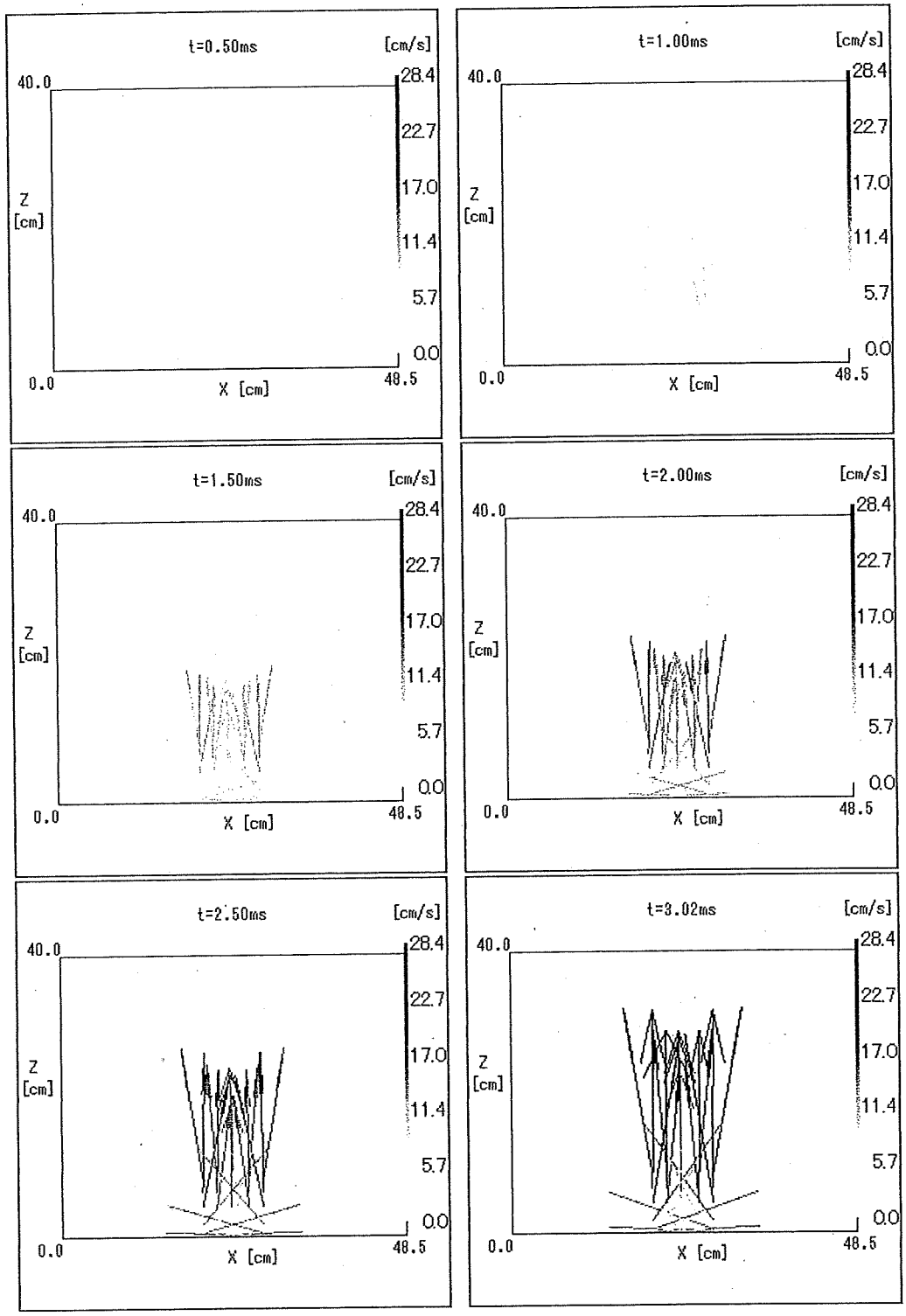

Fig. 5.5 Time history of velocity distribution

5.4 まとめ

MPI による乱流拡散計算のまとめを以下に示す.

・ MPI を用いて並列計算を行うことができた.

・ 並列計算により全体の計算時間を短縮することができ，特に並列化部では約 1.25 倍の高速化を行うことができた.

・ 並列化部の時間短縮量と全体の計算時間の短縮量が異なるため，MPI の最適化を行い，計算時間の無駄な部分をなくす必要がある.

・地面近傍の格子間隔を小さくとるならば，時間刻み幅も小さくとらざるをえず，大規模スケールの計算を行うためには壁近傍の格子を大きくとる必要がある.

・ 並列化により大きく時間の短縮，計算負荷の低減を行えたが，全体の計算時間の大部分を占める Poisson 方程式の計算をさらに高速化することが必要である.

6. 格子間隔を大きくした乱流の計算

地面付近で乱流エネルギーおよび乱流エネルギー散逸率を正しく捉えるために格子点を細かくとると時間刻み幅が非常に微小な値に制限されてしまい，全体の時間を進めるのに莫大な時間がかかってしまう．そこで，本研究では地面付近の格子を細かくとらずに比較的大きな時間刻み幅で計算を行うことができるかその可能性を調べた．以下に計算結果を示す．

6.1 計算条件

計算条件を Table 6.1 に示す．
計算格子は等間隔格子を用いた．

Table 6.1 Calculation conditions

	拡散
格子点数	33 点×33 点×33 点
計算領域	32cm×32cm×32cm
水素貯蔵領域	5cm×5cm×5cm
水素貯蔵量	10.237mg
格子間隔	1.0cm

