超高速 GPU 並列計算を利用した医用超音波イメージングのための数値解析法の確立

代表研究者　　大久保　　寛　　首都大学東京　システムデザイン学部准教授

1 目的・背景

本研究は，GPU（Graphics Processing Unit）による超高速並列計算と超音波・波動伝搬シミュレーション技術を開発することが目的である．

現在，医工学分野では生体への影響の少ない超音波による計測が多く行われている．特に，生体内超音波の非線形伝搬現象については，高調波を利用した超音波エコー法（Tissue Harmonic Imaging：THI）として，医用領域での工学的な応用を視野に可視化技術(アコースティック・イメージング）の開発が始まっている．しかし，超音波の伝搬シミュレーションの手法は確立しているとは言えない．

そこで，本研究では，そのためのシミュレーション技術として FDTD+CIP Hybrid 法を適用し，新しい手法の開発を行い，高速 GPU 並列計算とともに検討する．

2 数値解析

2-1 概要

近年の計算機環境の向上は，多くの分野での数値シミュレーションを可能としている．現在一般的に用いられているシミュレーション手法の 1 つに，時間領域での数値シミュレーションが挙げられる．この時間領域での解析手法はいくつかあるが，その中でも FDTD(finite difference time domain)法は音響シミュレーションの際に最も広く用いられている手法である．FDTD 法においては，時間・空間の中心差分が支配方程式の導関数の近似に用いられる．中心差分近似は精度により様々な差分式とすることができるが，コンパクト性や境界の取り扱いを考えると 2 次精度差分が一般的によく使われる．

そのため，FDTD 法は他の手法と比較して計算アルゴリズムが容易であり，また波源や構造物のモデル化が容易であるという長所があるが，一方で数値分散による誤差が生じるという欠点がある．この誤差は，波動伝搬に伴って蓄積されていくため数値計算上大きな問題となる．これにより大規模な波動伝搬を伴う音響シミュレーションでは，波長分割数によっては正確な結果を導くことができない．特にシミュレーション結果を可聴やイメージングへフィードバックさせようとする場合にはこの数値分散を低減する必要がある．

一方，CIP（the constrained interpolation profile/cubic-interpolated pseudo particle)法が近年音響数値シミュレーションにおいて新しい計算手法として報告されている．本手法は，マルチモーメントの考え方を特性曲線法に取り入れた手法であり，数値分散が極めて小さい手法である．この手法の特徴は，グリッド上の値だけでなくその空間微分値も用いることである．ただし，CIP 法は微分値を必要とするため，音源や物体のモデル化が FDTD 法に比べると若干面倒である．

以上のように，2 つの手法にはいずれも優れた点があるが，いずれもそのままでは大規模なシミュレーションに対応できないという問題点がある．そのため，これらの問題点を解決する必要がある．

2-2 数値解析手法

（1）FDTD 法

FDTD(finite difference time domain)法は，数値シミュレーションにおいて現在最も広く用いられている手法である．これは，変数に音圧と粒子速度を用いて音場の支配方程式である運動方程式と連続の式を，時間領域と空間領域において差分化する手法である．線形，無損失の場合，音場の支配方程式(運動方程式と連続の式)は

$$\frac{\partial p}{\partial t} = -\kappa\left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}\right) \tag{1}$$

$$\frac{\partial v_x}{\partial t}=-\frac{1}{\rho}\frac{\partial p}{\partial x},\frac{\partial v_y}{\partial t}=-\frac{1}{\rho}\frac{\partial p}{\partial y},\frac{\partial v_z}{\partial t}=-\frac{1}{\rho}\frac{\partial p}{\partial z} \tag{2}$$

で与えられる．ただし，p は音圧，v_x, v_y, v_z はそれぞれ x, y, z 方向の粒子速度，κ は体積弾性率，ρ は密度である．本研究では，主に z 軸方向に一様な 2 次元の音場について考えているため，次の条件が付加される．

$$\frac{\partial p}{\partial z}=\frac{\partial v_z}{\partial z}=0 \tag{3}$$

したがって，2 次元音場の支配方程式は(1)～(3)により，次のように与えられる．

$$\frac{\partial p}{\partial t}=-\kappa\left(\frac{\partial v_x}{\partial x}+\frac{\partial v_y}{\partial y}\right) \tag{4}$$

$$\frac{\partial v_x}{\partial t}=-\frac{1}{\rho}\frac{\partial p}{\partial x},\frac{\partial v_y}{\partial t}=-\frac{1}{\rho}\frac{\partial p}{\partial y} \tag{5}$$

FDTD 法において使用されるグリッドは，図 1 の Staggered grid である．ただし，p は音圧，v_x と v_y はそれぞれ x 方向と y 方向の粒子速度である．このグリッドでは，音圧と粒子速度は時間的・空間的に 1/2 グリッドずらして配置され，支配方程式(4)(5)を中心差分近似により直接離散化する．

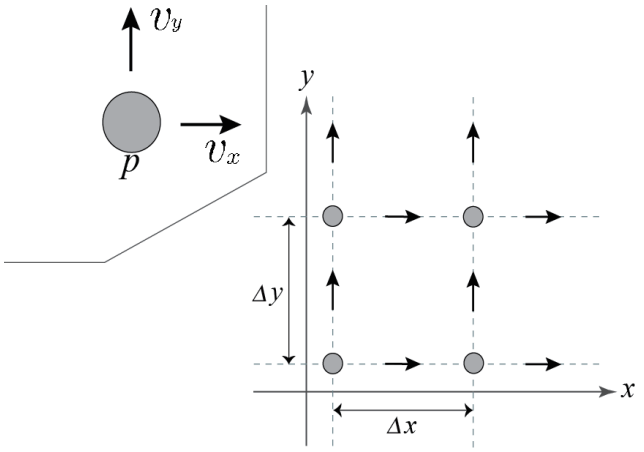

図 1　FDTD 法のグリッド

まず，時間方向の離散化を行う．時間方向にだけ微分を 2 次精度中心差分で離散化した式は，

$$\frac{p^{n+1}(i,j)-p^{n}(i,j)}{\Delta t}=-\kappa\left(\frac{\partial v_x}{\partial x}\Big|_{i,j}^{n+1/2}+\frac{\partial v_y}{\partial y}\Big|_{i,j}^{n+1/2}\right) \tag{6}$$

$$\frac{v_x^{n+1/2}(i+1/2,j)-v_x^{n-1/2}(i+1/2,j)}{\Delta t}=-\frac{1}{\rho}\frac{\partial p}{\partial x}\Big|_{i+1/2,j}^{n} \tag{7}$$

$$\frac{v_y^{n+1/2}(i+1/2,j)-v_y^{n-1/2}(i+1/2,j)}{\Delta t}=-\frac{1}{\rho}\frac{\partial p}{\partial y}\Big|_{i+1/2,j}^{n} \tag{8}$$

となる．(6)式ついては$t=(n+1/2)\Delta t$ において差分化し，(7)式と(8)式については$t=n\Delta t$ において，それぞれ差分化している．さらに，時間更新に着目して(6)～(8)式を変形すると，

$$p^{n+1}(i,j)=p^{n}(i,j)-\kappa\Delta t\left(\frac{\partial v_x}{\partial x}\Big|_{i,j}^{n+1/2}-\frac{\partial v_y}{\partial y}\Big|_{i,j}^{n+1/2}\right) \tag{9}$$

$$v_x^{n+1/2}(i+\frac{1}{2},j)=v_x^{n-1/2}(i+\frac{1}{2},j)-\frac{\Delta t}{\rho}\frac{\partial p}{\partial x}\Big|_{i+1/2}^{n} \tag{10}$$

$$v_y^{n+1/2}(i+\frac{1}{2},j)=v_y^{n-1/2}(i+\frac{1}{2},j)-\frac{\Delta t}{\rho}\frac{\partial p}{\partial y}\Big|_{i+1/2}^{n} \tag{11}$$

