図13．擬似的な力覚ベクトル

4.3 擬似的な力覚生成

擬似的な力覚生成には，フォンシェーディングを応用したForce shading[15]や力覚ベクトルを補間する手法[16]が提案されている．本研究では，文献[16]の手法を簡略化した手法で力覚ベクトルを生成する．具体的な手法について以下に示す．まず，図12に示すように，Handモデルが地点AからCへ対象物の表面を沿って動いていると仮定する．また，地点Aでの力覚ベクトルAは，衝突検出側のスレッドで生成されたベクトルとする．Hand モデルの動きにあわせて，衝突検出側のスレッドが，次に生成する力覚ベクトルをベクトルCとした場合，地点AからCへいたる経路上では，対象物表面にあわせた力覚ベクトルを発生することができない．本手法では，力覚ベクトルAとHandモデルの移動量Dより，地点AからCへ至るまでの間の力覚ベクトルBを想定する．具体的には，Handモデルの移動量をDとすると，図13に示すようにB=A+Dにより擬似的に力覚ベクトルの方向を決定し，式(2)を用いて力覚ベクトルの大きさを決定する．これにより，図12に示すように，地点Aで力覚ベクトルAが生成されてから，地点Cにおける力覚ベクトルCを算出するまでの間に，Handモデルの移動量 $D_1, D_2, D_3, \cdots$, より擬似的な力覚ベクトル $B_1, B_2, B_3, \cdots$, が順に生成される．

5. 結果と考察

本研究では，リアルタイムバーチャルタッチングシミュレーションを目指しているので，ポリゴン数の異なる対象物において，コンタクト数[17]が変化するとき，よりリアルタイムに近づけたい．コンタクト数とは，Hand モデルと対象物の衝突面総数である．そこで，ポリゴン数やコンタクト数の変化における，最適なレスポンスを表すPCクラスタのサーバ台数を調べるために，図3(b)に示す，Hand モデル(37,248ポリゴン)と球モデル(960ポリゴン)，図11に示す馬モデル(96,966ポリゴン)，および船モデル(194,092ポリゴン)とLoop細分割[18]を用いて，船のポリゴン数が4倍になるように細分割したモデル(776, 368ポリゴン)について，分散型衝突検出法[5]に基づく局所領域の衝突検出にかかる時間計測を行う．表1は，サーバ台数とコンタクト数の関係を検証するため，PCクラスタのサーバ台数を0，1，2，4，8台と変えたときの衝突検出時間を計測した結果である．

表1．サーバ台数ごとの衝突検出にかかる時間(ms)

	ball(960)		horse(96,966)		boat(194,092)		boat(776,368)	
コンタクト数	100	1500	100	1500	100	1500	100	1500
0台	444.4	8333.3	456.6	11111.1	776.2	12500	1010.1	14285.7
1台	215.1	3846.2	234.7	9090.9	429.2	7692.3	645.2	5623.1
2台	209.2	2325.6	549.4	4545.5	364.9	4716.9	581.4	4545.4
4台	401	1449.3	299.4	2702.7	564.4	1515.1	602.4	2325.5
8台	401	2381	561.8	1408.4	598.8	1754.3	699.3	2173.9

PCクラスタによる衝突検出のレスポンスについて

は，文献[17]で述べられているように，コンタクト数が 1000 未満のときは，クラスタを構成するサーバ台数は，1，2 台が効果的であることを示されている．また，コンタクト数が 1000 以上ではサーバ台数が多いほど効果的であることが示されている．表 1 に示すように，実験結果についても，同様の傾向が表れている．

表 2．サンプルプログラムと提案手法との比較結果(FPS)

	サンプル		提案手法	
FPS	HL	GL	HL	GL
ball(960)	780.54	68.38	1000*	92.91
horse(96,966)	6.21	45.4	1000*	24.39
boat(194,092)	2.74	16.83	1000*	9.62
boat(776,368)	1.38	6.33	1000*	2.65