乱流エネルギー $\tilde{k}$，乱流エネルギー散逸率 $\tilde{\varepsilon}$ の初期条件は，地面上で $\tilde{k} = 0$ [cm²/s²]，$\tilde{\varepsilon} = 0$ [cm²/s³]とした．

開放空間で $\tilde{k} = 10$ [cm²/s²]，$\tilde{\varepsilon} = \dfrac{\tilde{k}^{1.5}}{l}$ [cm²/s³]，$l = 5.0$ [cm]とした．

6.2 拡散の計算結果および考察

水素の質量分率分布の時間変化をFig. 6.1に，速度分布の時間変化をFig. 6.2に，乱流エネルギー分布の時間変化をFig. 6.3に，乱流エネルギー散逸率分布の時間変化をFig. 6.4に，圧力分布の時間変化をFig. 6.5に示す．

水素塊は鉛直上方向に急速に加速しながら上昇している．速度は時刻 121ms 時には最大427cm/s まで加速され，微小領域で計算した結果とは異なり，水平方向への拡散よりも浮力による上方への移動のほうが非常に顕著である．これは水平方向への拡散が進む前に浮力による加速が進むためである．微小領域での計算に比べて水素の貯蔵量が多く，浮力の効果がなくなるまで濃度が薄くなるのに時間がかかるため，それにより大きな加速が行われている．また，水素塊の急速な加速により水素塊と空気の水平方向の境界付近で乱流エネルギーの大きな増加が見られる．微小領域での計算において水素塊と空気の鉛直方向上部の境付近で乱れが発生していたのとは異なり，乱れは水素塊の上方への速い流れにより水平方向での速度勾配が大きくなり，増加が顕著になったと考えられる．乱流エネルギー散逸率についても微小領域での計算とは異なり，水素塊の最も速度の大きな部分よりやや上部で最大値をとり，分布は微小領域計算ではほぼ球状に分布していたのに対し乱流エネルギーと類似した分布となっている．

このように格子幅を大きくとり全体の時間を大幅に進めることができたが，計算を進めていくと計算が止まった．さらに地面近傍の格子を大きくしたために，乱流エネルギーと乱流エネルギー散逸率の値が地面と地面に隣り合う格子で不連続面となり，圧力分布も時間とともに振動しながら発散へ向かった．振動の原因は，本研究で用いている格子がコロケート格子であるためと考えられるが，計算が止まる原因としては地面近傍の格子幅が大きすぎて地面付近の大きな速度勾配，乱流エネルギー勾配，乱流エネルギー散逸率勾配を精度よく捉えられていないためと考えられる．

大規模スケールの計算を行う際，地面近傍の格子を細かくとるならば時間刻み幅を小さくせねばならず全体の時間を大きく進めるためにはあまりにも計算負荷がかかる．一方，全体の時間を進めるために地面近傍の格子幅を大きくとるならば計算は不安定になりうまく計算できなくなる．従って，地面および壁近傍での格子のとり方に十分注意する必要があり，格子幅を大きくとるためには地面および壁近傍で速度および乱流エネルギー，乱流エネルギー散逸率を近似する必要がある．現在，壁領域での対数則を用いた近似がよく利用されているが，本研究においても対数則を用いることが必要である．