となる．次に，空間方向に差分化を行う．式(9)～(11)より，

$$p^{n+1}(i,j)=p^{n}(i,j)-\kappa\Delta t\left\{\begin{array}{l}\dfrac{u_x^{n+1/2}(i+1/2,j)-u_y^{n+1/2}(i-1/2,j)}{\Delta x}\\ +\dfrac{u_y^{n+1/2}(i,j+1/2)-u_y^{n+1/2}(i,j-1/2)}{\Delta y}\end{array}\right\} \tag{12}$$

$$v_x^{n+1/2}(i+\frac{1}{2},j)=v_x^{n-1/2}(i+\frac{1}{2},j)-\frac{\Delta t}{\rho}\frac{p^{n}(i+1,j)-p^{n}(i,j)}{\Delta x} \tag{13}$$

$$v_y^{n+1/2}(i,j+\frac{1}{2})=v_y^{n-1/2}(i,j+\frac{1}{2})-\frac{\Delta t}{\rho}\frac{p^{n}(i,j+1)-p^{n}(i,j)}{\Delta y} \tag{14}$$

となる．ここで，$\Delta x,\Delta y$ はx,y 方向の空間刻み幅，Δt は時間刻み幅，i,j はx,y 座標に対応する空間離散地点，n は離散時刻を表している．この(12)～(14)式を標準の FDTD 法として，ハイブリッド化の際に用いる．

（1）CIP 法

CIP (the constrained interpolation profile/cubic-interpolated pseudo particle)法は，近年音響数値シミュレーションにおいて新しい計算手法として報告されている．この手法の特徴は，グリッド上の値だけでなくそれらの空間微分値も用いることである．そのため，数値分散に伴う誤差が小さく，大規模なシミュレーションにも対応することが可能である． 使用するグリッドは，図 2 の Collocated grid である． ただし， p は音圧， v_x と v_y はそれぞれ x 方向と y 方向の粒子速度であり， $\partial_x=\partial/\partial x,\partial_y=\partial/\partial y$ である．

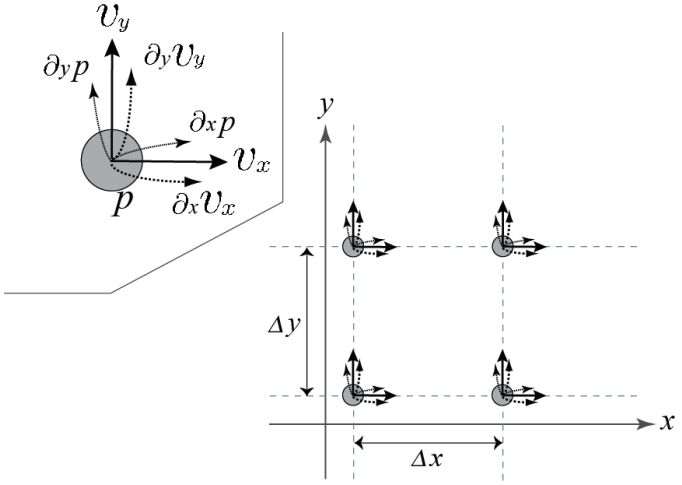

図 2　CIP 法のグリッド

CIP 法の運動方程式と連続の式は，次の通りである．

$$\rho\frac{\partial\vec{v}}{\partial t}=-\nabla p \tag{15}$$

$$\nabla\vec{v}=-\frac{1}{\kappa}\frac{\partial p}{\partial t} \tag{16}$$

ただし，ρ は密度，κ は体積弾性率，p は音圧，$\vec{v}$ は粒子速度である．以下，簡単のため，まず 1 次元で考える．(15)式を変形し，さらに $c=\sqrt{\kappa/\rho}$ を乗じることにより，次のように変形できる．

$$\frac{\partial}{\partial t}Zv_x+c\frac{\partial}{\partial x}p=0 \tag{17}$$

続いて，(16)式についても同様にして

$$\frac{\partial}{\partial t}p+c\frac{\partial}{\partial x}Zv_x=0 \tag{18}$$

(17)式と(18)式の和と差をとることにより

$$\frac{\partial}{\partial t}(p+Zv_x)+c\frac{\partial}{\partial x}(p+Zv_x)=0 \tag{19}$$

$$\frac{\partial}{\partial t}(p-Zv_x)+c\frac{\partial}{\partial x}(p-Zv_x)=0 \tag{20}$$

$f_{\pm}=p\pm Zv$ とおくことにより

$$\frac{\partial}{\partial t}f_{\pm}+c\frac{\partial}{\partial x}f_{\pm}=0 \tag{21}$$

が得られる．これが CIP 法の移流方程式である．さらに，(21)式を x で偏微分すると

$$\frac{\partial}{\partial t}\frac{\partial f_{\pm}}{\partial x}+c\frac{\partial}{\partial x}\frac{\partial f_{\pm}}{\partial x}=0 \tag{22}$$

さらに $g_{\pm}=\dfrac{\partial f_{\pm}}{\partial x}$ とおくことにより

$$\frac{\partial}{\partial t}g_{\pm}+c\frac{\partial}{\partial x}g_{\pm}=0 \tag{23}$$

となる．これらは y 方向についても同様にして導くことができる．次に，3 次エルミート補間を用いてタイムステップ n のグリッド上の値からタイムステップ $n+1$ におけるグリッド上の値を求める方法を述べる．補間関数 $F_{i\pm}^{n}(x)$ は

$$F_{i\pm}^{n}(x)=a_{3\pm}(x-x_i)^3+b_{3\pm}(x-x_i)^2+c_{3\pm}(x-x_i)+d_{3\pm} \tag{24}$$

(24)式を微分して

$$G_{i\pm}^{n}(x)=3a_{3\pm}(x-x_i)^2+2b_{3\pm}(x-x_i)+c_{3\pm} \tag{25}$$

ただし，

$$a_{3\pm}=\frac{g_{\pm}^{n}(i)+g_{\pm}^{n}(i\mp1)}{(\Delta x)^{2}}\mp\frac{2\left\{f_{\pm}^{n}(i)-f_{\pm}^{n}(i\mp1)\right\}}{(\Delta x)^{3}} \tag{26}$$

$$b_{3\pm}=\frac{3\left\{f_{\pm}^{n}(i\mp1)-f_{\pm}^{n}(i)\right\}}{(\Delta x)^{2}}\pm\frac{2g_{\pm}^{n}(i)+g_{\pm}^{n}(i\mp1)}{\Delta x} \tag{27}$$

$$c_{3\pm}=g_{\pm}^{n}(i) \tag{28}$$

$$d_{3\pm}=f_{\pm}^{n}(i) \tag{29}$$

である．ただし，$\Delta x,\Delta t$ はそれぞれグリッドサイズとタイムステップである．上記の(24)～(29)式は，グリッド上の境界条件より求めることができるが，通常補間計算を行う場合は以下のように積和計算の形にして値を求める．

$$\begin{aligned}F_{x\pm}^{n+1}(i,j)=&C_{\pm}(1)F_{x\pm}^{n}(i\mp1,j)+C_{\pm}(2)F_{x\pm}^{n}(i,j)\\&+C_{\pm}(3)G_{x\pm}^{n}(i\mp1,j)+C_{\pm}(4)G_{x\pm}^{n}(i,j)\end{aligned} \tag{30}$$

$$\begin{aligned}G_{x\pm}^{n+1}(i,j)=&C'_{\pm}(1)F_{x\pm}^{n}(i\mp1,j)+C'_{\pm}(2)F_{x\pm}^{n}(i,j)\\&+C'_{\pm}(3)G_{x\pm}^{n}(i\mp1,j)+C'_{\pm}(4)G_{x\pm}^{n}(i,j)\end{aligned} \tag{31}$$

