GPGPU によるレオロジー物体変形シミュレーションの高速化

Simulation Acceleration of Rheological Deformation based on GPGPU

○木下　正稔（立命館大学）　平井　慎一（立命館大学）

Masatoshi KINOSHITA, Ritsumeikan University
Shinichi HIRAI, Ritsumeikan University

In our daily life, there are many rheological objects like human tissues and human organs. FEM (Finite Element Method) is widely used in the numerical computation of rheological object deformation. Compared to another deformation method, FEM can simulate the deformation more accurate but takes longer computation time. Simulation of rheological object deformation is often required of real time processing in surgical simulation. To shorten the computation time, we apply GPGPU (General-Purpose computing on Graphics Processing Units) in the computation of rheological deformation. This computation mainly consists of matrix computation. Thus, we will implement matrix computation to GPGPU for real time computation.

***Key Words*:** GPGPU, Surgical Simulation, FEM

1. 緒言

近年の医療技術の発展に伴い，医師に要求される技術も高度化しており，その背景の下，手術シミュレーションが注目されている．人体などの生体組織のように，弾性物体と塑性物体の中間の変形を示す物体のことをレオロジー物体という．従来，手術に限らず，コンピュータによる物体変形のシミュレーションに関する研究は盛んに行われてきた．物体変形のシミュレーションをコンピュータ上で行うためにはモデリング手法が必要となる．

モデリング手法の一つである有限要素法は，複雑な形状であっても再現性が高いというメリットがあり幅広く利用されている．しかし有限要素法によって求められた運動方程式の行列は大きくなりがちで，シミュレーションに多くの時間がかかるといったデメリットがある．

このようなシミュレーション時間を短縮するため，画像処理専用の演算装置である GPU に注目が集まっている．本報告では，有限要素法に必要な行列計算を GPU で処理することによる高速化手法を提案し，その効果を検証する．

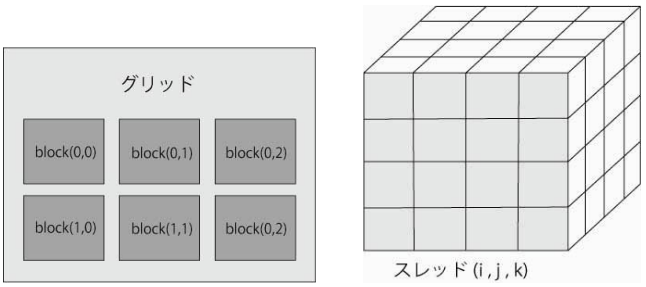
2. GPGPU

GPGPU とは General-Purpose computing on Graphics Processing Units の略称で，画像処理専用の演算装置である GPU を汎用計算に利用することの総称である． GPU の特徴として並列処理に特化した高い演算性能が挙げられる．これは画像処理が単純で大量の処理計算を求められるためであり，シミュレーションなどの汎用計算において，並列性の高い処理を行う場合において CPU よりも高い演算性能を得る．

なお本研究において，GPGPU のフレームワークである CUDA を使用した．CUDA とは Compute Unified device Architecture の略称で，NVIDIA 社からリリースされている GPGPU を目的とした統合開発環境である．シミュレーションの高速化にあたって重要な概念であるスレッドとメモリについて説明する．

CUDA の仕様では，最高で 65535×65535×512 個のスレッドの実行を命令することができる[1]．このような多数のスレッドを階層的に管理するためにグリッドとスレッドという概念がある．図 1(a)と図 1(b) にグリッド，ブロック，スレッドの概念図を示す．グリッド中のブロックは二次元的に管理され，ブロック中のスレッドは三次元的に管理される．この圧倒的に多いスレッドで計算することで GPU の高い性能を引き出すことができる．

次にメモリについて，CUDA では複数種類のデバイスメモリを使用することができる．例えばデバイスメモリの中にはシェアードメモリとグローバルメモリがあり，メモリへのアクセス速度はシェアードメモリの方が速い．しかしシェアードメモリはデータを共有できる範囲が各ブロック内のみで，メモリ容量も 16KB と少ない. 一方グローバルメモリはすべてのスレッドとホスト間で共有でき，またグローバルメモリとして使えるメモリ容量は GPU のメモリ搭載量と同じで 1GB を超すものも珍しくない．デバイスメモリについてまとめたものを表 1 に示す．この中から目的に応じて適切なメモリを使うことで GPU の性能を引き出すことができる．



(a)グリッドとブロック　(b)ブロックとスレッド

図 1　グリッド・ブロック・スレッド概念図

表 1　デバイスメモリ

メモリ種類	アクセス	使える範囲
レジスタ	R/W	当該スレッド内
シェアードメモリ	R/W	ブロック内の全てのスレッド
グローバルメモリ	R/W	全てのスレッドとホスト
コンスタントメモリ	R	全てのスレッドとホスト

