GPU の演算能力を用いた SPH シミュレーションの高速化に関する研究

京都大学　正会員　○小野　祐輔
京都大学　学生員　矢野　裕介
京都大学　正会員　清野　純史

１．背景と目的

近年，様々な数値解析手法において，計算速度向上のために GPU(Graphics Processing Unit)の活用が試みられている．GPU は画像処理に関わる演算処理に特化して開発が行われてきたものであるが, CPU(Central Processing Unit)に比べて，搭載される演算ユニットの数が多く浮動小数点演算が高速であること，同程度の演算性能に対して消費電力が小さいこと，価格が安いことといった利点がある．

そこで，本研究においても，GPU の演算性能を用いることにより粒子法の一種である SPH(Smoothed Particle Hydrodynamics)法の高速化を試みた．本研究では，NVIDIA 社によって開発されている CUDA(Compute Unified Device Architecture) [1)]という CPU と GPU の統合的処理の開発環境を用いて，GPU によるスレッド並列化演算に対応した SPH 法の解析コードを開発した．

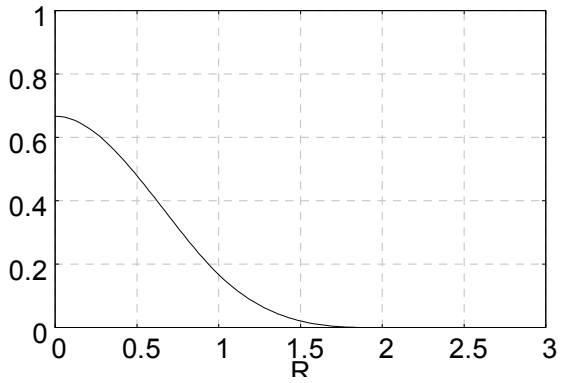

図-1　カーネル関数

２．SPH 法

SPH 法では，解析対象物体を粒子と呼ばれる微小要素に分割し，次式によって近似することで位置 $\mathbf{x}^i$ における物理量 $f(\mathbf{x}^i)$を求める．

$$f(\mathbf{x}^i)=\sum_{j=1}^{N}\frac{m^j}{\rho^j}f(\mathbf{x}^j)W(R,h) \tag{1}$$

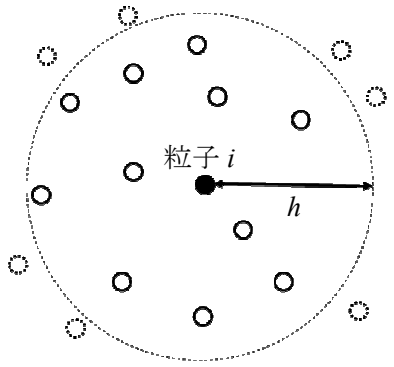

図-2　計算に用いる近傍粒子の概念

ここで, 上付き添え字 i, j は粒子を表すインデクスであり，m^j，ρ^j はそれぞれ各粒子の持つ質量，密度である．W はカーネル関数と呼ばれ，図-1 に示したような形状を持つ，二つの粒子間隔 $R=|\mathbf{x}^j-\mathbf{x}^i|$に関する関数である．$h$ は R に対するカーネル関数の広がりを定義するパラメータであり，W は $R<h$ の範囲において $W>0$，それ以外の範囲では全て零となる．したがって，図-2 のように粒子 i に対して物理量 $f(\mathbf{x}^i)$を求めるには，粒子 i から距離 h の範囲にある粒子 j (j=1,...,N) の持つ値 $f(\mathbf{x}^j)$が用いられる．

本研究では非圧縮性粘性流体を対象として解析コードを作成した. 非圧縮性粘性流体への SPH 法の適用に際しては Monaghan[2)]に従った．

３．解析コードの作成

SPH 法では，近傍粒子探索，密度の計算，速度の更新，座標の更新が各タイムステップ毎に繰り返して行われる．これらの処理のうち，近傍粒子の探索以外は粒子毎に計算が行えるため，CUDA によるスレッド並列化は容易である．近傍粒子探索に関しては，全粒子間において判定をすると膨大な計算コストが必要となる．そ

キーワード：SPH, GPU, CUDA，粒子法
連絡先：〒615-8540　京都府京都市西京区京都大学桂　C クラスタ　C1-2 棟 138 号室　TEL. 075-383-3251

のため，多くの粒子法にて行われているように，本研究での GPU 対応の元となった従来の解析コードでも，解析対象領域をセルに分割し，計算コストを下げる手法をとったいた．このコードでは粒子番号順に各粒子が位置するセルを求め，セル毎に含まれる粒子を格納するリストを作成していた．しかしながら，この作業を CUDA によるスレッド並列を用いて行った場合，リストへのデータ書き込みの競合が問題となる．そこで，Green[3]によるソートを用いたセル毎に格納される粒子リストの作成アルゴリズムを利用した．