ただし，

$$C_{\pm}(1)=-2\chi^{3}+3\chi^{2} \tag{32}$$

$$C_{\pm}(2)=2\chi^{3}-3\chi^{2}+1 \tag{33}$$

$$C_{\pm}(3)=-2\xi_{\pm}(\chi^{2}-\chi) \tag{34}$$

$$C_{\pm}(4)=-2\xi_{\pm}(\chi^{2}-2\chi+1) \tag{35}$$

$$C'_{\pm}(1)=6(-\chi^{3}+\chi^{2})/\xi_{\pm} \tag{36}$$

$$C'_{\pm}(2)=6(\chi^{3}-\chi^{2})/\xi_{\pm} \tag{37}$$

$$C'_{\pm}(3)=3\chi^{2}-2\chi \tag{38}$$

$$C'_{\pm}(4)=3\chi^{2}-4\chi+1 \tag{39}$$

である．ここで，$\xi_{\pm}=\mp c\Delta t,\chi=c\Delta t/\Delta x$ である．$C_{\pm}()$ 及び $C'_{\pm}()$ は媒質で決まる定数である．

2-3 ハイブリッド法

（1）ハイブリッド手法への考え方

前述の通り，FDTD 法は散乱体などや波源のモデル化が容易である一方で，伝搬解析において数値分散による誤差が生じる．この誤差が波動伝搬に伴い蓄積されていくため，正確な結果を得ることができない．そのため，より数値分散誤差が小さい手法が必要となる．一方，CIP 法は数値分散誤差が小さい一方で，微分値を用いるため構造物や波源のモデル化が煩雑となる．そこで，両手法を組み合わせ互いの長所を活かすことを考える．

シミュレーションのモデル化を FDTD 法で，その伝搬解析を CIP 法で行うことができれば，モデル化が行いやすく伝搬解析に伴う数値分散誤差が小さい手法を実現することが可能となる．またさらなる利点として，高次の FDTD 法を利用する場合に比べてタイムステップを大きく設定することが可能である．組み合わせる際

には，両手法の使用するグリッドが異なるため両者の境界条件の取り扱いが重要となる．そこで，次章ではその境界条件の取り扱いについて詳細に検討する．特に，境界で発生する反射の大きさや境界の伝搬に伴う伝搬波の減衰についての評価を行う．境界条件のモデルは図 3 のようになる．ただし，i_0 は境界である．

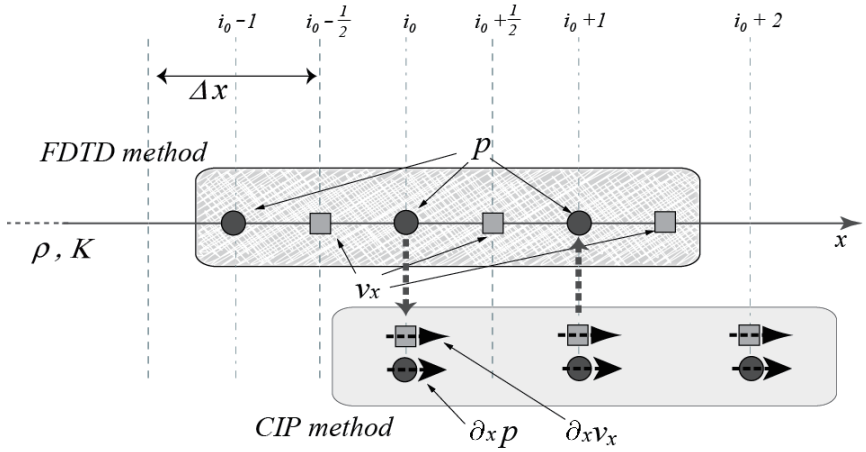

図 3　境界条件のモデル

（２）境界の補間条件について

ハイブリッド法の境界の取り扱いモデルを図 4 に示す．ただし，n はタイムステップ，i_0 は境界，○は音圧，◇は音圧の微分値，□は粒子速度，△は粒子速度の微分値を示している．

境界における計算手順を以下に示す．

・最初に FDTD 領域において粒子速度から時刻 n の音圧を導く（①）．

・CIP 領域の i_0+1 の音圧の値を FDTD 領域の i_0+1 の音圧とする（②）．

・②の値と①で求めた音圧の値から FDTD 領域の粒子速度を導く（③）．

・③で求めた粒子速度と①で求めた音圧を CIP 領域の初期入力とする（④）．

・④より時刻 $n+1$ の値を導出する（⑤）．

・①に戻る（⑥）．

この手順を繰り返す．これにより，FDTD 領域から CIP 領域へ値を移すことで CIP 領域の境界の値が決定する． この際，使用するグリッドが異なるため境界で反射が生じる．反射の大きさは，音圧や粒子速度及びこれらの微分値のサポート点数によって異なるため，次にその精度について詳細に検討する．

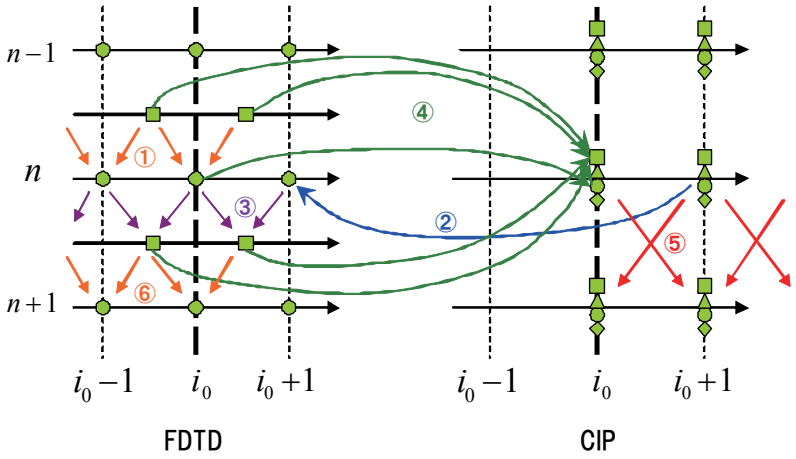

図 4　ハイブリッドのモデル

境界条件の精度はサポート点数によって異なる．そこで，音圧や粒子速度及び微分値の有無を変え，最も精度のよい条件を決定する．今回検討したのは，表 1 の type1～type9 の全 9 通りである．

表 1　境界の条件一覧

	音圧微分の補間精度	粒子速度の補間精度	微分の扱い(微分値の考慮)
type1	2 次精度	1 次精度	有
type2	2 次精度	2 次精度	有
type3	2 次精度	3 次精度	有
type4	2 次精度	1 次精度	無
type5	2 次精度	2 次精度	無
type6	1 次精度	1 次精度	有
type7	1 次精度	2 次精度	有
type8	1 次精度	1 次精度	無
type9	1 次精度	2 次精度	無

FDTD 領域から CIP 領域に値を移す際は，音圧を基準としてグリッドを合わせているため，音圧については全てに共通で以下の式(40)のようになる(図 4 中の④)．

$$p^n_{(C)}(i_0) \leftarrow p^n_{(F)}(i_0) \tag{40}$$

音圧微分の補間精度については，1 次精度と 2 次精度を用いており，1 次の場合は式(41)のように，2 次の場合は式(42)のようになる．また，境界での微分値を考慮しない場合には式(43)のようになる．

$$\partial_x p^n_{(C)}(i_0) \leftarrow \frac{p^n_{(F)}(i_0) - p^n_{(F)}(i_0 - 1)}{\Delta} \tag{41}$$

$$\partial_x p^n_{(C)}(i_0) \leftarrow \frac{p^n_{(F)}(i_0 + 1) - p^n_{(F)}(i_0 - 1)}{2\Delta} \tag{42}$$

$$\partial_x p^n_{(C)}(i_0) \leftarrow 0 \tag{43}$$

粒子速度については，1 次精度～3 次精度までの 3 種類を用いており，1 次の場合には式(44)のように，2 次の場合には式(45)のように，3 次の場合には式(46)のようになる．

