日本バーチャルリアリティ学会第 12 回大会論文集(2007 年 9 月)

CAVE 環境における動的負荷分散を用いたボリュームレンダリング

Volume Rendering Using Dynamic Load-Balancing for CAVE System

内野孝哉[1)]，小木哲朗[2)]
Takaya UCHINO and Tetsurou OGI

1）筑波大学大学院 システム情報工学研究科　コンピュータサイエンス専攻
（〒305-8577 茨城県つくば市天王台 1－1－1，uchino@gil.cs.tsukuba.ac.jp）
2）筑波大学大学院 システム情報工学研究科　学術情報メディアセンター
（〒305-8577 茨城県つくば市天王台 1－1－1，tetsu@cc.tsukuba.ac.jp）

Abstract: The volume rendering method can represent the complicated shape and inside structure of the three-dimensional object. When this method is used in the immersive projection display such as the CAVE, it is expected that the user can easily perceive the represented object intuitively. In the CAVE system, it is necessary to represent the volume data in real time, so that the user can interactively see the stereoscopic vision from his viewpoint. However, it is difficult to perform the volume rendering in the CAVE environment, because of the large rendering time due to the large scale data. In this study, the dynamic load-balanced rendering method was applied to represent the volume data in the CAVE display, and the optimal control method was discussed.

Key Words: Dynamic load-balancing, Immersive projection display, CAVE, PC cluster, Volume rendering

1. はじめに

人間の広い視野を覆う大型のスクリーンで構成される没入型ディスプレイは、立体視による可視化環境として高い臨場感の映像空間を提示することができる。例えば、CAVE[1]や CABIN[2]等の複数のスクリーンと複数のプロジェクタを組み合わせた没入型多面ディスプレイ環境では、視界を完全に覆うことができるため、より没入感の高い仮想空間を構築することができる。

一方、空間に分布する濃度や密度をボリュームデータとして表し描画する方法にボリュームレンダリングがある。自然界の複雑な形状は人工物のような幾何形状を持たないためポリゴンによる表現が困難であるが、ボリュームレンダリングでは効果的に表現することができる。ボリュームデータの三次元表示は、データの内部まで観察が可能であるため、対象に対する直感的な理解が可能となる。

このボリュームレンダリングを没入型ディスプレイ環境で行うと、利用者はデータ空間の中に入り込んだ状態で、三次元のデータを三次元のまま観察することができ、二次元のモニタで観察するよりも複雑な形状をより把握しやすくなることが期待される。

ボリュームレンダリングでは、三次元の空間を格子状に小さく分割し、区切られた１つ１つのボクセルに情報を与えることでデータを表現するが、この際三次元の空間を小さく区切るほど表現できる形状の精度があがる。しかし、精度を上げるほどデータ量は膨大になるため、膨大なデータを記憶しておくための記憶容量とそれを高速に処理できる CPU が必要となる。そのため、ボリュームデータを CAVE のようなマルチスクリーンの没入型ディスプレイで提示するときには、常に変化するユーザの視点に対して立体視映像をリアルタイムでレンダリングをする必要があるが、膨大なデータ量のためフレームレートが落ちてリアルタイム性の確保が困難になるという問題がある。

レンダリングの高速化手法として PC を複数台使ってクラスタシステムを構築してレンダリング負荷を分散させる方法がある。しかし、従来の負荷分散レンダリング手法[3][4]では、没入型ディスプレイにおけるスクリーン間の負荷の分散には対応することはできない。本研究で提案を行う動的負荷分散レンダリング手法[5]では、スクリーン間を含めた負荷のバランスを取り、レンダリング負荷を安定化させ、全体のパフォーマンスを上げることで、システムの最適化を行う。

今回の研究では、CAVE 環境におけて動的負荷分散レンダリングの制御方法に関する検討を行い、リアルタイムによるボリュームレンダリングを実現することを目指す。

2. システムの構成

本研究では映像を表示するためのディスプレイには、3面構成の CAVE 型の没入型ディスプレイシステム CS Gallery を使用した。スクリーン構成は、正面、側面、床面の3面で構成され、各スクリーンに対してそれぞれ2台ずつの DLP プロジェクタ NEC LT245J を用い、円偏光による立体視方式を用いている。また利用者の視点位置を計測するためのセンサとしては、Ascension Flock of Birds を使用している。

映像を生成するための計算機には、7 台の Linux PC(HP XW6200, Intel Xeon 3.4GHz, NVIDIA FX3400)を用いたクラスタシステムを構成し、分散レンダリングされた映像の重畳のためにコンポジタボードの ORAD DVG を導入している。1 台の PC はコントロール用計算機として用い、残りの 6 台の PC は分散レンダリング用計算機として使用している。コントロール PC は、センサデータの取り込み、各ノードに対する分散レンダリングの動的な制御に使用する。6 台の描画用クラスタＰＣは 3 台を右目用に使い、残りの 3 台を左目用の映像のレンダリングに使用している。映像は、3 台目ごとに重畳した結果を出力し、スキャンコンバータで各スクリーンの提示領域の映像を切り出し各プロジェクタに送出する。