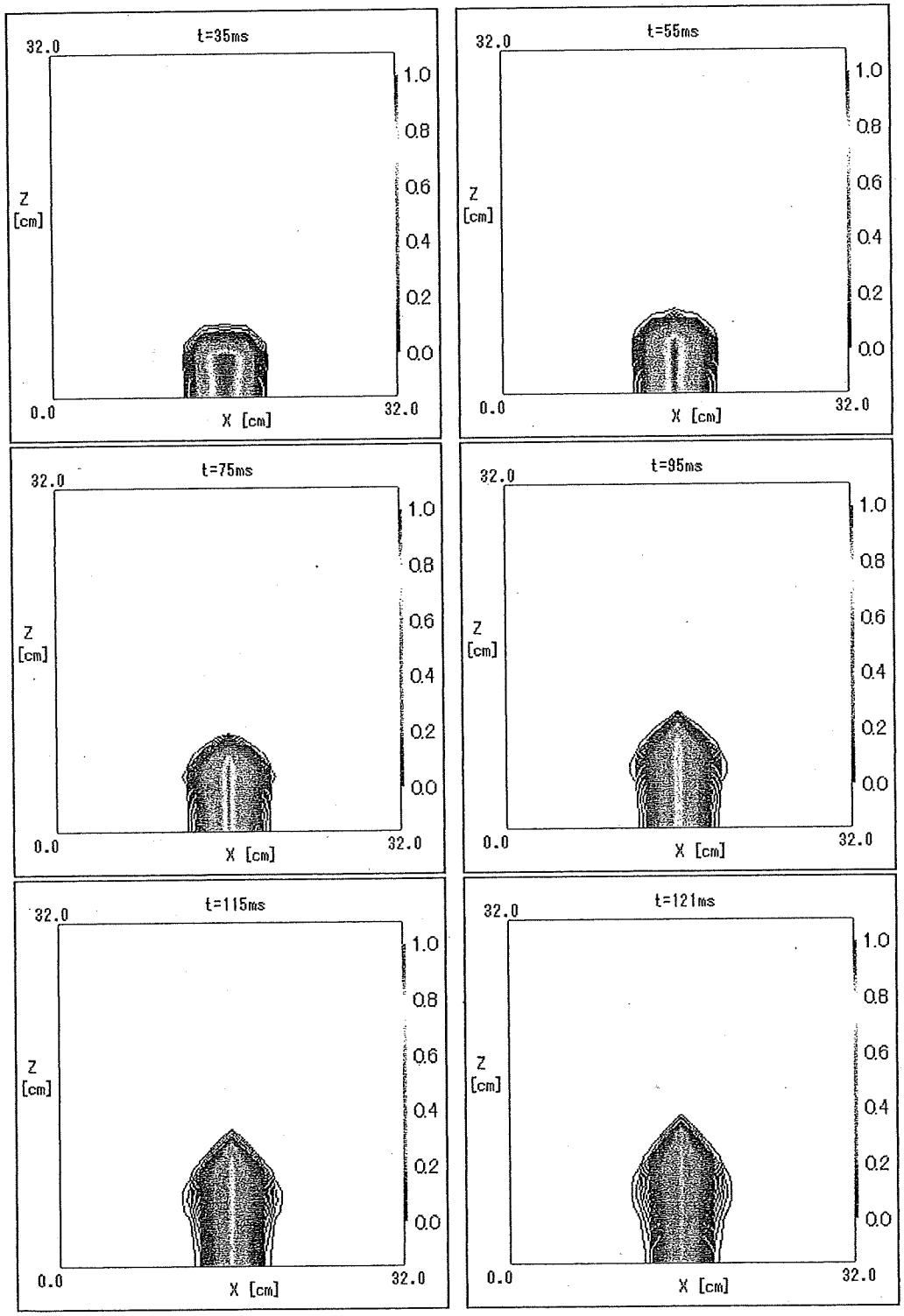

Fig. 6.1 Time history of H_2 mass fraction distribution

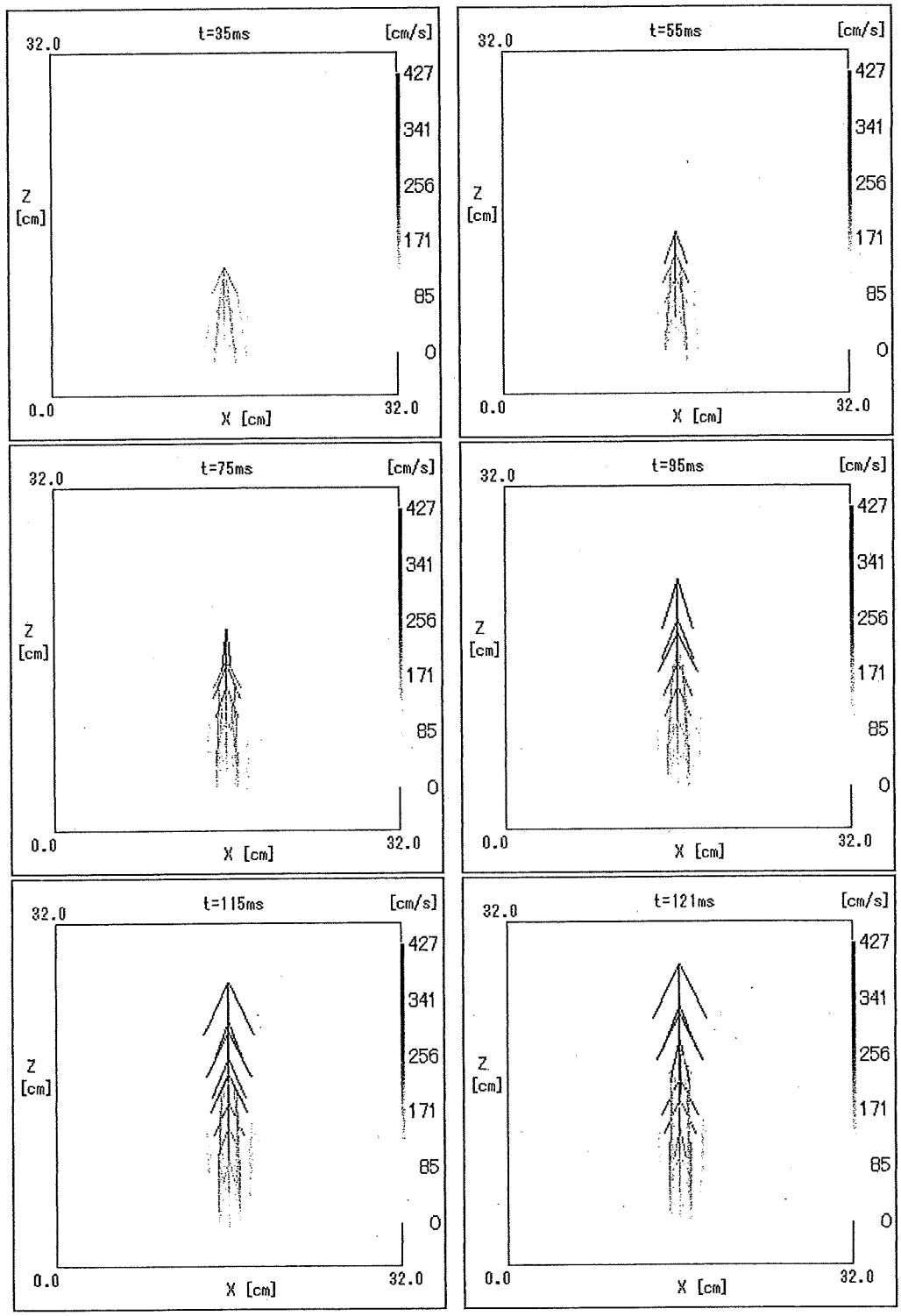

Fig. 6.2 Time history of velocity distribution