次に，本手法と同様な動作をする OpenHaptics Toolkit 付属のサンプルプログラムと PC クラスタのサーバ台数が 1 台のときのレスポンスを比較する．実験ではポリゴン数の異なる対象物を用いて，力覚提示のコールバックが 1 秒間に何回行われるかを表す HL FPS(Frame Per Second)と描画のコールバックが 1 秒間に何回行われるかを表す GL FPS をカウントする．表 2 に示すように，サンプルプログラムではポリゴン数が増えるにつれて HL FPS と GL FPS が減少している．これは，ハプティックライブラリを用いて力覚提示から干渉計算，描画までを 1 つのスレッド内で行っているため，ポリゴン数が多くなり衝突検出や，描画処理に時間がかかると力覚提示も必然的に遅くなってしまうためであると考えられる．また，ポリゴン数の多い対象物では，リアルタイムでの力覚提示が困難になり，20 万以上のポリゴン数を持つ対象物を扱うときには，ほとんど力覚は提示されない．一方，表 2 に示すように，提案手法では力覚提示側スレッドの処理が HL FPS に，分散衝突検出側スレッド側の処理が GL FPS に対応しており，力覚提示処理と，衝突検出及び描画処理の処理ループを分けているので，ポリゴン数が増加しても，衝突検出にかかるフレームレートが，力覚提示に深く影響を与えないことがわかる．表 2 に記載されている 1000*は，FPS 値が 1000 以上であることを表している．以上のように，本手法は，莫大なポリゴン数を持つ対象物でも力覚提示を行うことができると考えられる．

また，提案手法による力覚提示の有効性を検証するため，前述の球，馬，船，および細分割した船の各モデルに対して z 軸方向から 2 回接触したときの x 軸，y 軸，z 軸方向の力覚ベクトルの成分をグラフに示す．

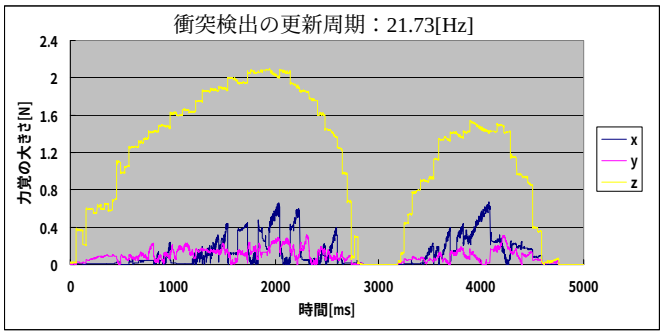


図 14.ball(960 ポリゴン)

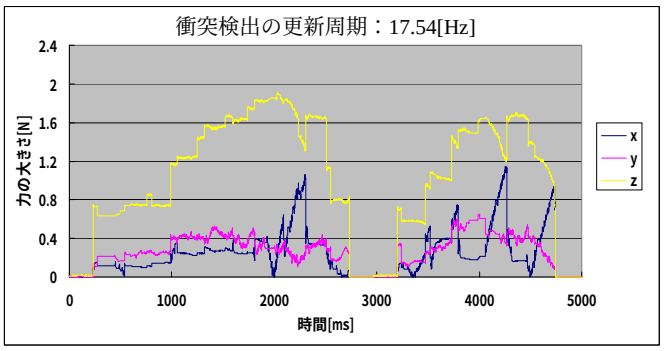


図 15．horse(96,966 ポリゴン)

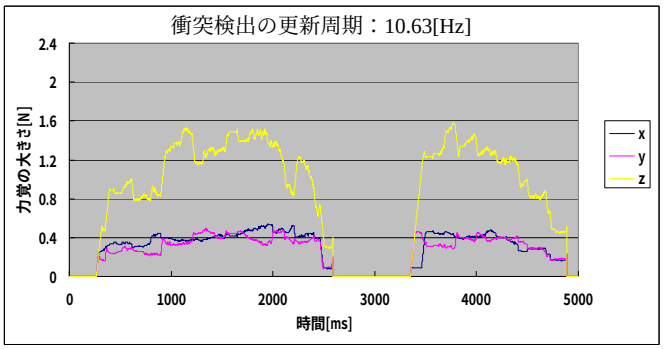


図 16.boat(194,092 ポリゴン)