$$v^n_{x(C)}(i_0) \leftarrow \frac{1}{4}\begin{pmatrix} v^{n-1/2}_{x(F)}(i_0) + v^{n-1/2}_{x(F)}(i_0 - 1) \\ + v^{n+1/2}_{x(F)}(i_0) + v^{n+1/2}_{x(F)}(i_0 - 1) \end{pmatrix} \tag{44}$$

$$v^n_{x(C)}(i_0) \leftarrow \frac{1}{20}\begin{pmatrix} -2v^{n-1/2}_{x(F)}(i_0 + 1) + 9v^{n-1/2}_{x(F)}(i_0) + 3v^{n-1/2}_{x(F)}(i_0 - 1) \\ -2v^{n+1/2}_{x(F)}(i_0 + 1) + 9v^{n+1/2}_{x(F)}(i_0) + 3v^{n+1/2}_{x(F)}(i_0 - 1) \end{pmatrix} \tag{45}$$

$$v^n_{x(C)}(i_0) \leftarrow \frac{1}{32}\begin{pmatrix} 5v^{n-1/2}_{x(F)}(i_0) + 15v^{n-1/2}_{x(F)}(i_0 - 1) - 5v^{n-1/2}_{x(F)}(i_0 - 2) + v^{n-1/2}_{x(F)}(i_0 - 3) \\ 5v^{n+1/2}_{x(F)}(i_0) + 15v^{n+1/2}_{x(F)}(i_0 - 1) - 5v^{n+1/2}_{x(F)}(i_0 - 2) + v^{n+1/2}_{x(F)}(i_0 - 3) \end{pmatrix} \tag{46}$$

粒子速度の微分の補間精度については，音圧微分と同様に 1 次精度と 2 次精度の 2 種類を用いており，1 次の場合は式(47)のように，2 次の場合は式(48)のようになる(どちらも同じ計算式となる)．また，微分値を考慮しない場合には，式(49)のようになる．

$$\partial_x v^n_{x(C)}(i_0) \leftarrow \frac{1}{2\Delta x}\begin{pmatrix} v^{n-1/2}_{x(F)}(i_0) - v^{n-1/2}_{x(F)}(i_0 - 1) \\ + v^{n+1/2}_{x(F)}(i_0) - v^{n+1/2}_{x(F)}(i_0 - 1) \end{pmatrix} \tag{47}$$

$$\partial_x v^n_{x(C)}(i_0) \leftarrow \frac{1}{2\Delta x}\begin{pmatrix} v^{n-1/2}_{x(F)}(i_0) - v^{n-1/2}_{x(F)}(i_0-1) \\ +v^{n+1/2}_{x(F)}(i_0) - v^{n+1/2}_{x(F)}(i_0-1) \end{pmatrix} \tag{48}$$

$$\partial_x v^n_{x(C)}(i_0) \leftarrow \frac{1}{48\Delta x}\begin{pmatrix} 23v^{n-1/2}_{x(F)}(i_0) - 21v^{n-1/2}_{x(F)}(i_0-1) - 3v^{n-1/2}_{x(F)}(i_0-2) + v^{n-1/2}_{x(F)}(i_0-3) \\ +23v^{n+1/2}_{x(F)}(i_0) - 21v^{n+1/2}_{x(F)}(i_0-1) - 3v^{n+1/2}_{x(F)}(i_0-2) + v^{n+1/2}_{x(F)}(i_0-3) \end{pmatrix} \tag{49}$$

$$\partial_x v^n_{x(C)}(i_0) \leftarrow 0 \tag{50}$$

ただし，上記の式(40)～式(49)のいずれにおいても，C は CIP 領域を，F は FDTD 領域を，i_0 は境界を示している．

今回の境界条件は，上記の式(40)と，式(41)～式(49)を組み合わせて使用する．例えば，type2 の場合は音圧微分と粒子速度の補間精度がともに 2 次であり微分値を考慮するため，使用される境界の計算式は式(40)，式(42)，式(45)，式(48)となり，以下のようになる．

$$p^n_{(C)}(i_0) \leftarrow p^n_{(F)}(i_0)$$
$$\partial_x p^n_{(C)}(i_0) \leftarrow \frac{p^n_{(F)}(i_0+1) - p^n_{(F)}(i_0-1)}{2\Delta}$$
$$v^n_{x(C)}(i_0) \leftarrow \frac{1}{20}\begin{pmatrix} -2v^{n-1/2}_{x(F)}(i_0+1) + 9v^{n-1/2}_{x(F)}(i_0) + 3v^{n-1/2}_{x(F)}(i_0-1) \\ -2v^{n+1/2}_{x(F)}(i_0+1) + 9v^{n+1/2}_{x(F)}(i_0) + 3v^{n+1/2}_{x(F)}(i_0-1) \end{pmatrix}$$
$$\partial_x v^n_{x(C)}(i_0) \leftarrow \frac{1}{2\Delta x}\begin{pmatrix} v^{n-1/2}_{x(F)}(i_0) - v^{n-1/2}_{x(F)}(i_0-1) \\ +v^{n+1/2}_{x(F)}(i_0) - v^{n+1/2}_{x(F)}(i_0-1) \end{pmatrix}$$
$$p^n_{(F)}(i_0+1) \leftarrow p^n_{(C)}(i_0+1)$$

同様に，type8 の場合は音圧と粒子速度が 1 次精度であり，微分値を考慮していないため境界の計算式は式(40)，式(43)，式(44)，式(49)となり，以下のようになる．

$$p^n_{(C)}(i_0) \leftarrow p^n_{(F)}(i_0)$$
$$\partial_x p^n_{(C)}(i_0) \leftarrow 0$$
$$v^n_{x(C)}(i_0) \leftarrow \frac{1}{4}\begin{pmatrix} v^{n-1/2}_{x(F)}(i_0) + v^{n-1/2}_{x(F)}(i_0-1) \\ +v^{n+1/2}_{x(F)}(i_0) + v^{n+1/2}_{x(F)}(i_0-1) \end{pmatrix}$$
$$\partial_x v^n_{x(C)}(i_0) \leftarrow 0$$
$$p^n_{(F)}(i_0+1) \leftarrow p^n_{(C)}(i_0+1)$$

以下では，これら 9 通りの組み合わせの精度について詳細に検討する．

（３）境界条件の精度について

まず，1 次元の反射についての検討を行った．評価は，伝搬波が境界を通過する際に FDTD 領域側に生じる反射波の大きさを評価した．はじめに，境界条件において微分値を考慮しているもの(表 1 の type1～type3, type6, type7)について比較を行った．結果を図 5 に示す．ただし，横軸は PPW(point per wavelength)であり，縦軸は反射の大きさを dB で表記している．

図より，最も反射が小さくなるものは type2 であり，その傾向は PPW が大きくなるにつれて顕著になっている．これより，これら 5 つの type の中で最も精度がよいものは type2 であるといえる．また，全体として PPW が小さくなるほど反射が大きくなっていることも特徴として挙げられる．

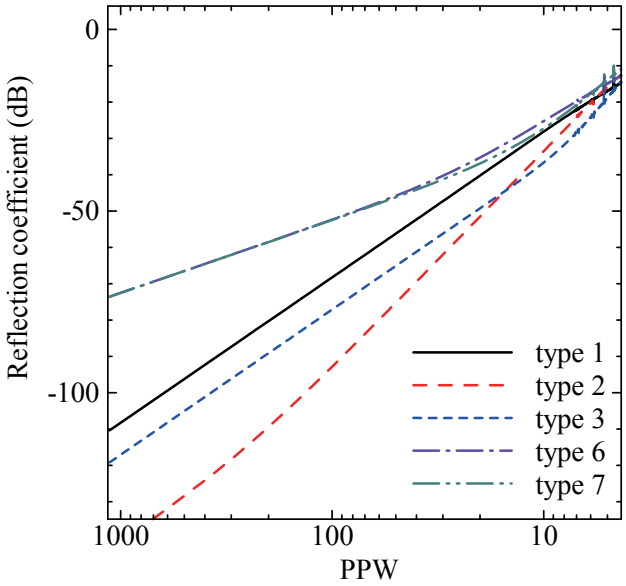

図 5　1 次元の反射特性①(微分値を考慮する場合)