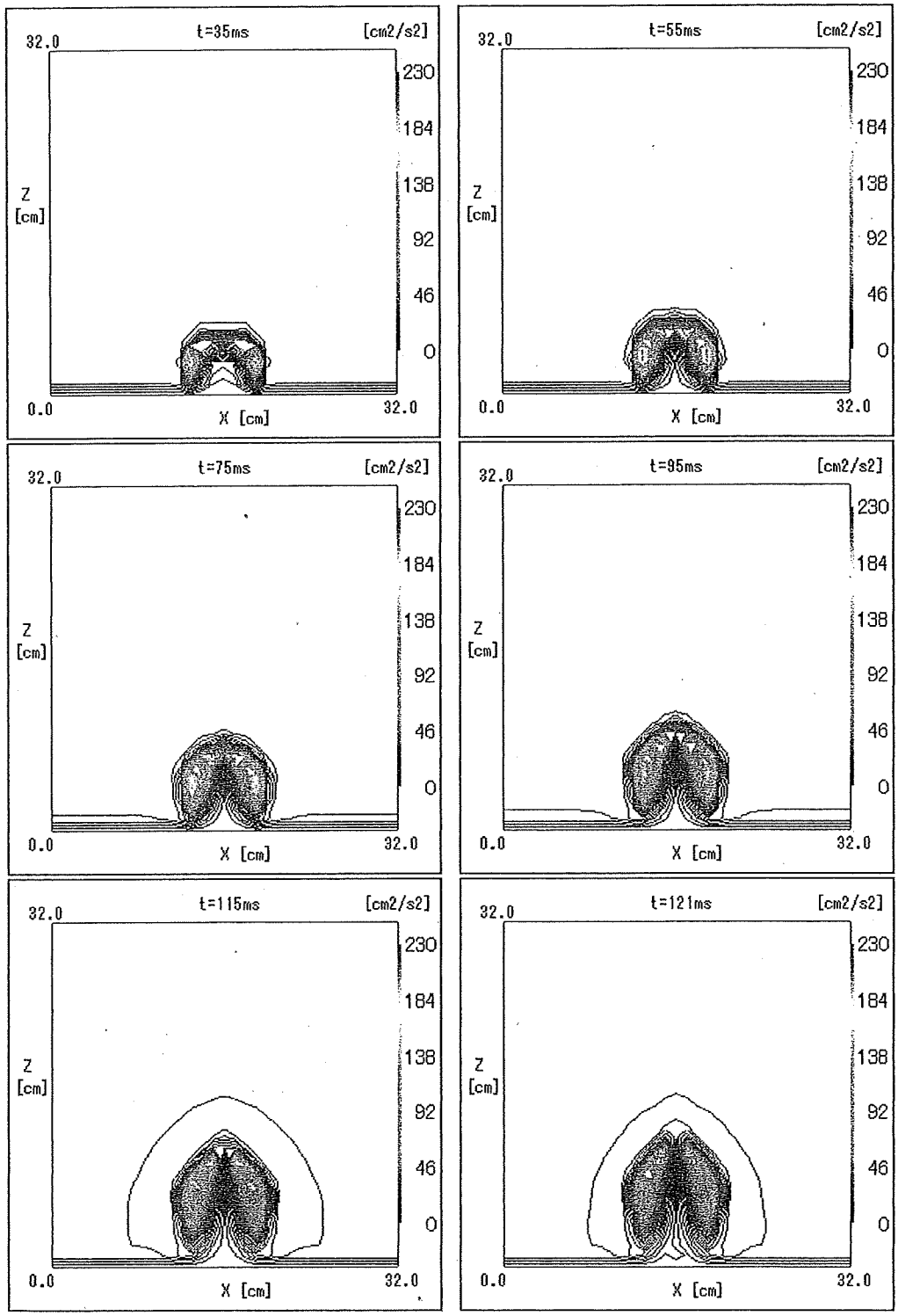

Fig. 6.3 Time history of turbulent energy distribution

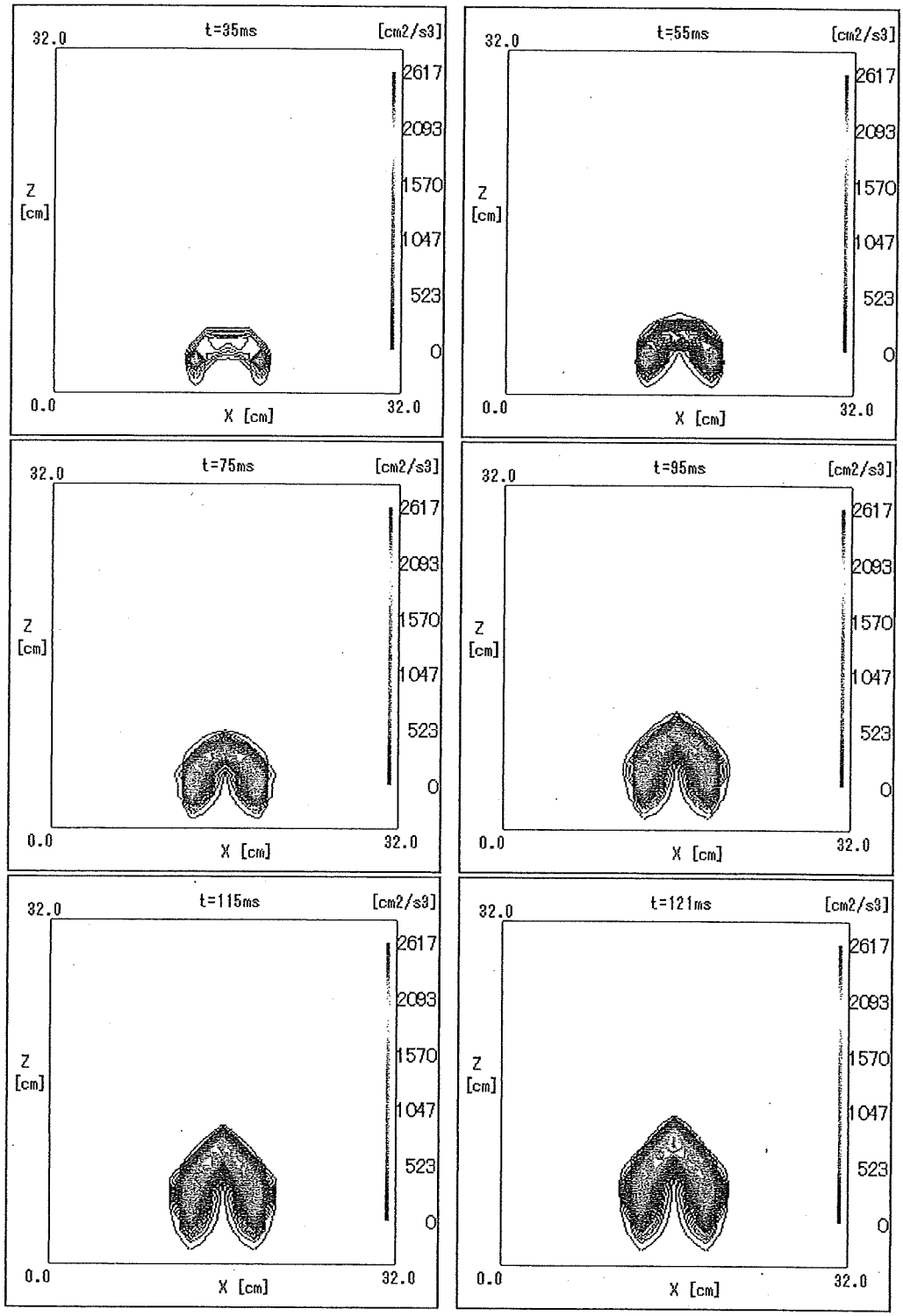

