C-07　実時間インタラクティブシミュレーションのための並列ボリュームレンダリング環境

Parallel Volume Rendering Environment for Interactive Real-Time Simulation

生雲 公啓*, 高山 征大*, 丸山 悠樹*, 津邑 公暁†
五島 正裕*, 森 眞一郎*, 中島 康彦†, 富田 眞治*,
M.Ikumo, M.Takayama, Y. Maruyama, T.Tsumura,
M.Goshima, S.Mori, Y.Nakashima, and S.Tomita

1　背景

近年の計算機性能の急速な向上に伴い, 大規模かつ高精度な数値シュミレーションへの期待が高まっている. 中でもリアルタイムな数値シュミレーションの可視化技術は, 大規模な 3 次元データを必要とする医療などの分野 [1] において, 高度な技術が要求されている分野である.

これまでも, PC クラスタ等の並列計算機環境を利用した, シュミレーションとその実時間可視化のためのシステムが開発されてきたが, 次世代のシミュレーション技術として, 従来の実験の代替手段となりうる 「仮想実験型/仮想体験型のシミュレーション環境」の構築が望まれている. ここでは, オペレータによるシミュレーション対象へのインタラクティブな操作 (ステアリング) に対応して実時間でシミュレーションを行うとともに, 即刻その結果を視覚その他の手段により提示することが求められている.

我々は , このような実時間インタラクティブシミュレーション環境の実現にむけて, 個人あるいは小規模な組織単位で占有利用可能な PC クラスタを用いて, インタラクティブな数値シミュレーション及びその可視化[1]を実時間並列処理する環境について研究を行っている .

本稿では, 現在我々が検討を行っている実時間インタラクティブシミュレーション環境の構想と, そこにおけるシミュレーション結果の可視化を支援する並列ボリュームレンダリング環境について述べる.

*京都大学大学院情報学研究科
†京都大学大学院経済学研究科

[1] 4096^3の 8bit ボリュームデータを SHD 規格相当のスクリーン (2048^2) に秒間 30 枚のフレームレートで出力する可視化システムが当面の目標である.

2　実時間インタラクティブシュミレーション環境

実時間インタラクティブシミュレーション環境の構築に際し, 計算機システムとして検討すべき課題は, シミュレーションの高速化のための 「スループット」, ならびにユーザの操作性向上のための 「レスポンス」といった, 両立が困難な要求への対応である. さらに, 高機能な入出力デバイスを活用したユーザインタフェースを実現するうえでは, UI に利用可能な計算機自体 (特に, OS) に制約が存在することもあり得るため, システム全体で見た場合に異機種混在型の並列計算機環境への対応も必須である.

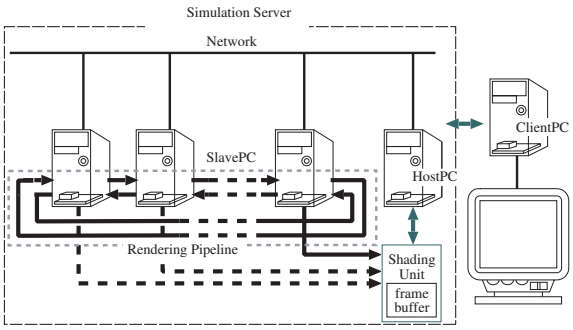

図 1: 実時間インタラクティブシミュレーション環境

そこで我々は, 計算負荷の高いシミュレーションと多次元データの可視化までを PC クラスタベースの計算サーバで実装し, シミュレーションのステアリングや結果の呈示を行うユーザインタフェース端末としてのクライアント計算機との間の通信を PVM を用いて実装することで, 全体として分散メモリ型の仮想並列計算機を構築することを検討している. これにより, 計算サーバを煩雑な入出力処理から解放するとともに, 柔軟な UI 環境の実現を目指す.

3　並列ボリュームレンダリング環境

大規模シミュレーション結果 / 過程の可視化においては, 並列計算機の各ノードで生成された膨大なデータを, 一旦収集した後に可視化するといった従来の可視化手順で実時間性の確保が不可能である. したがって, 実時間可視化を実現する上では, 生成されたデータに対して, その場で可視化に必要な処理を行い, 最低限の情報のみを全ノードから収集 / 処理してユーザに呈示する仕組みが必須である. 以下では, 3 次元データの可視化手段の 1 つであるボリュームレンダリング処理を対象として可視化システムの実現方法について検討する.