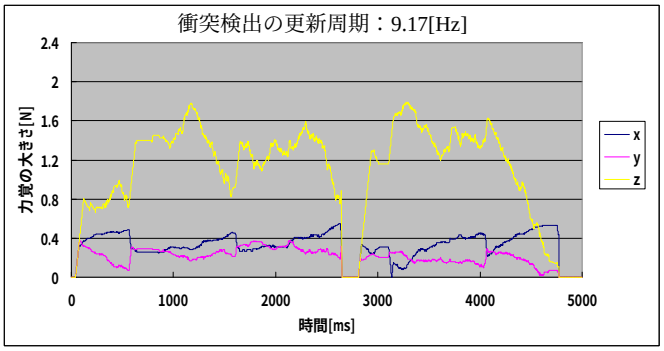


図 17.boat(776，368 ポリゴン)

なお，x 軸方向を青，y 軸方向を赤，z 軸方向を黄色で示している．なお，計算機は CPU Core™2 Quad 2.83GHz，メモリ 3.6GByte である．

図 14 を見ると分かるように，2000 ミリ秒と 4000 ミリ秒付近で，Z 軸方向に力のピークが表れている．このことから，力覚提示が意図したように行われていることがわかる．また，衝突検出の更新周期は 21.73Hz であった．図 15 から図 17 においても，同様の傾向が見られることから，ポリゴン数が多くなっても，適切な力覚提示が行われていることがわかる．図 15 から図 17 の衝突検出の更新周期は 9Hz 以上であり，情報提示として有用であることが分かる．

6. まとめ

本論文では，分散型衝突検出法に基づく局所領域の力覚提示システムについて述べた．本手法では，クライアント内での力覚提示処理と衝突検出，描画処理を別スレッドで行う．力覚を提示するための衝突検出を力覚提示処理ループから独立させることによって力覚提示のタイムラグを除き，力覚提示の途切れを防止する．また，衝突検出が間に合わないときに，力覚提示側のスレッドで擬似的に力覚を生成することによって，滑らかに力覚提示をする．本手法では，PC クラスタを使って衝突検出処理を分散することによって計算機 1 台にかかる計算時間を軽減し，全体としてのパフォーマンスを向上させることができた．これによって，OpenHaptics Toolkit 付属のサンプルプログラムでは力覚を提示することができないような莫大なポリゴン数を持つ対象物に対して，滑らかな力覚提示が可能となった．

謝辞

本研究の一部を実施していただいた，修了生の和田謙大氏に感謝する．本研究の一部は，科研費基盤研究(C)（20500880）の助成を受けたものである．

参考文献

[1] 北村善文， 竹村治雄， アフジャナンドラ， 岸野文郎，“Octreeと多面体表現を用いた三次元物体間の衝突面検出”，ロボット学会誌， Vol. 14， No. 5，pp. 121-130，September 1996.

[2] 和田謙大，今野晃市，徳山喜政，曽根順治，“分散型衝突検出手法に基づく局所領域の力覚提示システム”，第23回NICOGRAPH 論文コンテスト論文集，pp. 85-90，2007.

[3] 小堀研一， 中西大輔， “形状近似球群による高速な衝突判定の一手法”，電子情報通信学会論文誌，Vol. J85， No. 9，pp. 1455-1463，September， 2002

[4] Y. Choi ,Y. J. Kim and M. H. Kim，“Rapid Pairwise Intersection Tests Using Programmable GPUs”，The Visual Computer，Vol. 22，No.2 ，pp. 80-89，2006.

[5] 藤原慎也， 今野晃市，曽根順治， 徳山喜政,“階層化境界球群を用いた正確な衝突面検出法”， 画像電子学会誌， Vol. 35， No. 1， pp. 20-29， 2006.

[6] B. Raffin and L. Soares，“PC Clusters for Virtual Reality”，IEEE Virtual Reality Conference， 2006.

[7] W. Gropp， E. Lusk and A. Skjellum，“Using MPI: Portable Parallel Programming with the Message-Passing Interface”， Scientific and Engineering Computation Series. The MIT Press，1994.