Fig. 6.4 Time history of turbulent energy dissipation distribution

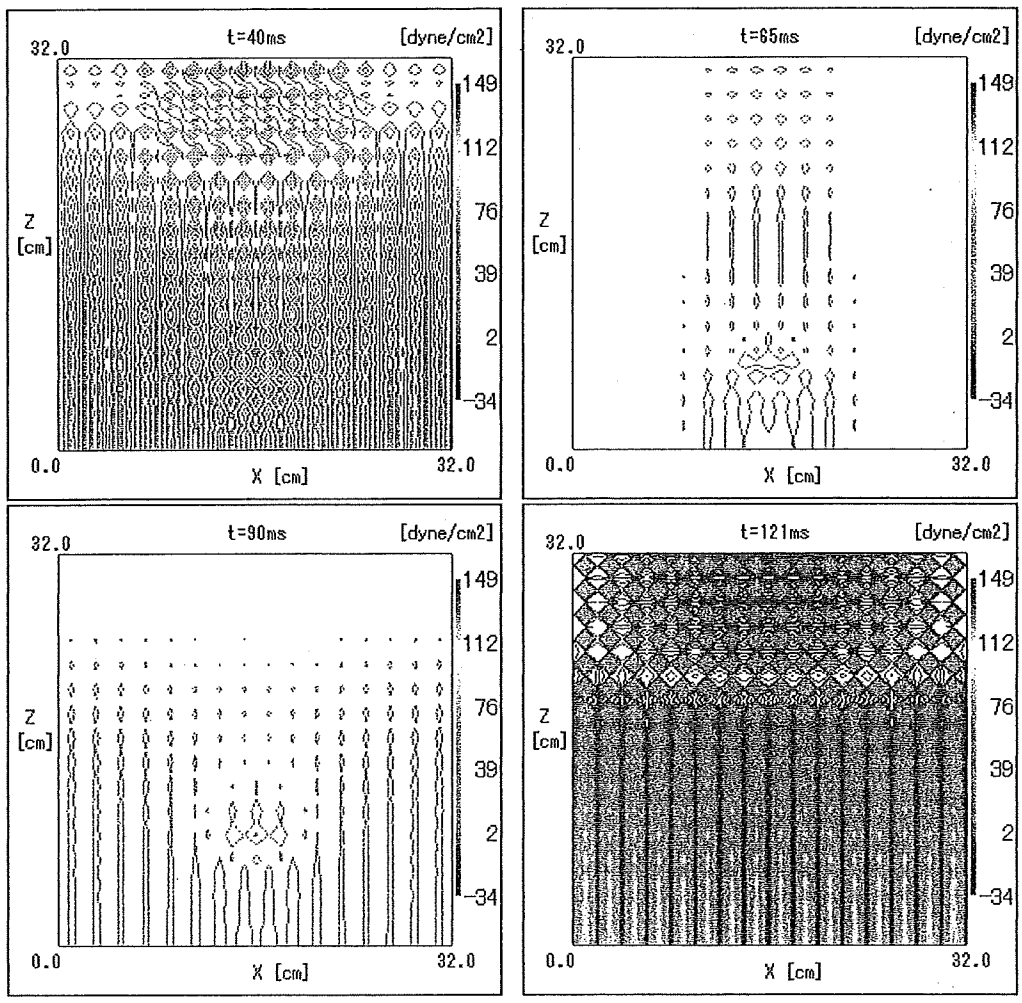

Fig. 6.5 Time history of pressure distribution

6.3 まとめ

格子間隔を大きくした3次元乱流拡散計算のまとめを以下に示す.

