

地球流体科学研究用のポータル Gfdnavi への 3次元可視化機能の組み込み

来見真理* 当麻英梨子** 石川千里**

高田雅美** 城和貴**

*奈良女子大学理学部情報科学科

**奈良女子大学大学院人間文化研究科情報科学専攻

近年、幅広い分野で可視化技術は必要不可欠となっており、特に科学技術の分野ではシミュレーションや計測の結果をわかりやすく表示するために2次元図や3次元図が用いられている。本稿では、地球流体科学研究用のポータルとして開発中である Gfdnavi に3次元可視化機能を組み込み X-window のリモート機能を利用して結果をクライアント側に出力する。さらに、3次元可視化機能を GPGPU で高速化を図ることにより、より使いやすいソフトウェアの開発を考える。

3D Visualization Function for Gfdnavi of Geophysical Fluid

Mari Kurumi *, Eriko Tohma **, Chisato Ishikawa **

Masami Takata**, Kazuki Joe**

*Department of Information and Computer Sciences, Nara Women's University

**Graduate School of Humanities and Sciences, Nara Women's University

In recent years, visualization technology is essential for various scientific research. Especially, Simulation or measurement results are used as forms of visualization results to be well analyzed. In this paper, we add 3D visualization functions to Gfdnavi with outputting visualization results to client-sides through a remote X-window as well as the performance improvement with GPGPU.

1. はじめに

近年、幅広い分野で可視化技術が必要不可欠となっている。特に、科学技術分野ではシミュレーションや計測の結果をわかりやすく提示するために、古くから可視化技術が利用されている。その際、専門的な知識を持たない人に対しても理解しやすいプレゼンテーションを行うために、2次元図や3次元図が用いられている。可視化を行うための技術は複雑であるため、一般の研究者でも簡単に操作できるようにソフトウェアの開発が行われている [1]。その一つとして、Gfdnavi の開発が行われている。

現在、Gfdnavi の可視化機能としては2次元可視化のみが提供されている。これに3次元可視化機能を追加することにより、研究者にとって対象データを分析しやすいツールになると考える。そこで本研究

では、Gfdnavi への3次元可視化機能に着手した。3次元可視化機能においては、膨大なデータを扱う事による動作の遅延が懸念される。よりインタラクティブな操作を提供するために、本研究では3次元可視化機能に GPGPU を適用し、高速化を図る。

以下、2章では地球流体科学研究用のポータル Gfdnavi について述べ、3章では Gfdnavi への3次元可視化機能の追加と GPGPU についての設計指針を述べる。

2. Gfdnavi

Gfdnavi とは、地球流体電脳倶楽部が開発している地球流体データのデータベースや可視化のためのデスクトップツール兼サーバである[2]。クライアントはウェブブラウザを介してサーバ上の Gfdnavi にアクセスし、地球流体データの2次元可視化や分

析を行うことができる。クライアント側の利用者は図1のブラウザからログインし、図2の画面からGfdnaviを利用する。



図1: Gfdnaviのログイン画面

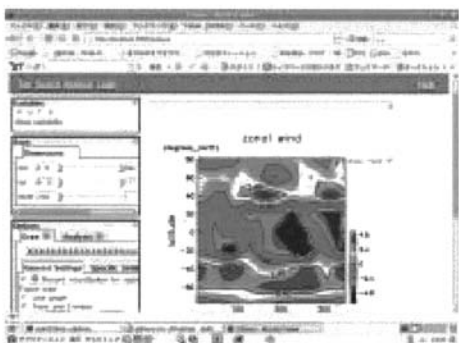


図2: Gfdnaviの2次元可視化機能画面

3. 設計指針

大量のデータを効率よく扱うために、Gfdnaviに対して3次元可視化機能の追加を行う。この機能において、大量のデータを扱うことによる動作の遅延を防ぐために、GPGPUによる高速化を図る。以下、3.1にて追加する3次元可視化機能について述べる。高速化のためにこの機能に対して適用するGPGPUについては3.2において述べる。

3.1 3次元可視化機能

3次元可視化機能の構築には Visualization ToolKit (VTK) [3]を用いる。VTKを用いて3次元ボリュームデータを読み込むことのできる3次元可視化機能を可視化サーバに設置する。VTKの機能を用いて、3次元可視化機能に対しマウスによるオブジ

ェクトの回転等インタラクティブ操作を実装する。この3次元可視化機能を以下のような形式で実現する。まず、クライアント側がサーバに対して3次元可視化機能起動命令を送信する。サーバは3次元可視化機能を起動し、クライアント側に出力結果を表示させる。クライアント側はこの出力結果に対し、更にマウスドラッグによる操作を行うことで、サーバに対して操作命令を送信する。サーバは操作命令を可視化オブジェクトに反映させ、クライアント側の出力結果を更新する。