４．解析例

作成した解析コードを用いて，二次元[2]および三次元[4]の水柱崩壊問題を対象とした解析を実施し，CPU による計算と GPU を用いた計算の実行時間を比較した．CPU による計算を実行した環境は Intel Core i7-920 Processor (8M Cache, 2.66 GHz, 4.80 GT/s Intel(R) QPI，4 コア)，メモリ 8GB を搭載した PC である．オペレーティングシステムとしては，OpenSUSE Linux 11.1 の 64-bit 版が導入されている．一方，GPU による計算は，同じ PC に NVIDIA 製の Tesla C1060，9800GTX+，GTX275 を搭載して行った．また，開発・実行に用いた CUDA のバージョンは 2.3 であり，C++で書かれた解析コードのコンパイルには g++ Version4.3.2 を用いた．**図-3** は三次元解析結果を可視化した一例を示したものである．

解析に用いる粒子の個数を変化させ，CPU のみの計算に対する GPU を用いた計算速度の向上率を**図-4** のように得た．二次元解析に対しては，粒子を 100 万個用いた計算に対して 23 倍程度の計算速度の向上がみられた．三次元解析に対しても，同様に粒子 100 万個を用いた計算において，およそ 18 倍の計算速度向上が実現できた．ただし，三次元解析の場合は粒子数が 50 万個となった時点で計算速度の向上は頭打ちとなった．

参考文献

1) NVIDIA CUDA Programming Guide, Version 3.0, 2010.

2) Monaghan, J.J., Simulating Free Surface Flows with SPH, *Jounal of Computational Physics*, Vol.110, pp.399-406, 1994.

3) Gree, S.:CUDA Particles, NVIDIA, 2007.

4) Issa, R. and Violeau, D.:3D schematic dam break and evolution of the free surface, SPH Benchmark Test Cases, http://wiki.manchester.ac.uk/spheric/index.php/Validation_Tests, 2006.

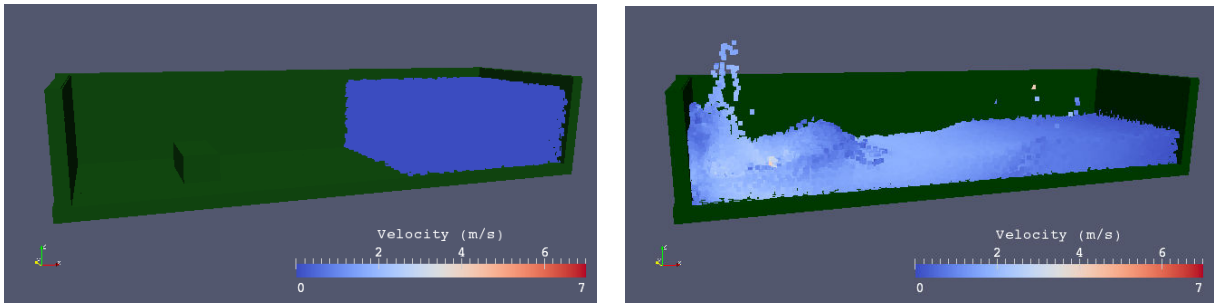

図-3 三次元解析のスナップショット

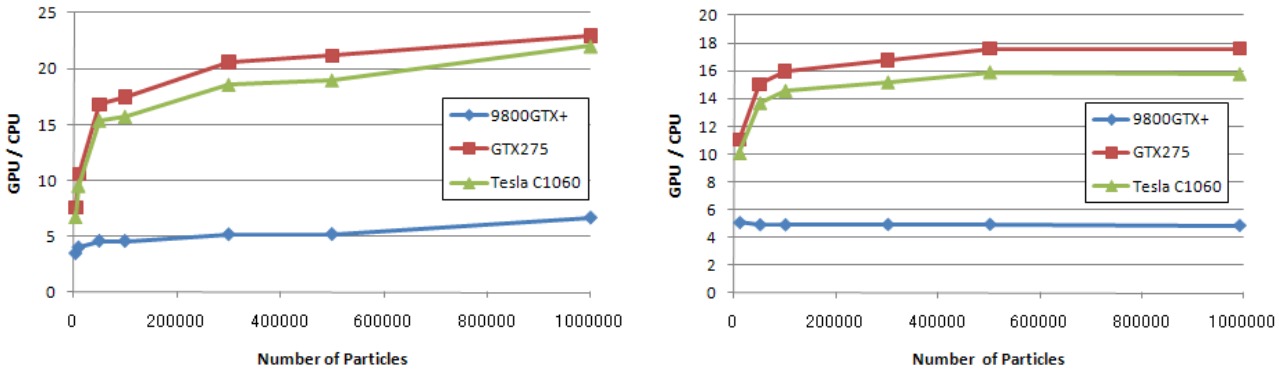

図-4 計算速度の向上率（左：二次元解析，右：三次元解析）