重畳映像は、正面、側面、床面の映像を合わせた１つの大きなウインドウを開き、画面を縦と横に区切って 4 分割した領域にそれぞれ正面、側面、床面の映像を描き込む。分散レンダリングは、この正面、側面、床面の分割とは別に全体の画面を任意の領域に分け、任意台数の PC (本システムでは 3 台の PC)によって分散処理によるレンダリングを行う。合成画面のうち右下の 1/4 の領域は未使用になるため、レンダリングの負荷がかからないように黒色で塗りつぶした。合成した 1 画面の解像度は 1600X1200 とし、これ を 4 つに切り分けた各スクリーン映像の解像度は 800X600 とした。

図 1 はシステム全体の構成と分散レンダリングの処理の流れを示したものである。

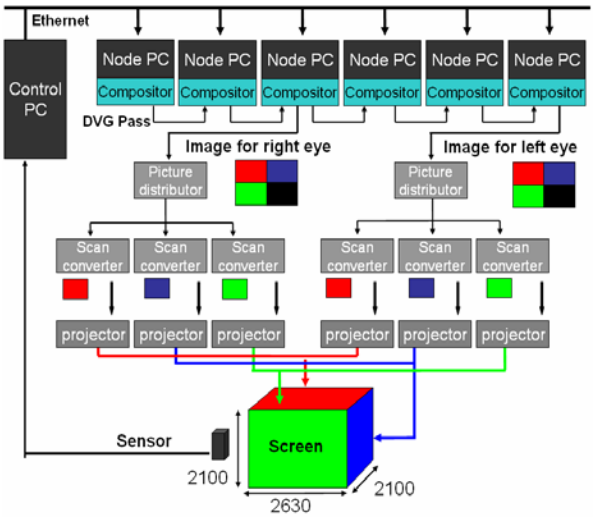

図 1　CS Gallery システムの構成

3. 動的負荷分散レンダリング

3.1 描画方法

この動的負荷分散機能は、片眼の映像にそれぞれ 3 台ずつの PC を使っているため、正面・側面・床面のスクリーン領域の分割とは別に、全体の映像空間を三分割したレンダリング領域を各 PC に割り当てている。従って各 PC は一定の視体積に対する映像をレンダリングするのではなく、スクリーンの一部分あるいは正面・側面・床面の複数のスクリーンをまたがる視体積を設定して映像をレンダリングする。動的負荷分散レンダリングの制御方法は映像空間を三分割する分割境界線の位置を左右に移動することで各 PC のレンダリング領域を変更する方法を行った(図 2)。

負荷の変動に対して動的な負荷分散を実現するためには、画面を分割する各領域の幅を各 PC のレンダリング負荷の変化に従って変更する必要がある。このシステムでは、各ノード PC はレンダリング速度を常時計測しコントロール PC にデータを送る。コントロール PC は、レンダリンタイムが遅かったノード PC に対して、担当するレンダリング領域の幅を小さくするように分割境界線の位置を移動させる。この動的負荷分散レンダリングを行うことで各 PC の描画負荷のバランスを取り、全体のパフォーマンスを向上させる。

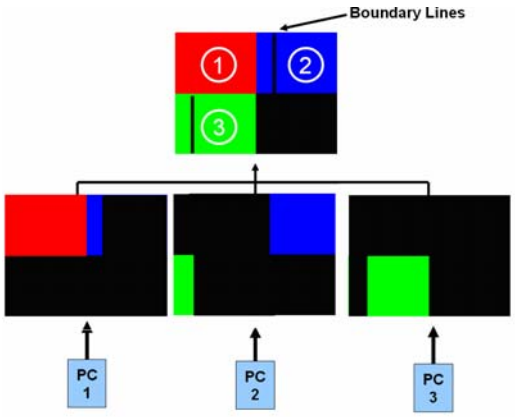

図 2　動的負荷分散レンダリング

3.2 動的負荷分散の制御

各 PC が描画する分割領域の制御方法は､3 面を横一列に並べた重畳画面を 3 分割する 2 本の境界線を横方向に移動させることで行った。基本的にはノード 1 とノード 2 の間でレンダリングタイムが大きい方に最初の分割境界線を移動させ、ノード 2 とノード 3 の間でレンダリングタイムが大きい方に次の分割境界線を移動させる。各スクリーン内のレンダリングタイムの分布を考えて最適化するように分割境界線を決める方法も考えられるが、分割境界線の最適位置を決める計算負荷が逆に大きくなってしまうことを避けるため、ここではスクリーン内の速度分布までは考慮せずに、各 PC の現在のレンダリングタイムから次の描画領域を決定する方法を取っている。