・ 時間刻み幅を大きくとるため，地面近傍の格子間隔を大きくとった乱流拡散計算を行った.

・ 全体の時間は大幅に進んだが，圧力が振動しながら発散に向かい，最終的には計算が止まった.

・ 圧力が振動する原因としてコロケート格子を用いていることが一因であると考えられ，スタッガード格子を用いる必要がある.

・ 地面近傍で細かい格子間隔をとることをせず，大きな格子幅で計算を行うためには，壁関数を用いることが必要である.

7. 結言

本研究では，1次元層流拡散・燃焼，3次元層流拡散・燃焼，3次元乱流拡散・燃焼計算コードの構築を行い，水素の大気中拡散・燃焼現象の解析を行った．以下にその結果を示す．

(1)　1次元層流拡散，3次元層流拡散，3次元乱流拡散の計算コードを構築し，水素が大気中を上昇しながら拡散していく現象をシミュレーションすることができた．
(2)　1次元層流拡散・燃焼，3次元層流拡散・燃焼，3次元乱流拡散・燃焼の計算コードを構築し，水素拡散中の任意の時間に外部熱源によりエネルギーを与え，着火・燃焼する現象をシミュレーションすることができた．
(3)　大規模スケール計算に向けて計算コードの並列化を行い，計算時間を短縮させることができた．
(4)　計算時間の大部分はPoisson方程式の解法に依存し，さらなる計算負荷の低減のためにはPoisson方程式の高速解法が望まれる．
(5)　乱流計算において地面近傍では非常に細かい格子幅をとらざるを得ず，そのため，時間刻み幅が制限を受け全体の時間を進めるのに非常に計算負荷がかかる．壁近傍を壁関数などで近似させ，最小格子間隔を大きくとることが必要である．
(6)　圧力振動を抑えるため，スタッガード格子を用いる必要がある．

以上のような結果を得たが，計算領域数m×数m×数mのような大規模スケールの計算をより低計算負荷でかつ少ない計算回数で行うためには，壁近傍においても格子間隔を大きくとらなければならないことがわかった．そのためには，壁上まで格子を配置せず乱流境界層の近似を用いることが必要である．一般的に用いられている壁関数はある限られた範囲の乱流流れしか扱えないため，本研究のような無風に近い状態やほとんど境界層ができないような流れにおいては適用できない．あらゆる状況下においても用いることのできる壁関数を用いることが望ましい．

今後は超並列計算による大規模スケールの計算および燃焼計算によるデータから圧力波の計算、実験との比較が課題である．

8. 参考文献

[1] 文部科学省 科学技術政策研究所 科学技術動向センター,図解 水素エネルギー最前線（工業調査会，2003）

[2] 斉藤 寛泰ほか，第42回燃焼シンポジウム講演論文集，55-56.

[3] 日本機械学会，燃焼の数値計算（2001），59-86,131-207.

[4] A.L.Sanchez, G.Balakrishnan, A.linan, F.A.Williams, Combustion and Flame 105 (1996), 569-590.

[5] A.Murty Kanury, Introduction to Combustion Phenomena (1975), 138-139.

[6] R.J.Kee,F.M.Rupley,J.A.Miller,Chemkin-II:A Fortran Chemical Kinetics Package for the Analysis of Gas Phase Chemical Kinetics (SANDIA REPORT)

[7] Harry.A.Dwyer, Notes on the Numerical Solution of the low Mach number Navier Equations (Lecture note)

[8] Harry.A.Dwyer , Numerical Fluid Mechanics and Heat Transfer A: Reaction/Diffusion (Lecture note)

[9] G.D.スミス，電算機による偏微分方程式の解法（サイエンス社，1971），41-45

[10] 荒川忠一，数値流体工学（東京大学出版会，1994）

[11] J.H.Ferziger, M.Peric, Computational Methods for Fluid Dynamics 3rd edition (Springer-verlag), 112-116

[12] 大宮司 久明ほか，乱流の数値流体力学（東京大学出版会，1998）

[13] 大塚 輝人ほか，第42回燃焼シンポジウム講演論文集，313-314.

[14] FUJITSU, MPIプログラミング ～Fortran, C～（2002）

[15] 冨田 豊ほか，初心者のためのFORTRAN77プログラミング 第2版（共立出版株式会社，1995）

[16] 杉江 日出澄ほか，FORTRAN77と数値計算法（培風館，1991）

[17] ラリー・ニーホフ他，入門Fortran90（ピアソン・エデュケーション，2001）

[18] Kenneth Kuan-yun Kuo, Principles of Combustion (Wiley-Intersience, 1986)

[19] Wikkiam H.Press et.al., NUMERICAL RECIPES in FORTRAN 2nd edition (CAMBRIDGE, 1992)

[20] Bernard Lewis, Guenther von Elbe, Combustion, Flames and Explosions of Gases 3rd edition (ACADEMIC PRESS, 1987)

A. ADI (Alternative Direction Implicit) 法

本研究で用いたADI法のアルゴリズムを以下に示す.

ここで, 簡易な拡散方程式

$$\frac{\partial u}{\partial t}=\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2} \tag{A.1}$$

を扱う. たたし格子点数はM×M×Mとする.

今時刻 t=ndt で解が既知であるとする.