続いて，図 5 において最も精度がよかった type2 と微分値を用いないもの(表 1 の type4，type5，type8，type9)との比較を行った．結果を図 6 に示す．ただし，縦軸と横軸は図 5 と同様である．図より，最も反射が小さいのは type2 であり，微分値を考慮していない他の 4 つはほぼ同じ結果となっている．また，図 5 と同様に PPW が大きくなるほど反射が大きくなっている．

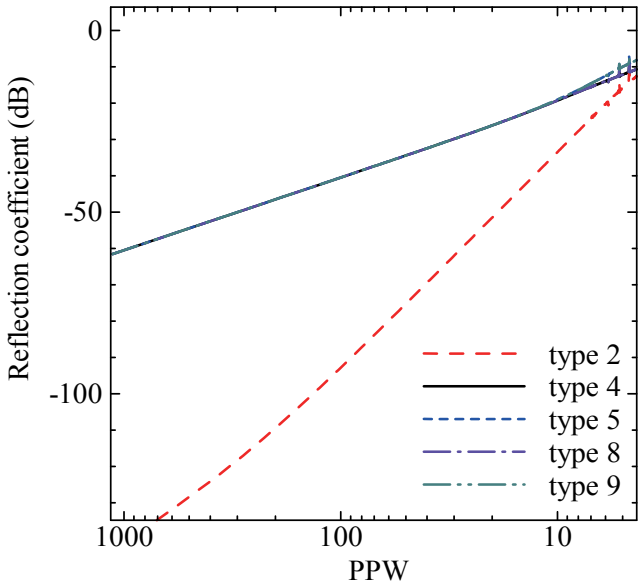

図 6　1 次元の反射特性②(微分値を考慮していない場合)

以上より，最も精度がよいものは type2 の音圧と粒子速度がともに 2 次精度で，微分値を考慮している場合であることがわかった．また，境界条件において微分値を与えない場合に反射が大きくなっていることから，FDTD 領域から CIP 領域への値の引渡しには微分値の効果が非常に大きいこともわかった．

次に，境界の透過に伴う伝搬波の減衰について評価を行う．この境界の透過に伴う伝搬波の減衰が小さいほど，精度がよいということができる．この評価は 1 次元で行った．まず，表 1 の中で微分値を考慮しているものについて比較した．結果を図 7 に示す．ただし，縦軸は伝搬波の減衰を境界の前後の比で示しており，0 に近いほど減衰が小さいということができる．横軸は周波数である．

この図より，最も減衰が小さいのは type2 であることがわかる．続いて，この type2 と微分値を考慮して

いないものとを比較した．結果を図 8 に示す．軸の設定は図 7 と同様である．この図より，最も精度がよいものは type2 であることがわかる．

よって，これらの図と先ほどの反射特性の結果を合わせると，表 1 のうち最も精度がよいものは type2 であるということができる．

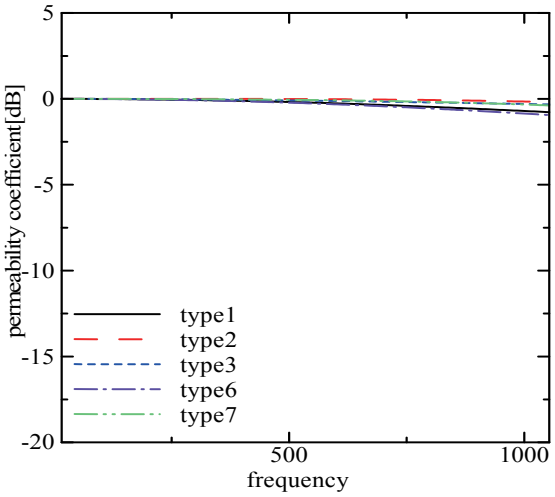

図 7　伝搬波の減衰①(微分値を考慮したもの)

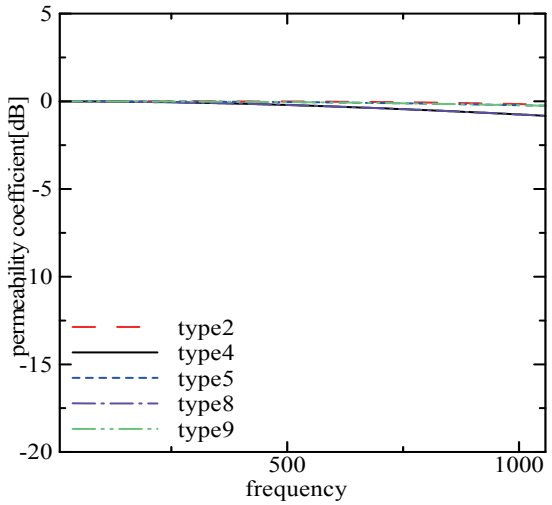

図 8　伝搬波の減衰②(type2 と微分値を考慮していないもの)

2-4 ハイブリッドシミュレーション結果

本節では，前節で決定した最も精度のよい境界条件である type2 を中心にハイブリッド法を音響空間に適用する．検討にあたっては，ハイブリッドと従来法である FDTD 法のみを用いた場合を比較し，ハイブリッド法の有効性について言及する．

使用したモデルは図 9 の通りである．グリッドサイズは $\Delta x = 0.05$ であり，グリッド数は 1000×1000 としている．また，FDTD 領域と CIP 領域の境界グリッドは 450，550 としている．この FDTD 領域の中心に初期条件として音圧を空間的に与え，境界に前章の条件を適用した．ここでは，最も精度がよい type2 と最も精度が悪い type8，及び FDTD 法のみで計算した 3 パターンについて比較検討する．最初に，全体図で比較する．境界条件に type2 を用いたものを図 10 に，type8 を用いたものを図 11 に，CIP 領域を設定せず全領域を FDTD 法のみを図 12 に示す．図 10 と図 11 を比較すると大きな違いは見受けられないが，これらに対し図 12 は全体に大きく歪んでいることから，ハイブリッド法が有効であることがわかる．

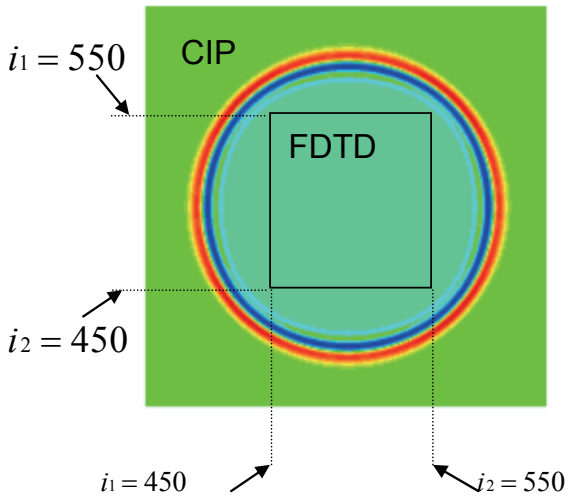

図 9　音圧の空間分布のモデル

続いて，ハイブリッド法の精度を検証するために，type2 と type8 について下記の図 10 と図 11 を鳥瞰図で比較することで，反射の比較を行った．図 13 に type2 を，図 14 に type8 を示す．図 13 では中心付近にほとんど反射の成分が見られないのに対し，図 14 ではかなりの反射成分が見られる．このことから，微分値の設定が境界の反射特性に大きな影響を与えていることがわかる．

