

GPU 上での CIP 法に基づく 数値シミュレーション環境の開発

1. 背景

単一コアでの性能向上が頭打ちとなった CPU はマルチコア化による性能向上を目指す段階になり、ソフトウェア側でのマルチコア対応が必須となってきた。そのために効率よい並列計算プログラミングモデルの開発が重要になってきている。

並列演算装置である GPU(Graphics Processing Unit)は 3D コンピュータ・グラフィクス処理を高速化することに特化したハードウェアで、主にゲームのグラフィクス処理に用いられている。そして最新家庭用ゲーム機から携帯電話にまで搭載されるほど身近な存在となっている。GPU 内部は高度に並列化されたベクトル型演算装置の集合体で、最新 GPU では 1 チップ内に 100 を超える並列演算装置を実装している。単一コア CPU の性能向上が近年頭打ちとなる一方で GPU は毎年ほぼ 2 倍近い性能向上を達成してきており、また Windows Vista の Aero GUI では GPU が必須となっている。GPU は今後全ての PC に搭載され、新たな身近で高性能な計算資源として特に注目されてきている。

この GPU をグラフィクス処理以外の汎用計算に応用する研究は GPGPU(General Purpose GPU)として研究分野の一つとなっており、行列ベースの古典的数値シミュレーション手法の実装・高速化等も海外の研究者により報告されている。CPU のマルチコア化はそれを活用するソフトウェア側に変革を強制するが、今後の CPU は単なるマルチコア化に留まらずヘテロニアスなマルチコア化へと確実に向かっている。これにより GPU は CPU コアに統合されて大幅な高クロック化と高速化を果たすと期待されるが、その活用には GPU アーキテクチャに関する高度な知識に基づいた計算モデルの開発が急務である。

2. 目的

本プロジェクトではベクトル型並列演算装置である GPU 上で最新数値シミュレーション手法である CIP(Cubic Interpolated Propagation)法を実装し、身近な PC 部品だけで最新の数値シミュレーションを高速に実行可能にすることを目的とする。

CIP 法は東京工業大学の矢部孝教授らにより開発された数値シミュレーションの革命的な手法で、従来手法に比べて問題を大幅に高速・高精度・安定的に解くことができる。CIP 法では電磁波の伝播や圧縮性流体のシミュレーションはもちろんのこと、従来手法では到底不可能であった気体・液体・固体などが混在する状況でも高精度なシミュレーションが行えるという革新的な特徴を持つ。さらに並列計算に向いており、実際地球シミュレータ上で行われる数値シミュレーションの大半が CIP 法およびその拡張であり、地球温暖化予測、津波や地震のシミュレーションなどに威力を発揮している。現在世界的にも最も注目される日本発の数値シミュレーション技術である。

上述の通り GPU は性能面でも普及の点からも今後最も期待される計算資源であるが活用には高度な知識が必要である。日本発の画期的な数値シミュレーション手法である CIP 法を GPU 上で実装しライブラリ化して提供すれば、ユーザは GPU に対する高度な知識無しに CPU に比べて大幅に高速で本格的なシミュレーションが可能になる。また、GPU で行ったシミュレーション結果は GPU のメモリ上にあるため、GPU を使った高速・高品質な可視化が可能となる。

3. 開発の内容

本プロジェクトでは GPU 上での CIP 法に基づく数値シミュレーション環境の開発を行い、シミュレーション自体のコア部分、シミュレーション結果の可視化ライブラリと可視化ツール、シミュレーション条件の設定を行う環境設定・モデリングツールを開発した。以下にそれぞれの部分について成果の説明を行う。

まず全体的な開発のポリシーとして、GPU は CPU と比較して圧倒的に高速であるため、シミュレーションシステム全体の性能を確保するために可能な限りの処理を CPU でなく GPU に行わせるように設計することとした。そのために既存のアルゴリズムを GPU に合わせて拡張・修正したり、GPU に向けたアルゴリズムの選定

や効率的実装のための設計に多くの検討時間を割き、試作による検証を繰り返した。また開発手順としてはまずコア部分に先行して可視化ツール類を実装し、コア部分の開発を効率よく行えるようにした。