制御方法としては、まず各ノード PC から送られてきたレンダリングタイム情報をコントロール PC に集め、隣接

する PC 間でレンダリングタイムが大きい方に境界線を一定量移動させる方法が考えられる。一般にバーチャルリアリティのアプリケーションでは、利用者の視点移動や仮想物体の運動によって負荷の変動が生じるが、この場合負荷は徐々に変化するため、急激に偏ることは稀である。そのためここで用いる負荷分散制御の方法は簡易な方法でありながら十分に制御可能と考えられる。例えば、分割境界線の移動量が 80 ピクセルのとき、境界線は画面の端から端まで（2400 ピクセル、800 ピクセル×3 画面）を 30 回の更新で移動できる。1 つの PC だけに負荷が集中するような急な負荷の変動がある場合でも、このときフレームレートが 30Hz 以上でていれば、スムーズに負荷のバランスを取ることができる。この方法では、一回の制御における移動量を大きくすることで速い移動を行うことができるが、移動量が大きすぎるとパフォーマンスが安定しないという状況も生じてしまう。

また、パフォーマンスが安定する前に急激な負荷の偏りが連続して起こるような場合には、分割境界線を固定値で移動させる方法では対応できない。特に今回用いるボリュームレンダリングでは、レンダリング負荷が高く描画フレームレートが低いため、負荷のバランスが取れる前に急激な負荷変動が発生する場合がある。そのため、各 PC 間で最適な描画領域を求め、境界線の目標移動位置までいっきに移動させる方法も考えられる。ここでは、現在のレンダリング周波数(Hz)の比から、これに比例させて次の描画領域の幅を決定する方法を行った。例えば、現在のレンダリング周波数の比が 1：m：n であった場合、それぞれの描画領域は 2400×1÷(1＋m＋n)、2400×m÷(1＋m＋n)、2400×n÷(1＋m＋n)ピクセルの領域幅となる。こうすることにより、高負荷の PC が変動した場合には、レンダリングタイムに合わせて移動幅をより大きく取ることができるため、急激な負荷の偏りが連続して起こったときの対策となる。しかしこの方法でもスクリーン内の不可分布は考慮していないため最適値ではない。そのため、大まかな移動を行った後に少しずつの位置調整を行うことで最適分割へ収束させるような方法を取ることが必要であり、実用的には前述の一定量の移動と組み合わせた方法を取ることが考えられる。

4. 評価実験

提案する動的負荷分散の制御手法を用いることでボリュームレンダリングにおけるリアルタイム性能を評価するために、次のようなレンダリング速度の計測実験を行った。

4.1 実験内容

映像としては 256x256x256 のボリュームデータ[6]をボリュームレンダリングし(図 3)、分割境界線の移動方法の違いによるパフォーマンスへの影響を調べた。映像を描画する際、視点を図 4 の①→②→③→④→⑤→⑥→①の順に少しずつ移動して一周させて、それぞれの位置からオブジェクトを観測した映像を描画した。

この実験では 4 通りの制御方法によるデータを取って比較した。一つ目は、動的負荷分散を適用していない場合のデータを取った。二つ目は、動的負荷分散制御において各 PC の描画速度の比によって分割境界線の移動位置を決定する方法。三つ目は、動的負荷分散制御において、分割境界線の移動量は固定値とする方法。四つ目は、二つ目の方法と三つ目の方法を融合した方法で、負荷の変動が小さい場合は分割境界線を固定量だけ移動させ、急激に負荷が変動した場合には各 PC の描画速度の比によって分割境界線の移動位置を決定する方法である。ここではレンダリング周波数比が 3 以上の場合に、描画速度比による境界回線の決定方法を取るようにとした。

図 3 没入型ディスプレイへの提示

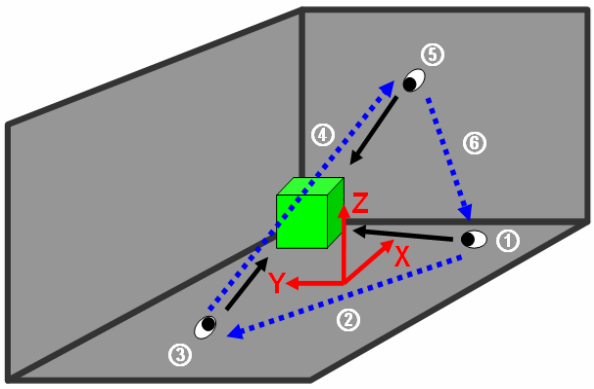

図 4 視点位置の移動

4.2 実験結果

評価実験の結果は図 5 の通りである。

①動的負荷分散レンダリングを適用しないときは、大きくフレームレートが落ちる場合があり、平均フレームレートも低かった。

②分割境界線の移動位置を各 PC の描画速度の比によって決定する場合は、動的負荷分散レンダリングを適用しないときよりフレームレートは大きいが、平均フレームレートは大きく変動していた。