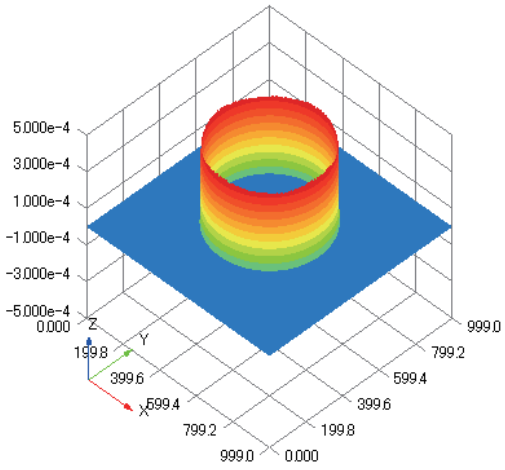

図 10　type2 を用いたもの

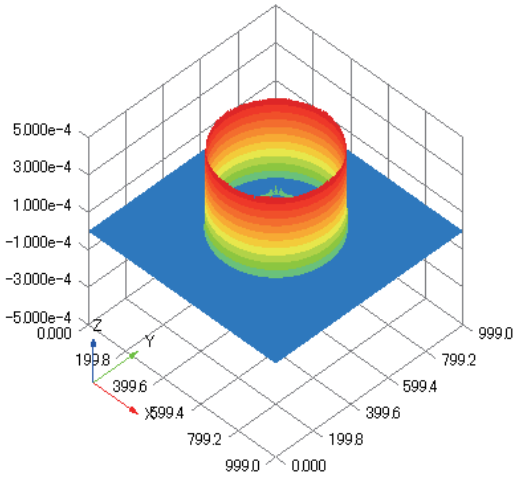

図 11　type8 を用いたもの

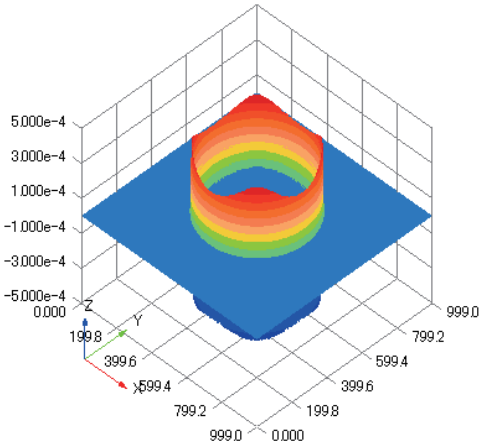

図 12　全領域を FDTD 法のみで計算したもの

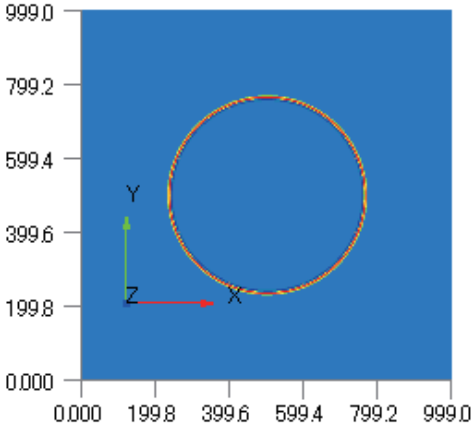

図 13　type2 の鳥瞰図

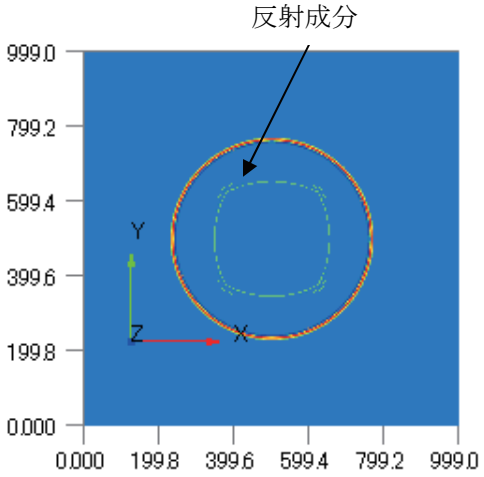

図 14　type8 の鳥瞰図

次に，中心点を通る線上の音圧分布を比較する．図 15 が図 13(type2)に対応し，図 16 が図 14(type8)に対応している．両図とも，黒線は x 方向の線上の音圧分布（$y=500$）であり，赤線は斜め方向（$(x,y)=(0,0)$ から $(x,y)=(1000,1000)$ まで）の音圧分布である．また，横軸は図 13 と図 14 の 500 グリッドを中心としている．図 15 では中心付近にいずれも反射成分が見られるのに対し，図 16 ではそのような成分は見られない．これらのことからも，微分値の設定が境界条件の精度に大きく影響を与えることがわかる．

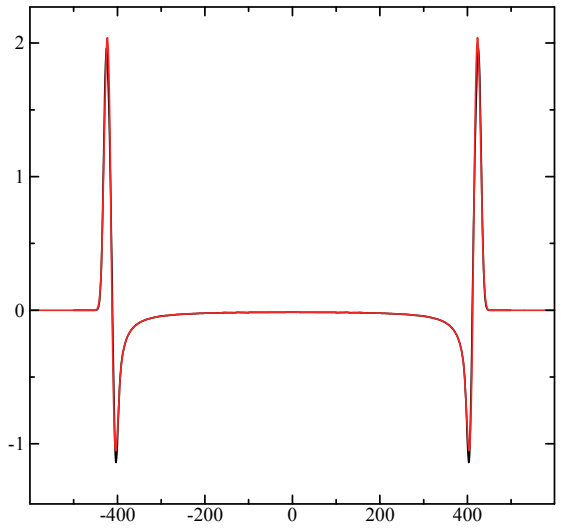

図 15　図 12(type2)の音圧分布

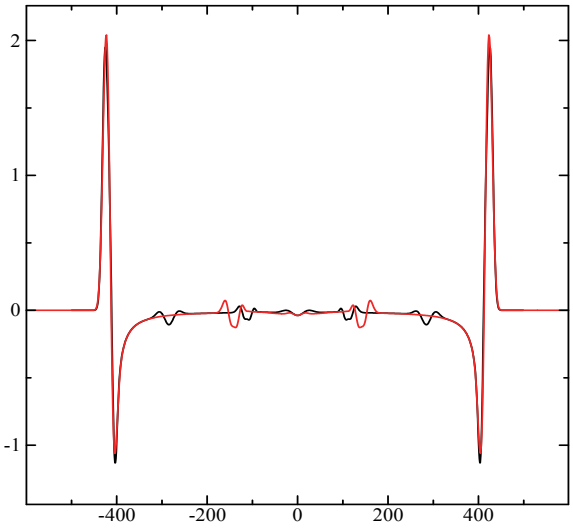

図 16　図 13(type8)の音圧分布

3 GPU による計算

3-1 GPU 化

計算の高速化という点では，従来スーパーコンピュータやクラスタなどの大型の計算機が利用されているが，最近では GPU(Graphics Processing Unit)を用いて汎用的な数値計算を行おうとする GPGPU(General Purpose computation on GPUs)が様々な分野で注目され始めている．現在では GPU の演算性能は数年前より飛躍的に向上しており，最新の GPU は少し前のスーパーコンピュータ並の性能を秘めているとも考えられる．

ここでは音響数値シミュレーションのための GPU によるパーソナルスーパーコンピュータの実現へ向けて，マルチ GPU を搭載した計算機を用いた高速並列計算による音響数値シミュレーションを試みる．

3-2 計算手法の概要

前節でも述べた計算手法であるが，ここで簡単にまとめる．FDTD 法は Staggered grid を用いて支配方程式を直接差分化する手法である．本手法において音圧と粒子速度は 1/2 グリッド離れた点に配置される．以降では，もっとも基本的な解法である時間・空間 2 次精度の中心差分近似によって解く手法（Yee-FDTD 法）を用いている．

GCIP 法は音圧・粒子速度という物理量とともに，それらの空間微分値を同時に用いることで計算を行う手法で，CIP 法を一般化した計算方法である．GCIP(3,1)法が従来の CIP 法にあたる．FDTD 法とは異なり，Collocated grid を用いており，音圧と粒子速度及びその空間微分値は同一グリッド上に配置される．いずれの手法も時間領域で解く手法であり，各グリッド上の値はそれぞれ独立して求めることができるため並列化には向いている．