ボリュームレンダリングでは , シミュレーションにより得られた 3 次元空間上の数値データを, 色 C と透明度 t に対応づけて可視化することで, 3 次元空間内部のデータの分布状況を可視化する. 具体的には, 視線上のボクセルの値を視点に近い順に vo,v1,v2,... とするとピクセル値は次の式 (畳み込み演算) で計算される .

$$C_k = \sum_{i=0}^{k} (1 - t(v_i)) \cdot c(v_i) \cdot \prod_{j=0}^{i-1} t(v_i) \qquad (1)$$

$$T_k = \prod_{i=0}^{k} t(v_i) \qquad (2)$$

ここで $c(v_i), t(v_i)$ はそれぞれボクセル値 v_iを , 色 , 透明度に変換して値であることを示している . さらに , 式 (1) , (2) は以下のような漸化式で表わされる .

$$C_k = C_{k-1} + (1 - t(v_i)) \cdot c(v_k) \cdot T_{k-1} \qquad (3)$$

$$T_k = t(v_k) \cdot T_{k-1} \qquad (4)$$

この畳み込み演算は, 演算区間をいくつかの部分区間に分割し, それぞれの区間に対する計算結果に対して, 再度畳み込み演算を行うことが可能であるという性質がある. そこで, 1) シミュレーションサーバの各ノードが, そのノード内で生成されたデータに対する部分 3 次元空間 (以下 サブボリュームと呼ぶ) に対して畳み込み演算を行い, 2) そこで得られた画像と画素毎の透明度を, 視点からの距離の順番に従って合成 (composition) する, という手法で並列処理が可能である.

このようなボリュームレンダリング処理の実装に際しては, 可視化処理に対してどれだけの投資が可能かに応じて, 可視化用のハードウェア ・アクセラレータを用いるか否か, また, ハードウェア支援を行う場合に, テクスチャベースの汎用グラフィックスカードを用いるか, あるいは, ボリュームレンダリング専用のハードウェアを実装するかの選択が可能である[2] .

[2]PC クラスタシステムを構築する際に, ネットワークシステムにどれだけ投資するかの選択に匹敵する.

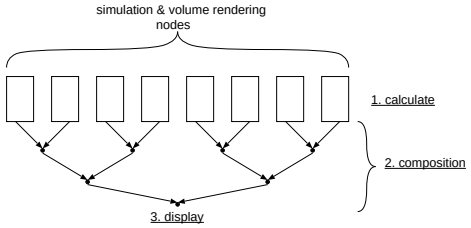

図 2: 並列ボリュームレンダリング処理の概略

我々は, このいずれのケース対しても対応可能な可視化支援環境の構築を目指している. 以下, これら 3 つのケースについて述べる.

3.1　ソフトウェアによる並列ボリュームレンダリング

この方法では, シミュレーションに用いたプロセッサを可視化処理にも利用するため, 可視化のための新たな投資が必要ないという利点があるが, 可視化処理を行っている間は, シミュレーションが止ってしまうという欠点がある.

各ノードで生成したサブボリュームにたいする中間画像データ (RGBA) ができると, これをもとに最終合成画像を生成する (composition). 多くの場合, この合成処理は各ノードが保持するサブボリューム間の隣接関係に基づく 2^N進木構造の合成ネットワークを構成してパイプライン的に処理を行う.

各ノードでの可視化処理において, 視点, スクリーン, サブボリュームの相互の位置関係を考慮したキャッシュ・ブロッキング手法を用いた最適化, ストリーミング SIMD 命令等を活用した最適化, キャッシュバイパスやプリフェッチ等のメモリ最適化, さらには, Shear-Warp, 早期視線終端 (ERT) 等のアルゴリズムレベルの最適化を行うことで専用のグラフィックスカードにせまる性能を出すことも不可能ではない.

ソフトウェアによる手法では, レンダリング処理時間が大きくなる傾向があるが, その反面, 非均質構造格子への対応等の柔軟な処理が可能である. ただし, この場合は可視化パラメータに依存した負荷不均衡を解消するための動的負荷分散処理等が必要になる [2].

3.2　汎用 3 次元グラフィックスカードを用いた並列ボリュームレンダリング処理

ソフトウェアによる並列ボリュームレンダリングにおいて, 計算サーバの各ノードのプロセッサが担当していたサブボリュームに対するレンダリング処理を, 汎用の 3 次元グラフィックスカード (3DCG カード) にオフロー

ドし, 3DCG カードのハードウェアアクセラレーション機能を活用して高速化を図る方法である.

ここでは, ボリューム空間を座標軸に垂直なスライスの重ね合わせで表現し, それらをテクスチャとしてポリゴンにマッピングしたものをα ブレンディングすることによってボリュームレンダリング処理を行う [3].

3DCG カードがレンダリング後, 3.1章と同様に個々のレンダリング結果をネットワークを介して合成する. 一般に 3DCG カード自信は通信機能を持たないため, 合成のためのノード間通信においては, 計算サーバの CPU を介した処理が必要となる.