3.2 GPGPU

GPGPUとは、General-Purpose GPU[4]のことで、GPUで数値計算を行うことである。

GPUによる数値計算では、まず、演算データをテキストチャにセットする。プログラマブルシェーダで記述された演算処理を適用し、演算結果をテキストチャへの描画を利用して保持する。そして、演算結果はテキストチャデータから読み出す。

GPGPUの計算では、データの受け渡し時間がボトルネックとなる。よって、GPGPUによる高速化が図れる処理として、演算に依存関係がなく、演算数がデータ数よりも大きいものが挙げられる。以上のことから、GPGPUに向いているアプリケーションの特徴は、データの巨大な処理を行うもので、各データ間の依存性が少なく、高い並列性で処理することであると言える。

GPUで計算を行うにあたって使用する言語で代表的なものに、Hls[5]、Gls[6]、Cg[7]、CUDA[8]という4つの言語が挙げられる。一般的には標準のC言語でプログラミングできるCUDAが利用しやすいとされているが、Geforce8以上という規定があるため、本稿ではNvidia社が開発しているCライクなCg言語を採用する。

3.3 リモート出力

本提案手法の特徴は、3次元可視化計算はすべてサーバ側にて実行し、可視化結果の表示のみをクライアント側に送信する事で、性能が低いクライアントPCにおいても高性能な可視化機能を利用可能とすることである。そのためには、サーバとクライアント間でディスプレイ出力を可能とする通信を確立する必要がある。本稿では、通信に関してはX-windowのリモート機能を適用する。X-windowリモート機能の低速通信ではGPGPUによる3次元可視化機能の高速化を生かしきれないと考えられるが、本稿では、VTKによるインタラクティブ3

次元可視化機能の Gfdnavi への追加という事に関して、実現性を実証するために試験的に採用する。

4 実装と実験

3章で述べた設計指針にしたがって実装を行う。また、3次元可視化機能への GPGPU 適用のための予備実験として、GPGPU の可視化計算性能実験を行う。プロトタイプ構築、GPGPU の可視化性能実験について、それぞれ 4.1 節、4.2 節にて延べる。現在、Gfdnavi のブラウザにボタンを追加し、3次元可視化がサーバ側で実行され、クライアント側で出力されるようになり、3次元可視化のボリュームレンダリング部分の GPGPU を用いた高速化は実装中である。

4.1 3次元可視化機能の追加

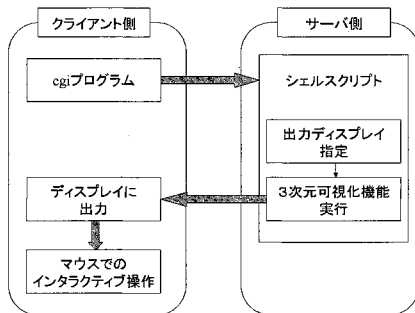


図 3: 3次元可視化機能追加の設計

VTK でボリュームレンダリングを用いた3次元可視化機能の実装を行った。これは、マウスでドラッグすることにより回転が可能で、インタラクティブな操作が可能である。

クライアント側の Gfdnavi のウェブブラウザから cgi を使ってサーバにあるシェルスクリプトを実行する。そして、シェルスクリプトから X-window の環境変数 (DISPLAY) をクライアント側の出力先に指定し、3次元可視化機能を実行する。実行された結果は、クライアント側に出力される。その際、クライアント側は可視化サーバのポートを許可しておく必要がある。

4.2 GPGPU の可視化性能実験

3次元可視化機能への GPGPU 適用のための予備実験として、GPGPU の可視化計算性能実験を行う。本実験では、ボリュームレンダリングを行うサン

ルプログラムを異なる性能を持つ5種類のGPU上で実行する。それぞれのGPUの計算時間を比較し、GPUのスペックによる計算性能の差異を確認する。

4.2.1 使用サンプルプログラム

実行するサンプルプログラムは、512*512*512の格子を持つボリュームデータをレイキャスティングによるボリュームレンダリングするプログラムである。レイキャスティング法は、並列計算に適したボリュームレンダリングの一手法であるため、GPGPUの性能評価に適していると考えられる。サンプルプログラムで用いているシェーダ言語 Cg 言語である。

4.2.2 実行環境

サンプルプログラムを実行させる環境を表1に示す。それぞれのマシンには、異なる性能のGPUが搭載されている。

| マシン番号 | GPU | GPUのメモリサイズ (MB) | VRAMのバンド幅 (GB/s) | fps |
|-------|---------------|-----------------|------------------|-------|
| 1 | QuadroFX4500 | 512 | 33.6 | 19.78 |
| 2 | QuadroFX1400 | 128 | 19.2 | 6.24 |
| 3 | GeForce7300GS | 512 | 6.5 | 4.77 |
| 4 | GeForce7200GS | 256 | 6.4 | 2.67 |
| 5 | GeForce6200 | 128 | 2.1 | 1.29 |