3-3 CUDA による GPU プログラミング

GPU は本来画像処理に特化したアーキテクチャとして生まれたものであるが，最近では並列計算を行う場合などでは CPU を使った計算より圧倒的な高速化が可能であるという情報もある．

GPGPU の初期の段階では，コンピュータグラフィックス向けの専用言語を用いてプログラミングする必要があつたため,変数の型に GPU 特有の型しか使えないなど汎用的なプログラムの記述はかなり困難であつた．しかしながら，GPGPU 向けのプログラミング言語として CUDA(Compute Unified Device Architecture)などの C 言語に近い言語が開発されたことにより，C 言語の知識だけで比較的手軽にプログラミング可能となった．

CUDA（NVIDIA GPU）のハードウェアモデルを図 17 に示す．同図は本研究で用いた Tesla C1060 のモデルであり，30 個のマルチプロセッサ(MP)で構成され，各 MP では 8 個のストリーミングプロセツサ(SP)が並列に動作する．各 MP は高速アクセス可能な共有メモリ(SM)を持っており，これは 8 個の SP で共有されている．

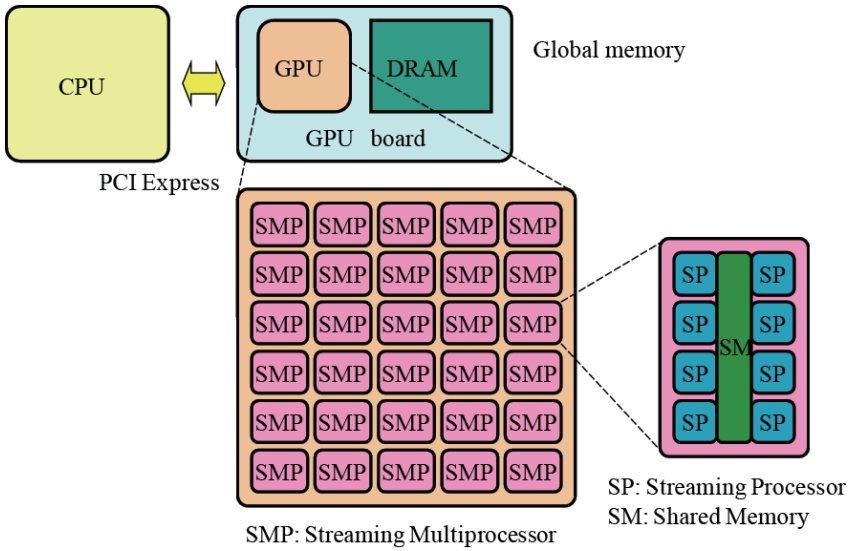

図 17　ハードウェアモデル

GPU 上のグローバルメモリ(GM)から MP へのメモリ転送はかなり高速ではあるが,MP の並列計算に掛かる時間と比べるとデータ転送時間が総計算時間のかなりの部分を占めることになる．したがって，計算アルゴリズムによるが頻繁に GM にアクセスする必要がある場合には，参照するデータを一旦 SM へ転送してから演算を行う方が高速に処理できる．

CUDA における最小実行単位はスレッド（Thread）と呼ばれる．さらに 32 スレッドをワープと呼び，1 ワープの単位で MP の中の多数の SP により並列実行される．また，CUDA では図 18 に示すように，スレッドのまとまりをブロック，ブロックのまとまりをグリッドと呼び管理している．グリッドは PC（ホスト側）から実行を指令する単位で，グリッド内の全スレッドは同じプログラム(カーネルと呼ぶ)を実行する．

以下に CUDA で GPU プログラミングを行う場合,高速化の重要なポイントであるスレッドモデルについてまとめる．

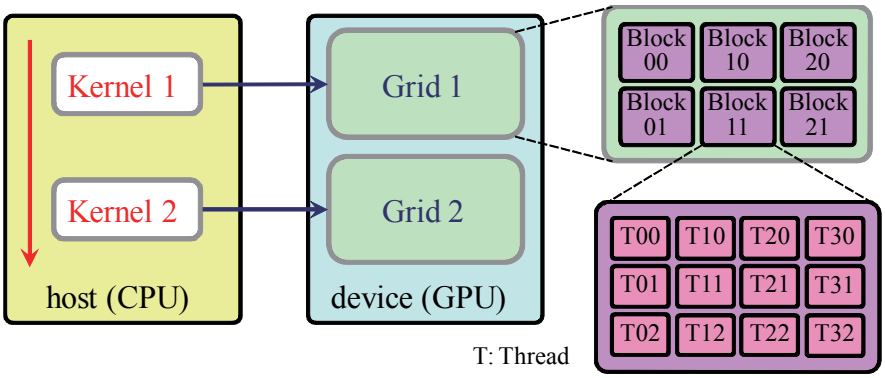

図 18　スレッドモデル

スレッド配置のモデルについて述べる．

・1 ブロック内のスレッド数

前述の通り，CUDA-GPU では 1 ワープの単位で実行される．したがって，1 ブロック内のスレッド数は 32 の整数倍であると実行時に無駄が少ない．

一方，1 ブロック内のスレッド数には限界があり，最大で 512 スレッドが割り当てられる．

・1MP に対するスレッド数

実行時に 1MP でアクティブになることができるスレッド数には限界があり，1024 スレッドである．

・1MP に対するブロック数

実行時に 1MP でアクティブになることができるブロック数には限界があり，最大が 8 ブロックである．

・メモリ内のデータ転送について

グローバルメモリへのアクセスが coalesced access か non coalesced access かという違いは計算時間に大きな影響を与える．coalesced access とは簡単にいうと，グルーバルメモリからマルチプロセッサのキャッシュまたは shared memory（後述）にデータを転送する際に，複数番地のデータを一括して転送する方法である．一方，non coalesced access と呼ばれる場合はデータが一括して送れず，番地ごとのデータが個別に転送することになり，当然個別に送る分のオーバーヘッドが生じ，時間がかかってしまう．グリッド上の次の時刻の値を計算する際には，FDTD 法，CIP 法いずれも前の時刻の値を参照する必要があるが，この時，coalesced access であることが高速化では必須である．

FDTD 法や CIP 法で coalesced access を実現するために，メモリのアクセス番地が 64bit の整数倍である必要があるため，ブロック内のスレッドの配置を単精度の場合は 16 の整数倍，倍精度の場合は 8 の整数倍とすることが重要である．

・共有メモリの容量

SM は 1MP あたり 16KB であるため，SM を利用する場合はこの制限を考慮する必要がある．

以上の 5 個の項目を満たす最良の条件が最適なスレッド配置と言える．ブロック内のスレッド数を Nx × Ny とすると，特に SM を利用する場合は，

（アクティブブロック数）×（Nx × Ny）×（1 スレッドで SM を利用する変数の数）×4B（単精度）＜ 16KB

を満たすことが条件の 1 つとなる．ただし，FDTD 法では 1 つのグリッドの音圧を計算するためにその両側の粒子速度の値を使うため，1 ブロックの計算領域よりも 1 行余計に SM に確保する必要があるため，その考慮が必要となる．このようにそれぞれの手法により，スレッド配置のモデルは調整することが求められることもある．本研究では検討の結果，FDTD 法では Nx × Ny =32×4 とすることが最速であることがわかった．

次に，共有メモリの利用について述べる．SM はブロック内だけで共有されるため，プログラミングはブロックを意識する必要がある．また，GM から SM へのデータ転送は複数データを一括して転送するため，ブロックで参照する GM のアドレスは連続である方が高速に転送できる．