3.1 GPU 上での CIP 法に基づく数値シミュレーションコア部分

シミュレーションのコア部分としては、CIP 法による高精度な移流計算、差分法による粘性計算、そして圧力ポアソン式の計算等を GPU 上で効率よく実装した。まず CIP 法による移流については標準の C 型 CIP 以外に M 型 CIP も実装した。また有理関数、デジタイザ、保存型にも対応した。図 1 は2次元移流問題として有名な Zalesak ディスク問題を例題とし、2次元 CIP で正確な移流が行えることとの検証と、処理時間の CPU との比較を行っている様子である。移流はきわめて高精度に行われ、また処理時間は CPU と比較して数倍～数十倍という結果であった。実際にはシミュレーションのグリッド数が増大して計算負荷が増えるほど GPU が相対的に高速に処理可能という結果であった。これは GPU での処理は CPU からのドライバの呼び出しによって行われるため、シミュレーショングリッド数と無関係に一定の時間がかかってしまうため、グリッド数の増加による計算時間の増大によってドライバ呼び出しのオーバーヘッドが相対的に減少するためである。図 2 は3次元空間内での CIP 移流の実行例で、回転速度場による移流によっても CIP 法でちゃんと境界面の追跡が行われていることを示している。

次に差分法による粘性計算の実装を行ったが、これは比較的単純な作業であるため特筆すべき点は特にない。圧力ポアソン式の計算のために、GPU 上でヤコビ法、ガウスザイデル法およびレッド・ブラック・ガウスザイデル法を実装した。また共役勾配法の実装も行った。

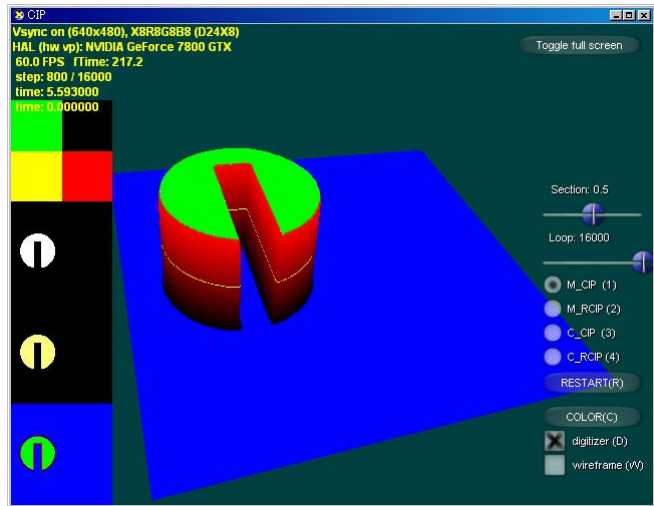


図 1: 2次元 CIP の動作確認画面
(Zalesak ディスクによる検証)

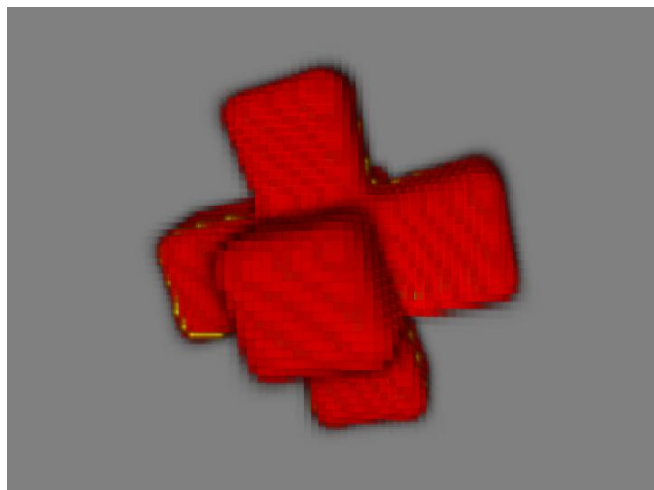


図 2: 3次元 CIP の動作確認画面

3.2 可視化ライブラリとツール

GPU 上で CIP 法に基づくシミュレーションが可能になったとして、次に重要なのは結果の可視化である。GPU 上でのシミュレーション結果は GPU 上に保持されているため、これを GPU を用いて可視化するのは非常に効率がよく、また GPU は本来表示処理用のハードウェアであるため可視化は得意である。我々は GPU プログラミングの知識と経験を活かし、他に類のない高速・高品位な可視化を実現した。

図 3 は2次元流体シミュレーションにおいてカルマン渦が発生している様子を可視化したものである。流れ場のようなベクトル場の可視化については我々はパーティクル(粒子)を流れ場に沿って移動させることで流れの様子を分かりやすく表示する方法と、流れの大きさと向きを表す矢印を画面内に配置する方法を実装した。これらは全てリアルタイムに可視化可能であるため、ユーザはシミュレーション中に自由に可視化方法を変更することが可能である。粒子や矢印には速度や圧力などに応じて色を付けることも可能で、これにより一層シミュレーション結果の理解を手助けできる。また圧力や温度などのスカラー値については、値を色に対応させて