表 1: 実験で使ったマシンの実行環境

4.2.3 実験結果

4.2.1 節にて述べたサンプルプログラムをそれぞれのマシンにて実行させた。性能評価の指標として、1秒間の描画枚数 (fps) を用いる。表 2 に、それぞれのマシンの fps を示す。また図 4 は、それぞれのマシンの GPU スペックであるメモリサイズとバンド幅を fps と共にグラフ化したものである。

| マシン番号 | 1 | 2 | 3 | 4 | 5 |
|----------------|-------|------|------|------|------|
| 1秒間の描画枚数 (fps) | 19.78 | 6.24 | 4.77 | 2.67 | 1.29 |

表 2: GPGPU の可視化性能実験結果

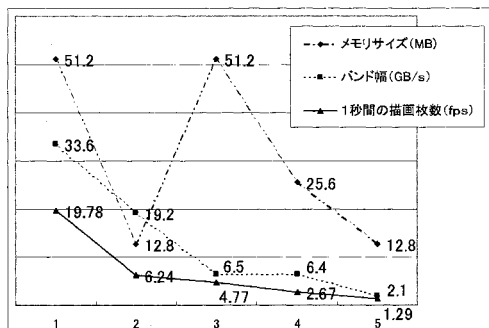


図 4：一秒間の描画枚数と GPU の比較

4.2.4 考察

図 4 より, fps とバンド幅の相関が高いことがわかる. ここで, GPU の性能による fps 値のモデル化を試みる. GPU のメモリサイズ x_1 とメモリバンド幅 x_2 を用いて, fps y が以下の式によって推定できると仮定する.

$$y = ax_1 + bx_2$$

a, b を求めるために, 実験結果のデータを用いて回帰分析を行った. その結果 $a = 0.00476$, $b = 0.452$ であった. よって, fps の推定式は

$$y = 0.00476x_1 + 0.452x_2$$

となる. 図 5 は, 推定式から得られる fps の予測値と, fps の実測値をグラフにしたものである.

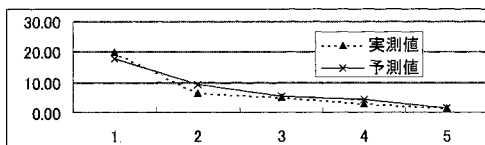


図 5：予測値と実測値

図 5 の結果から, GPU のメモリとバンド幅によっておおよその fps の推定が可能であると考えられ, サーバに搭載する GPU の検討に有効であると考えている.

最新版の GPU である NVIDIA Quadro FX 5600 のメモリサイズ 1536MB, VRAM のバンド幅 76.8GB/s では fps の推定式は

$$y = 0.00476 * 1536 + 0.452 * 76.8 = 42.02496$$

となり, 1 秒間に 42.02496 枚の描画ができると推

定できる.

5 おわりに

本研究では, 地球流体科学研究用のポータル Gfdnavi に 3 次元可視化機能を追加した. また, GPGPU の可視化性能実験により, GPGPU によるボリュームレンダリングの高速化を実証した.

今後, Gfdnavi に追加した 3 次元可視化機能を GPGPU を用いて高速化し, 地球流体物理科学者にとってより快適な可視化・分析を行える環境を構築したいと考えている. また, X-window を用いず, スナップショットを次々と送る可視化機能を Ajax を用いて実装する予定である.

参考文献

- [1] Touma, E., et. Al.: An Interactive 3D Visualization Model by Live Streaming for Remote Scientific Visualization, 3rd Intl Conf. on Computer Graphics Theory and Applications, accepted (2008)
- [2] Gfdnavi, 地球流体電脳クラブ : www.gfd-dennou.org/arch/davis/gfdnavi/
- [2] VTK: public.kitware.com/VTK/
- [3] GPGPU: www.gpgpu.org/
- [4] HlsI: www.microsoft.com/japan/msdn/directx/welcome/dsmsdn/directx04152003.aspx
- [5] GlsI: www.sgi.com/support/custeducation/emea/courses/sgi/opengl_shading.html
- [6] Cg: developer.nvidia.com/object/cg_toolkit.html
- [7] CUDA: developer.nvidia.com/object/cuda.html

謝辞

本研究遂行にあたってさまざまな協力やコメントを頂いた堀之内武氏に感謝する. なお, 本研究の一部は, 文部科学省科研費特定領域「情報爆発時代に向けた新しい IT 基盤技術の研究」課題 A01-14 (課題番号 19024039) による.