3-4 GPU による高速化

以下では 2 次元音場領域の FDTD 法と GCIP 法について計算を行い，計算時間の比較を行う．本計算では GPU として GeForce GTX 295 を使用している．また，比較のための CPU の計算環境は CPU ： Intel Core i7 920 2.67GHz を用いて実行する．Core i7 は 4 コアを有し，Hyper-Threading Technology を実装しており，最大で同時 8 スレッドの処理が可能である．本計算では領域は正方形の領域として，吸収境界条件は除いた領域

のみで評価を行っている．

図 19 に GPU を用いて FDTD 法で計算を実行した場合の計算時間を示す．ただし，計算には単精度型を用いている．また横軸は解析領域内のグリッド総数を示している．同図には GPU の数を 1，2，4，8 と変化させた 4 つの場合の結果を示している．同図より，グリッド数が少ない場合は GPU 間のデータ転送によるオーバーヘッドの影響が顕著になるが，グリッド数の増加に伴いほぼ線形に計算時間が増加している．ある程度，計算領域が大きくなるとマルチ GPU の効果が現れることがわかる．

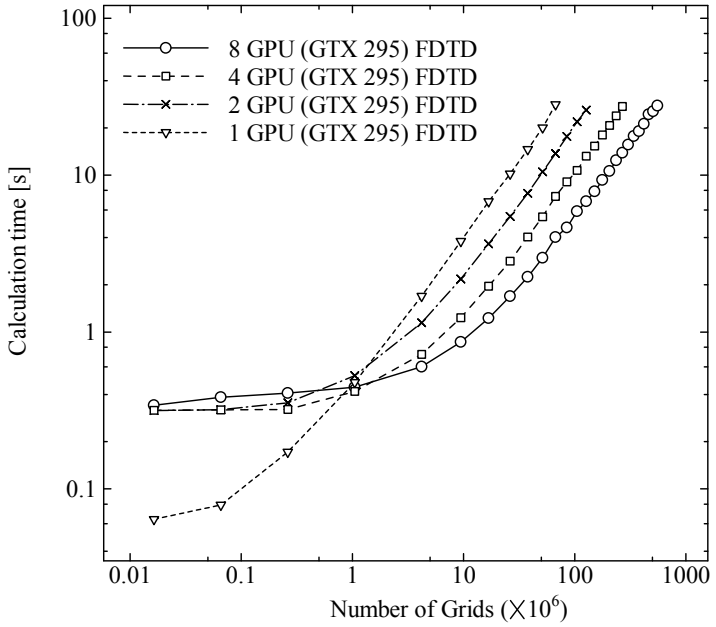

図 19　計算時間比較（FDTD，単精度型，8 GPU）

図 20 に GPU を用いて GCIP(3,1)法で計算を実行した場合の計算時間を示す．ただし，計算には単精度型を用いている．横軸は解析領域内のグリッド総数を示している．FDTD 法の場合と同様に，グリッド数が少ない場合は GPU 間のデータ転送の影響が顕著になるが，ある程度計算領域が大きくなるとマルチ GPU の効果が現れる．FDTD 計算と GCIP 計算の解析領域サイズが異なるのは，それぞれの手法で必要なメモリが違うことによるものである．

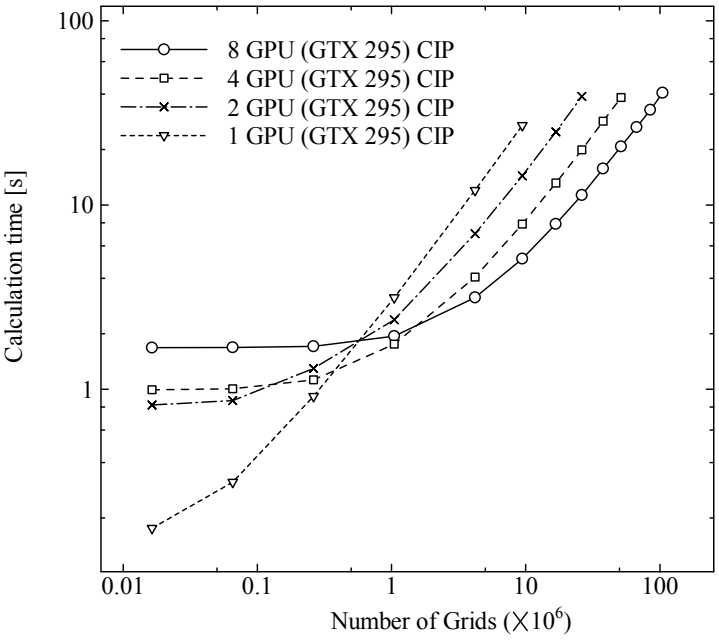

図 20　計算時間比較（CIP，単精度型，8 GPU）

表 2 に計算領域のグリッド数を 8192×8192，計算回数を 1024 と固定した時の FDTD 解析について，マルチ GPU 計算を実行した場合の計算時間を示す．ただし計算には単精度型を用いている．また，CPU については OpenMP を用いて 8 スレッド並列化をした結果を示している．同表より，最も計算時間が少ないのは，8 個の GPU（GTX 295）を並列化した結果であり，FDTD 法では 8 スレッドの CPU の約 44 倍の高速化が実現できている．8GPU の場合では，1 秒あたりに計算することができる音場（のグリッド数）は 17.1 GFUPS:Field Update Per Seconds とスーパーコンピュータ並の値となっている．

表 2　各 GPU での計算時間（単精度型）

計算環境	FDTD
8 thread i7(CPU)	177.9 s (0.39 GFUPS)
GTX 295 (1 GPU)	28.18 s (2.44 GFUPS)
GTX 295 (2 GPU)	13.74 s (5.00 GFPUS)
GTX 295 (4 GPU)	7.326 s (9.38 GFUPS)
GTX 295 (8 GPU)	4.028 s (17.1 GFUPS)

最後に計算領域のグリッド数を 3072×3072，計算回数を 1024 と固定した時の GCIP(3,1)解析及び GCIP(7,1)解析について，マルチ GPU 計算を実行した場合の計算時間を表 3 に示す．同表より，GCIP(3,1)法では 8 スレッドの CPU の約 80 倍，GCIP(7,1)法では約 103 倍の高速化が実現できている．さらに，CPU では 1.36 倍の違いがあった GCIP(3,1)と GCIP(7,1)の計算時間だが，8GPU ではその違いが 1.06 倍となっており，GCIP(7,1)法の GPU 実装の有効性が明らかである．

以上，マルチ GPU 並列計算による時間領域の音響数値シミュレーションを行い，高速化の検討を行った．その結果，8 個の GPU を用いることで CPU に比べて数 10 倍以上の高速化ができることがわかった．また，GPU 計算によって FDTD 法に比べて GCIP 法の計算時間が大幅に減少することがわかり，GCIP 法の GPU 実装の有効性が明らかとなった．

表 3　各 GPU での計算時間（単精度型）

計算環境	GCIP31	GCIP71
8 thread i7(CPU)	409.7 s	559.1 s
GTX 295 (1 GPU)	27.04 s	27.16 s
GTX 295 (2 GPU)	14.40 s	14.55 s
GTX 295 (4 GPU)	7.878 s	8.052 s
GTX 295 (8 GPU)	5.107 s	5.434 s

〈発　表　資　料〉

題　　名	掲載誌・学会名等	発表年月
FDTD 法と CIP 法を用いたハイブリッド音響シミュレーションに関する検討	信学技報 US2010-15	2010 年 6 月
媒質境界を考慮した GPU 音響シミュレーション	音響学会発表会予稿集	2010 年 9 月
CUDA と OpenGL を用いた音響数値解析の高速可視化に関する検討	信学技報 US2010-114	2011 年 2 月
見やすい・わかりやすい 3 次元音響伝搬の可視化方法 ― PMCC （Permeable Multi Cross-section Contours）の提案と評価	音響学会発表会予稿集	2011 年 3 月
GPU 計算を用いた時間領域電磁界数値解析の高速化性能に関する比較	信学論	2011 年 3 月