③動的負荷分散における分割境界線の移動量が固定値の場合は、全体のパフォーマンスは安定しているが、負荷のバランスが取れるのに時間がかかり、負荷の大きな変動

には対応できない。

④通常は分割境界線を固定量だけ移動させ、急激に負荷が変動した場合には各 PC の描画速度の比によって分割境界線の位置を決定する場合は、負荷の大きな変動にもある程度対応でき、全体のパフォーマンスは安定しているのが分かる。

動的負荷分散レンダリングを適用しないとき場合のフレームレートの平均値は 5.22Hz であったが、提案手法の融合方法を適用した場合のフレームレートの平均値は 11.68Hz だった。フレームレートの平均値で比較すると、動的負荷分散レンダリングを適用するとフレームレートは 2.23 倍となり、片目の映像に対して 3 台の描画 PC を用いているので、PC の台数分の効果はあると言える。

しかし、動的負荷分散を適用させた場合でも、パフォーマンスは小さく変動をしていて完全には安定していない。分割境界線を固定量だけ移動させる場合の固定量に関しては、コンテンツごとの映像の変化量に応じて最適な値を決定させなければならない。このことからも、制御方法（分割境界線の移動方法）によって全体のパフォーマンスが大きく変化することが分かり、最適な制御方法を確立するためには改良の余地が残っている。

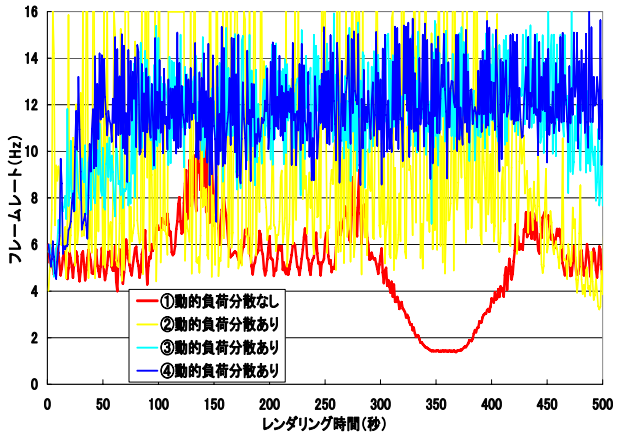

図 5 評価実験のフレームレート

5. まとめ

本研究では、没入型ディスプレイ環境におけるボリュームレンダリングを行うために動的負荷分散レンダリングの適用を行った。それにより、計算機のレンダリング能力をより効率的に安定して利用できることを示すことができ、没入型ディスプレイでボリュームレンダリングをリアルタイムに実行することが可能になった。

しかし、動的負荷分散の制御方法によって大きくパフォーマンスが変更してしまうという問題があり、最適な制御方法を確立する必要がある。今後の課題は様々なシーンにおいて効率よくパフォーマンスが発揮されるような分散制御方法を確立することである。また、今回用いた動的負荷分散機能はレンダリング負荷を分散させることはできるが、アプリケーションにおける計算負荷やメモリ容量の分散を含めたアプリケーションの効率化を考えることも必要である。

謝辞

本研究は情報通信研究機構の民間基盤技術研究促進制度における「テレ・イマーシブ・カンファレンス・システムに関する研究」の一部として行った。

参考文献

[1] C. Cruz-Neira, D.J. Sandin, T.A. DeFanti: “Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE,” Proceedings of SIGGRAPH’93, pp. 135-142, 1993.

[2] 小木哲朗：PC-CABIN による共有型没入仮想環境の構築，第 32 回可視化情報シンポジウム講演論文集，可視化情報 Vol.24，Suppl. No.1，pp.305-308，2004.

[3] 村木 茂，鈴木靖子，藤代一成，ボリュームグラフィックス(VG)クラスタによる 3D LIC レンダリングの並列化，情報処理学会研究報告 CAD 108-12, August 2002

[4] Y. Watashiba, J. Nonaka, N. Sakamoto, Y. Ebara, K. Koyamada and M. Kanazawa, "A Streaming-based Technique for Volume Rendering of Large Datasetes," Proc. CGIM 2003, The 6th IASTED International Conference on Computers, Graphics and Imaging, Hawaii, 2003

[5] 小木哲朗、内野孝哉：動的負荷分散レンダリングを用いた CAVE システム、日本バーチャルリアリティ学会論文誌 Vol.11, No.3, pp.403-410, 2006

[6] http://www.volvis.org/