[8] F. E. Remond Ⅲ， DCOM デプロイメントガイド，翔泳社，1998.

[9] Y. Adachi，T. Kumano and K. Ogino，“Intermediate Representation for Stiff Virtual Objects”，IEEE Virtual Reality Annual International Symposium，1995.

[10] 長谷川昌一， 石井雅博，小池康晴，佐藤誠，“動的な仮想世界の力覚提示のためのプロセス間通信”， 電子情報通信学会論文誌，Vol. j82，No. 10，pp. 1758-1756， 1999.

[11] M. Bergamasco and F. Salsedo et. al， “High Performance Haptic Device for Force Rendering in Textile Exploration”， The Visual Computer，Vol. 23，No. 4，pp. 247-256， 2007.

[12] P. パチェコ：MPI 並列プログラミング，倍風館，2001.

[13] M. O. Alhalabi and S. Horiguchi，“Effect of Haptic Properties on Human Performance in Haptic Virtual Environment”，International Conference on Virtual Systems and MultiMedia (VSMM)，Gifu，Japan. pp.202-209，2000.

[14] T. Takahashi, K. Wada, K. Konno, J. Sone and Y. Tokuyama, “A Study of Local Area Force Feedback System Based on Distributed Collision Detection”, Proc. of IWAIT2009, 2009.
[15] H.B.Morgenbesser and M.A.Srinivasan, “Force shading for haptic shape perception", Proceedings of A Dynamic Systems and Control Division, DSC-Vol. 58, pp. 407-412, 1996.
[16] 赤羽克仁, 長谷川晶一, 小池康晴, 佐藤誠, “10kHz の更新周波数を実現する高解像度ハプティックコントローラの開発”, 日本バーチャルリアリティ学会論文誌, Vol. 9, No. 3, pp. 217-226, 2004.
[17] K. Wada, K. Konno, J. Sone, and Y. Tokuyama, “An Investigation of Calculation Performance to Construct a VR System Based on Distributed Collision Detection”, Proc. of IWAIT 2007, pp. 153-158, 2007.
[18] C. T. Loop, “Smooth Subdivision Surfaces Based on Triangles”, Master's thesis, University of Utah, 1987.

高橋 哲也

2010 年3 月岩手大学大学院工学研究科情報システム工学専攻修了. 現在, 日立ソフトウェアエンジニアリング勤務.

今野 晃市

1985 年, 筑波大学第三学群情報学類卒業. (株)リコーソフトウエア研究所, ラティス・テクノロジー(株)を経て. 現在, 岩手大学工学部教授. CG,CAD,VR,遺物計測などの研究に従事. 著書に「3次元形状処理入門」がある. 博士(工学). 芸術科学会, 映像情報メディア学会, 日本情報考古学会, 情報処理学会, IEEE CS の会員.

曽根 順治

1985年, 豊橋技術科学大学大学院修士課程修了. (株)東芝, 慶應義塾大学 SFC 研究所訪問所員を経て, 現在, 東京工芸大学准教授. CAD,CGにおける自由曲面生成手法, 形状制御手法の研究に従事. 博士(工学). 情報処理学会, 映像情報メディア学会, 精密工学会会員.

徳山 喜政

1986 年, 東京大学工学部産業機械工学科修士課程修了, (株)リコーを経て, 現在, 東京工芸大学教授. G,CAD,VR, モデリング手法, ハプティックインターフェースなどの研究に従事. 博士(工学). 情報処理学会, 映像情報メディア学会, 画像電子学会. 芸術科学会会員.

原美 オサマ

1998 年上海大学情報科学修士課程修了. 2001 年, 北陸先端科学技術大学院大学情報科学研究科後期博士課程単位取得. 同年同大学, 知識科学研究科富士通寄附講座教員. 2003 年岐阜大学バーチャルシステムラボラトリー研究員, 2006 年岩手大学工学部助教現在に至る. 博士 (情報). 触覚レンダリング, ハプティック. インターフェース, ＶＲ触診訓練, レーザグラフィックスなどに関する研究に従事.