この解法では2階の導関数のうちまずz方向だけを第 (n+1/3) 時点の未知のuの格子点値で表した陰差分近似で置き換え, 他の2階導関数を陽差分近似で置き換える.

$$\frac{u_{i,j,k}^{n+1/3}-u_{i,j,k}^{n}}{\Delta t}=\frac{u_{i+1,j,k}^{n}-2u_{i,j,k}^{n}+u_{i-1,j,k}^{n}}{(\Delta x)^2}+\frac{u_{i,j+1,k}^{n}-2u_{i,j,k}^{n}+u_{i,j-1,k}^{n}}{(\Delta y)^2}$$
$$+\frac{u_{i,j,k+1}^{n+1/3}-2u_{i,j,k}^{n+1/3}+u_{i,j,k-1}^{n+1/3}}{(\Delta z)^2} \tag{A.2}$$

Ozに平行な1行にのっているM-1個の格子点のそれぞれに適用すると時刻(n+1/3)dtのこれらの格子点における連立方程式が得られる. つまり未知数の数がM-1個を持つM-1個の独立な連立方程式となる. これを解く. この作業を各zの格子点において行う.

次に第(n+2/3)時点に解を進める. これはy方向の2階導関数を陰差分近似で, 他を陽差分近似し, Oyに平行な列上の各格子点に対応して差分方程式を書き下ろすことで達成される.

$$\frac{u_{i,j,k}^{n+2/3}-u_{i,j,k}^{n}}{\Delta t}=\frac{u_{i+1,j,k}^{n}-2u_{i,j,k}^{n}+u_{i-1,j,k}^{n}}{(\Delta x)^2}+\frac{u_{i,j+1,k}^{n+2/3}-2u_{i,j,k}^{n+2/3}+u_{i,j-1,k}^{n+2/3}}{(\Delta y)^2}$$
$$+\frac{u_{i,j,k+1}^{n+1/3}-2u_{i,j,k}^{n+1/3}+u_{i,j,k-1}^{n+1/3}}{(\Delta z)^2} \tag{A.3}$$

第(n+1)時点に解を前進させる. これはx方向の2階導関数を陰差分近似し, その他を陽差分近似する.

$$\frac{u_{i,j,k}^{n+1}-u_{i,j,k}^{n}}{\Delta t}=\frac{u_{i+1,j,k}^{n+1}-2u_{i,j,k}^{n+1}+u_{i-1,j,k}^{n+1}}{(\Delta x)^2}+\frac{u_{i,j+1,k}^{n+2/3}-2u_{i,j,k}^{n+2/3}+u_{i,j-1,k}^{n+2/3}}{(\Delta y)^2}$$
$$+\frac{u_{i,j,k+1}^{n+1/3}-2u_{i,j,k}^{n+1/3}+u_{i,j,k-1}^{n+1/3}}{(\Delta z)^2} \tag{A.4}$$

以上により新しい時間のuが求められる.

B. SOR (Successive Over-Relaxation) 法

本研究で用いたSOR (Successive Over-Relaxation) 法について以下に示す.
ϕ に関するPoisson方程式を離散化すると次のようになる.

$$A_P\phi_{i,j,k} + A_e\phi_{i+1,j,k} + A_w\phi_{i-1,j,k} + A_n\phi_{i,j+1,k} + A_s\phi_{i,j-1,k} + A_t\phi_{i,j,k+1} + A_b\phi_{i,j,k-1} = B$$

上式を (i, j, k) 点に関して解くと

$$\phi_{i,j,k} = \left\{B - A_e\phi_{i+1,j,k} - A_w\phi_{i-1,j,k} - A_n\phi_{i,j+1,k} - A_s\phi_{i,j-1,k} - A_t\phi_{i,j,k+1} - A_b\phi_{i,j,k-1}\right\} \Big/ A_P$$

これを領域内部の全格子点について繰り返し計算を行う. 通常, この計算において収束を早め, また発散するのを防ぐために, 繰り返し1回前の値をZに記憶しておき, 過緩和係数rpを用いて以下のようにする.

$$\phi_{i,j,k} = Z + rp \times \left\{\phi_{i,j,k} - Z\right\}$$

収束判定として誤差 $RSDU = \phi_{i,j,k} - Z$ の最大値がある誤差範囲CCより小さくなるまで反復計算を行う.

C. 多重格子法

C.1 多重格子法の特徴

偏微分方程式の楕円型の境界値問題において, その行列式を直製解くよりも, 適当な緩和法を利用することによって効率よくその近似解を求めることができる. ここで, 緩和の途中で真の解からの誤差を早く収束させるために, 格子点数の少ない粗い格子に移り, 近似解の修正量を短時間で計算し, それを真の解に近づけることができる. この方法をさらに粗い格子へ回帰的に適用して, 緩和計算を加速する方法が多重格子法である.

誤差の周波数の観点から述べる, 緩和法などの反復計算では, 計算のはじめの数回の反復で方程式の残差が急速に小さくなり, 解は収束する傾向にあるが, その後は収束が遅くなる. これは, 緩和法が, 格子の大きさと同程度の波長の誤差を最も効率よく減衰させる性質を持つのに対し, 長波長の誤差に対しては効率が悪い性質があるからである. そのため波長の大きな誤差を減衰させるためには多くの反復回数を必要とする. 多重格子法では, 誤差の様々な周波数成分を, 各々に対応した粗い計算格子を用いて, 効率よく減衰させることができる.