現在, 汎用 3DCG カードのグラフィックスメモリサイズは高々128MB 程度であり, サブボリュームサイズよりもはるかに小さいということが考えられる. また, テクスチャサイズの上限がボリュームスライスのサイズより小さいこともあり得る. このような場合, ノード内のレンダリング処理においてサブボリュームをサブサブボリュームに分割し, それに対してレンダリングした後に結果を合成して, サブボリュームに対するレンダリング結果を得るといった処理が必要となる.

現時点ではまだ十分な考察が行えていないが, Pentium4 2.2GHz, 主記憶 512MB(PC800), 3DCG カード GeForce4 Ti 4600 (Graphics Memory: 128MB) を用いて, $128 \times 128 \times 69$ のボリュームデータに対して, 512^2の画像を生成させたところ, データ転送時間を除くボリュームレンダリング処理に 30.01msec を要することが分かった. また, データ転送に関しては, テクスチャメモリへの書き込み (`glTex3dImage()`) に要する時間は 1MB あたり 1.84 msec と AGP4X の最大転送性能に近い値を示しているのに対し, フレームバッファからの読み出し (`glReadPixels()`) や書き込み (`glDrawPixels()`) に要する時間は, それぞれ, 1MB あたり 6.16msec ならびに 82.02 msec と低速であることが分かった.

3.3 専用ハードウェアによる可視化

3.3.1 システム構成

専用ハードウェアによる可視化システム [4] は, 我々が既に開発した ReVolver/C40[6, 7] で採用したアーキテクチャをベースにしたもので, N × N × L のサブボリューム単位で並列化し, Ray Casting アルゴリズムを用いて, 1 ピクセル分のピクセル値計算をサブボリューム単位でパイプライン処理することで目標とする描画速度を得る. 可視化システムの構成としては 128 ノード構成の PC クラスタに, ボリュームレンダリング向けアクセラレータ (VisA:Visualization Accelerator)[5] を装備し, VisA 間を双方向高速リンクで接続する. 計算結果は VisA 間リンクと同一規格のケーブルによりフレームバッファに送られ, ディスプレイへ表示される. 生成された 2 次元画像に対して, さらに高度な後処理が必要な場合は, ホスト PC に送り処理を行う.

ReVolver/C40 では, 視線生成, ピクセル値計算, シェーディングの 3 ステージでそれぞれ専用のハードウェアで構成したが, VisA ではピクセル値計算のみを重点的に専用ハードウェア化し, その他のステージで必要であった処理は各 PC の CPU や汎用グラフィックスカードを用いて実行する. 以下, VisA の主要構成要素について簡単に説明を行う.

- ピクセル値計算パイプライン

32 個のピクセル値計算ユニット (PCU) をパイプラインに接続して構成する. 表示に必要な RGB は 8bit であるが, 誤差伝播の影響を軽減するため 16bit 固定小数点として色と透明度の演算を行う. パイプライン周波数は画面サイズとフレームレートから 128MHz となる. この部分の詳細は後で述べる.

- Look Up Table (C& α LUT)

RGB 各 8bit の色情報と 8bit の透過率を保持する 256 エントリのルックアップテーブルである.

- ボリュームメモリ

ボリュームデータを格納するための容量 2GB のメモリで, $4000^2 \times 32$ のサブボリュームを 4 セットまで格納可能とする.

- プリフェッチ機構

ボリュームメモリはパイプライン構成している全てのピクセル値計算ユニットから同時にアクセスされるため, 所望のパイプライン周波数を実現するために, LUT とボリュームメモリの間にプリフェッチバッファを装備し, ボリュームメモリへのアクセス遅延に伴うパイプラインストールを最小化する.

3.3.2 ピクセル値計算部分の設計

3.1 節の式 (3), (4) の計算を行う回路を FPGA 上に実装する. 以下実装方針について述べる.

式 (3) においては加算 1 回, 減算 1 回, 乗算 2 回, 式 (4) では乗算 1 回の計算が必要となる. C_k と T_k 一段分の計算を行うためには, R,G,B3 色分の計算が必要であることを考慮すると, 加算器 3 個, 減算器 1 個, 乗算器 7 個が必要であり, また各演算を 1 クロックで実行するとしても, 4 クロックかかる.