3. 有限要素法による運動方程式

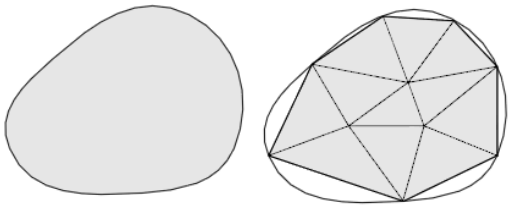


図 2　二次元物体要素

有限要素法は物体を図 2 のように有限個の要素の集合とみなし，個々の要素の変形から物体全体の変形を求めるモデリング手法である．二次元物体の場合，物体を三角形要素の集合とみなして計算するのが一般的であり，この物体がレオロジー変形特性をもつとき，各三角形要素のノード点座標における運動方程式は以下のようになる．ただし ω_λ，ω_μ は二次元レオロジー変形の動的方程式を導くために導入したベクトルである．

$$\begin{bmatrix} \dot{u}_N \\ \dot{v}_N \\ \lambda \end{bmatrix} = \begin{bmatrix} I & & \\ & M & -A \\ & -A^T & \end{bmatrix}^{-1} \begin{bmatrix} \dot{v}_N \\ -\lambda^{ela} J_\lambda \omega_\lambda - \mu^{ela} J_\mu \omega_\mu + f_{ex} \\ A^T\left(2\omega v_N + \omega^2 u_N\right) \end{bmatrix} \quad (1)$$

$$\dot{\omega}_\lambda = -\frac{\lambda^{ela}}{\lambda_1^{vis}+\lambda_2^{vis}}\omega_\lambda + \frac{\lambda_2^{vis}}{\lambda_1^{vis}+\lambda_2^{vis}} v_N + \frac{\lambda_1^{vis}\lambda_2^{vis}}{\lambda_1^{vis}+\lambda_2^{vis}}\dot{v}_N \quad (2)$$

$$\dot{\omega}_\mu = -\frac{\mu^{ela}}{\mu_1^{vis}+\mu_2^{vis}}\omega_\mu + \frac{\mu_2^{vis}}{\mu_1^{vis}+\mu_2^{vis}} v_N + \frac{\mu_1^{vis}\mu_2^{vis}}{\mu_1^{vis}+\mu_2^{vis}}\dot{v}_N \quad (3)$$

ここで u_N，v_N は各ノード点における変位と速度，f_{ex} は外力，M は慣性行列，J_λ と J_μ は接続行列であり，$A^T\left(2\omega v_N + \omega^2 u_N\right)$ の式は物体の拘束のための制約安定化法である．

これらの式を積分することで各ノード点の変位を得る．各ノード点につき積分に必要な行列の要素数 E は拘束条件を除くと 4 変数$(\dot{u}_N, \dot{v}_N, \dot{\omega}_\lambda, \dot{\omega}_\mu)$×2 方向(x, y)分必要である．例えば図 3.1 のようにノード点が 10 個の場合，行列の要素数 E は最小で 80[個]となり，80 行 80 列×80 行 1 列の行列積計算を行う．

4. 実験

4.1 実験概要

行列の要素数 E を変更し，CPU と GPU についてそれぞれの処理時間を計測する．ループ回数について，有限要素法の時間ステップ幅が 0.001 秒と想定し，10 秒分のシミュレーション時間である 10000 回の行列計算を行う．

GPU による計算では，要素数 E 個に対して E^2 個のスレッドによる並列処理によって計算を行う．

4.2 実験環境

実験に使用したマシン環境を以下に示す．

- CPU:intel Core2Quad，クロック周波数:2.83GHz，
  メモリ:3.24GB
- GPU:NVIDIA GeForce GT 240，クロック周波数:1.34GHz，
  メモリ:512MB

5. 実験結果

行列の要素数 E を徐々に増やしていったときのシミュレーションにかかる時間を計測した．以下にその結果を示す．

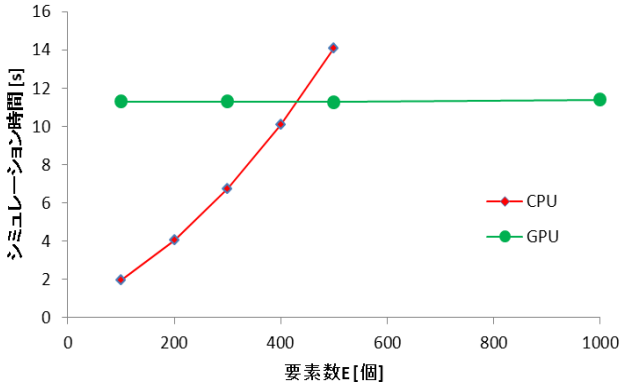


図 3　要素数に対するシミュレーション時間

図 3 より，CPU による処理時間は要素数 E に対して増加している．一方で GPU による処理時間は要素数 E によらず 11 秒ほどで安定している．

6. 考察

行列要素数が増えるにつれて CPU のシミュレーション時間は大幅に大きくなっているのに対し，GPU の処理時間はほとんど変わらないことを示した．本報告では行列計算しか行っていないが，人体のように複雑な形状である場合や三次元物体の場合，行列要素数は非常に大きくなるため GPU による高速化は有効と思われる．

7. 結言

本報告では GPGPU を用いて有限要素法によるレオロジー物体変形シミュレーションに必要な行列計算の並列化を提案し，その結果を示した．結果より要素数 E が大きいほど GPU の性能を発揮し，高速化することができた．

文　献

[1] 青木尊之，額田彰，"はじめての CUDA プログラミング"，工学社, pp.44-45, 2009