表示する方法や、等値線を表示する方法を実装した。これによってスカラー値の分布の変化をリアルタイムに把握可能になった。これらの可視化手法はそれぞれライブラリとして実装され、必要に応じてアプリケーションに組み込むことも可能である。また可視化手法は自由に組み合わせる表示可能であるので、矢印と粒子と等値線を一度に表示することも可能である。可視化のための種々のパラメータは GUI によりユーザが自由に変更可能である。

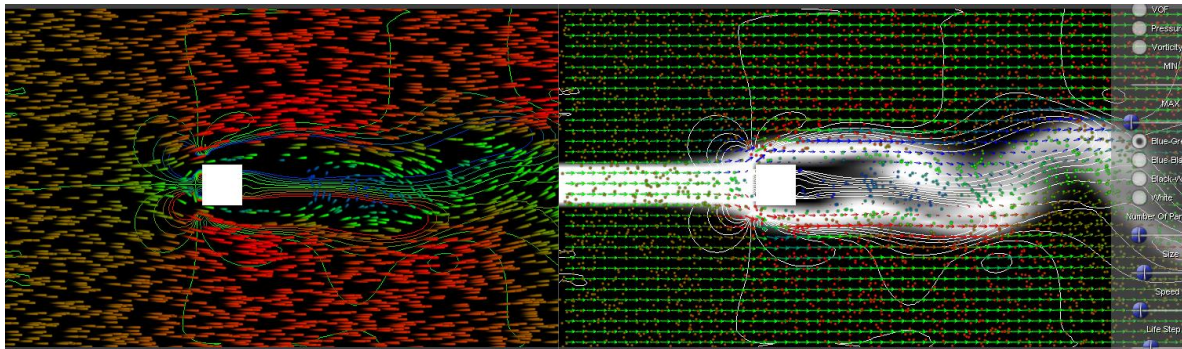


図 3: 2次元流体シミュレーションによるカルマン渦の可視化例(左:粒子追跡、右:矢印と密度)

図 4 は3次元シミュレーションの可視化例である。3次元シミュレーション結果についてもパーティクルや矢印によって流れ場をリアルタイムに可視化することが可能である。またスカラー場については指定平面上に分布図や等値線を表示することも可能である。また本プロジェクトで開発された GPU 上のシミュレーションによる結果の表示だけでなく、別の外部のシミュレーションシステムによる計算結果の表示にも対応するため、外部ファイルからの読み込みにも対応した。

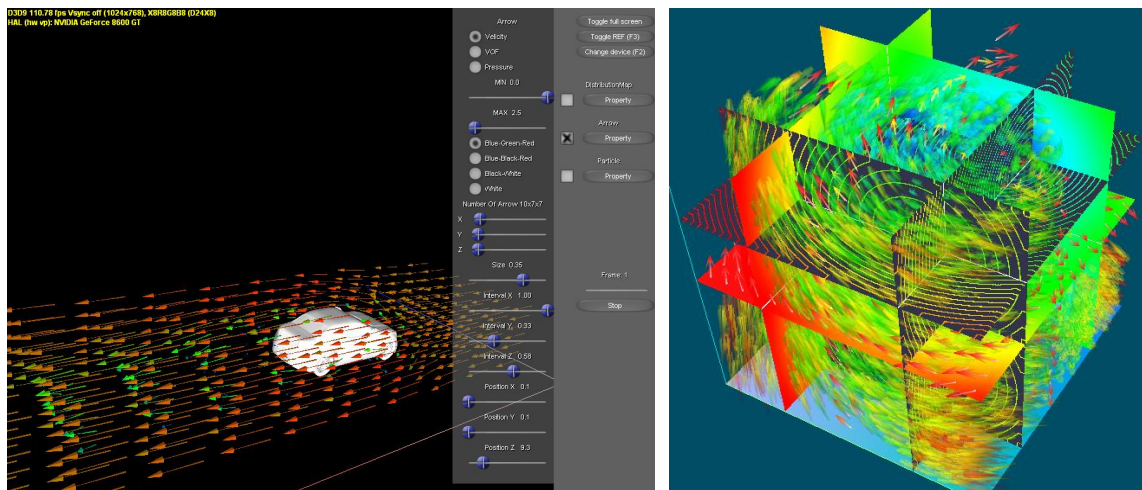


図 4: 3次元流体シミュレーションのリアルタイム可視化例

3.3 環境設定・モデリングツール

数値シミュレーションを行うためには、まずシミュレーション環境の設定を行わなければならない。必要な物理パラメータの設定、障害物や壁の配置などを簡単に行えて、そのままシミュレーションが可能になるような環境設定ツールを開発した。このツールも通常的手法ではリアルタイムでの設定やモデリングが困難であるが、今回は GPU の利用を前提に対話的に環境設定・モデリングを可能とした。図 5 はそのツールの利用画面の例で、左側の図では2次元シミュレーションのための障害物の配置と編集を行っている。右側では3次元シ