そこで, 回路の縮小, 及びパイプライン段数の削減を図るため, 透明度 t の代わりに, 不透明度$\alpha(= 1 - t)$ を用いると, 式 (3), (4) は式 (5), (6) のように変形される.

$$C_k = C_{k-1} + \alpha(v_i) \cdot c(v_k) \cdot T_{k-1} \tag{5}$$

$$T_k = T_{k-1} - \alpha(v_k) \cdot T_{k-1} \tag{6}$$

式 (5)，式 (6) を見ると，$\alpha(v_k) \cdot T_{k-1}$の計算が共通しているため，必要な演算器の数は，加算器 3 個，減算器 1 個，乗算器 4 つとなった．また，式 (5) の加算の計算を次の段の式 (5) の第 2 項の計算とオーバラップさせることにより 1 段の計算を 2 クロックで行うことが可能である (図 3参照)．このように回路を作成することで，32 段のパイプライン計算を行うためには 65 クロック必要である．

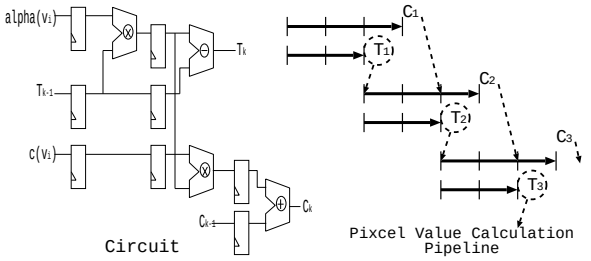

図 3: ピクセル値計算回路

この回路を Xilinx 社の FPGA デバイスである VirtexII を用いて実装する．この際，乗算器には VirtexII 特有の 18 ビット× 18 ビット構成のエンベデッド乗算器を，また LUT やプリフェッチバッファには SelectRAM を，それぞれ使用した．最終的には 32 段のピクセル値計算ユニット (PCU) を図 4のように構成してパイプライン処理を行う．そのためにまず 1 つの PCU を FPFA 上に実装

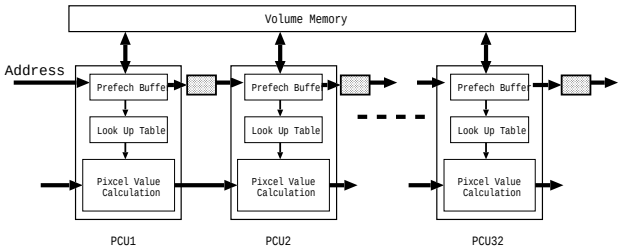

図 4: FPGA 上への実装

した．その結果，127MHz で動作することがシミュレーションで確認できた．この値は目標値である 128MHz に十分近い値であるといってよい．現在はこの PCU を 8 段，16 段となぎ，最終的な 32 段へと実装を進めている段階である．回路の規模としては，LUT やプリフェッチバッファの RAM ブロックを除いた計算部分だけでは，1 段あたり 2 万ゲート弱だるため，32 段では 60 万ゲート程度になると考えられる．分散 selectRAM を用いた実装を検討しているが，現在はブロック SelectRAM を用いて構成しており，ブロック SelectRAM に制約があるため必要以上のメモリを実装しているが，メモリを含めたゲート規模は 16 段の回路で約 240 万ゲートとなる．

今後はプリフェッチ機構を具体的に実装し，評価を行っていく．

4 まとめ

我々はインタラクティブなシミュレーションの可視化を実時間で行う環境について 3 つの方針で研究を進めている．今後はそれぞれの方針で環境の構築を目指していく．

謝辞

日頃より御討論いただく京都大学大学院情報学研究科富田研究室の諸氏に感謝します．なお， 本研究の一部は文部省科学研究費補助金 (基盤研究 (B)(2) 課題番号 13480083 ならびに特定領域研究 (C)(2) 「情報学」課題番号 13224050) による．

参考文献

[1] 山本 恭弘 他 “有限要素法を用いた心臓大動脈の触診シュミレーション”, 日本バーチャルリアリティ学会第 6 回大会論文集, 2001.

[2] L.Chen, I Fujishiro, and K Nakajima, “Parallel Performance Optimization for Large-Scale Unstructured Data Visualization for the Earth Simulator”, *Proc. of Parallel Graphics and Visualisation*, pp.133-140, 2002.

[3] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner, T. Ertl, ”Interactive Volume Rendering on Standard PC Graphics Hardware Using Multi-Textures and Multi-Stage Rasterization”, In Proc. Eurographics/SIGGRAPH Workshop on Graphics Hardware, 2000.

[4] 原瀬 史靖 他, “数値シミュレーション過程の実時間可視化を支援するハードウェア” , 可視化情報シンポジウム,2002.7.

[5] 山内, 他, “アクティブボリュームレンダリングに基づくシミュレーションステアリング,” 信学技報 CPSY2001-35, pp.1-8, 2001 年 8 月.

[6] 對馬 雄次 他, “ボリューム ・レンダリング専用並列計算機 ReVolver のアーキテクチャ”, 情報処理学会論文誌, 第 36 巻, 第 7 号, pp.1709-1718, 1995.

[7] 吉谷直樹 他, “ボリュームレンダリング専用並列計算機 ReVolver/C40 の性能評価”, 情処研報告, 99-ARC-132, pp.79-84, 1999.