ュレーションのための障害物のモデリング例であり、この車のモデルに対してシミュレーションを行うことで車体の周りの空気の流れや空気抵抗を評価することが可能になる。この図の車体を用いたシミュレーション結果が図4の左図である。

またこのツールはシミュレーションを目的としない3次元モデルの構築に利用することもでき、物体の切削が容易に行えることから金型の切削シミュレーション等にも利用可能である。

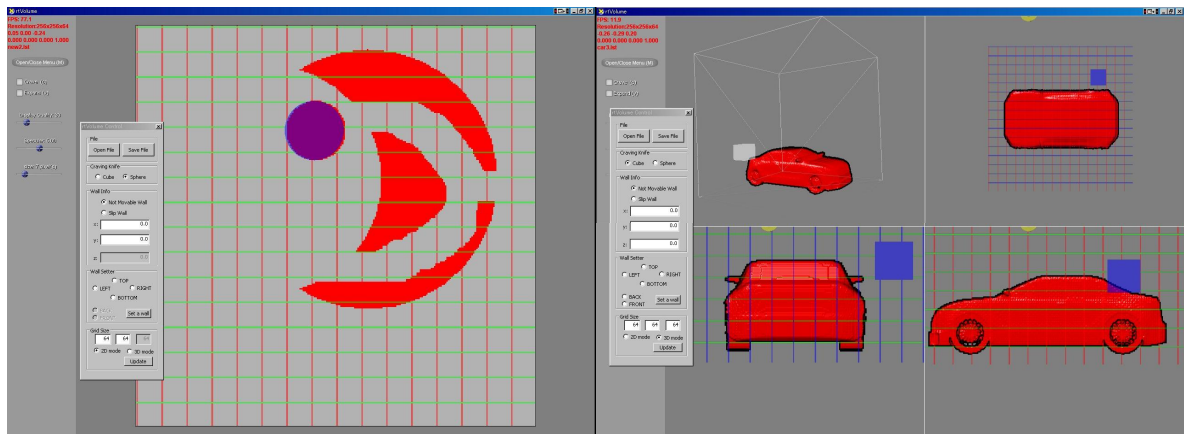


図 5: 環境設定ツールの利用例
(左:2次元、右:3次元)

4. 従来の技術(または機能)との相違

本プロジェクトの開発成果であるシミュレーション環境の特徴は、既存技術に基づく製品と比較して以下の通りである。

- ・既存製品よりもはるかに高速に実行可能であるため、シミュレーション時間を大幅に短縮できるか、同じ時間内にずっと精度の高いシミュレーションを実行可能であること。
- ・高速・高品質なリアルタイム可視化も可能で、環境設定ツールも付属するなど、シミュレーションのコアのみならず「環境」として既存の高価な製品よりも性能・機能共に大幅に優れていること。

5. 期待される効果

日本発の本格的流体数値シミュレーション手法を身近で高速な計算資源であるGPU上で利用可能になるため、流体数値シミュレーションに関心がある初心者ユーザから本格的に利用している専門家にまで広範囲のユーザに恩恵がある。流体数値シミュレーションと高速可視化は多くの産業分野において有用であり、開発成果は非常に広範囲に利用可能である。また GPU は毎年2倍以上の高速化を続けており、将来的にも有望な計算資源であるため、開発成果の利用価値はますます向上すると期待できる。

6. 普及(または活用)の見通し

開発成果の利用条件としては、身近にあるごく一般的な GPU を備えていることのみであり、多くの人に恩恵がある。またツールも含めた環境として提供するものであり広範囲に普及・活用されることを期待している。今後も様々な面で拡張を進め、さらに本格的に利用可能なものとしてゆく予定である。

7. 開発者名(所属)

安藤英俊(山梨大学大学院医学工学総合研究部)

鳥山孝司(山梨大学大学院医学工学総合研究部)