C.2　アルゴリズム

以下に多重格子法のアルゴリズムを示す.
一般に Poisson 方程式を離散化すると次のように記述できる.

$$\mathbf{L}\boldsymbol{\varphi} = \mathbf{F} \tag{C.1}$$

ここで，階層的な格子G^kを用いる．添え字kは格子の細かさを表し，この値が小さくなるにつれて格子は粗くなる．G^kの格子間隔をh^kとすると$h^{k-1} = 2h^k$である．ここでは，もっとも簡易な 2 段階多重格子法について述べる．(C.1) を細かい格子G^k上で離散化すると次のようになる.

$$\mathbf{L}^k\boldsymbol{\varphi}^k = \mathbf{F}^k \tag{C.2}$$

適当な初期値を仮定し，(C.2) を繰り返し法等により解くと，近似解$\boldsymbol{\varphi}^k{}_1$を得る.
この段階における各格子点での残差$\mathbf{R}^k$は次のようになる.

$$\mathbf{R}^k = \mathbf{F}^k - \mathbf{L}^k\boldsymbol{\varphi}^k{}_1 \tag{C.3}$$

さらに近似解の修正量を次のように定義する.

$$\mathbf{v}^k = \boldsymbol{\varphi}^k - \boldsymbol{\varphi}^k{}_1 \tag{C.4}$$

(C.2)，(C.3)，(C.4) より細かい格子上での修正方程式を得る.

$$\mathbf{L}^k\mathbf{v}^k = \mathbf{R}^k \tag{C.5}$$

多重格子法では，低周波成分の誤差を粗い格子で減衰させるために，(C.5) を粗い格子に制限補間してから解く．残差の制限補間および粗い格子上での修正方程式を以下に示す.

$$\mathbf{R}^{k-1} = \mathbf{I}_k^{k-1}\mathbf{R}^k \tag{C.6}$$

$$\mathbf{L}^{k-1}\mathbf{v}^{k-1} = \mathbf{R}^{k-1} \tag{C.7}$$

$\mathbf{I}_k^{k-1}$は細かい格子から粗い格子への変換を行う制限補間演算子である.

(C. 7) を解くことにより修正値 $\mathbf{v}^{k-1}$ を得る.
細かい格子上での修正量 $\mathbf{v}^{k}$ は $\mathbf{v}^{k-1}$ を延長補間して得ることができる.

$$\mathbf{v}^{k} = \mathbf{I}_{k-1}^{k}\mathbf{v}^{k-1} \tag{C. 8}$$

$\mathbf{I}_{k-1}^{k}$ は粗い格子から細かい格子への変換を行う延長補間演算子である.

これより $\mathbf{L}^{k-1}$ は次のように表せる.

$$\mathbf{L}^{k-1} = \mathbf{I}_{k}^{k-1}\mathbf{L}^{k}\mathbf{I}_{k-1}^{k} \tag{C. 9}$$

この過程で得られた修正量 $\mathbf{v}^{k}$ を近似解 $\boldsymbol{\varphi}^{k}{}_{1}$ の修正として用いる.

$$\boldsymbol{\varphi}^{k}{}_{2} = \boldsymbol{\varphi}^{k}{}_{1} + \mathbf{v}^{k} \tag{C. 10}$$

さらに $\boldsymbol{\varphi}^{k}{}_{2}$ を用いて緩和計算を行う.
以上のサイクルをある誤差範囲に収束するまで繰り返し計算を行う手法を 2 段階多重格子法と呼ぶ. さらに修正方程式 (C. 7) を解くために, さらに粗い格子を用いて, 再帰的に上述のサイクルを適用し, 何重にも修正方程式を構成していく手法が多段階多重格子法になる.

C. 3 制限補間演算子

本研究では最も簡単で効果的な制限補間演算子 7 点 scheme を用いた.

$$R^{k-1}{}_{i,j,k} = \frac{1}{12}(R^{k}{}_{i-1,j,k} + R^{k}{}_{i+1,j,k} + R^{k}{}_{i,j-1,k} + R^{k}{}_{i,j+1,k} + R^{k}{}_{i,j,k-1} + R^{k}{}_{i,j,k+1} + 6R^{k}{}_{i,j,k}) \tag{C. 11}$$

C. 4 延長補間演算子

本研究で用いた延長補間演算子は双一次展開である. 3 次元において密な格子には 4 種類ある. 粗い格子点と一致する点は一致する点の値が与えられる. 二つの粗い格子の点につながっている線上の点は二つの粗い格子の点の値の平均で与えられる. 四つの粗い格子の点が形成する面上の真ん中に位置する点は四つの隣り合う点の平均で与えられる. 最後に八個の粗い格子の点が形成する格子体積の真ん中に位置する点は同じように八個の点の平均値で与えられる.

D. Message Passing Interface (MPI)

D.1 MPIについて

Message Passing Interface (MPI) は分散メモリ型並列計算における，メッセージパッシングをパラダイムとする並列プログラミングのための標準的な Application Programming Interface (API) を提供するライブラリである．
MPI を用いることにより，可搬性の高い並列プログラミングが可能となる．また，本来は並列計算機用であったが，その後ワークステーションまたはパーソナルコンピューターを用いたクラスタシステムが普及するにあたり，それらを並列計算機システムとして扱うことを可能にする最も有力な手法として重要視されている．

D.2 メッセージパッシングに基づく並列プログラム

分散メモリ型並列計算機（クラスタシステムもこれに含まれる）では，各プロセッサ上でそれぞれプロセスを走らせ，それらがメッセージ交換を行うことにより協調的な並列処理を進めるというプログラムパラダイムが一般的である．ここでは，各プロセスは独立のアドレス空間をもつため，変数を共有した並列プログラミングは行えない．プロセス群は処理を進めるために，適宜，各々のデータを他のプロセスに送信したり，他のプロセスから必要なデータを受信したりして処理を進める．

D.3 メッセージパッシング

MPI プログラム中で各プロセスはメッセージを交換しあう．メッセージ交換には 2 つの種類がある．1 つ目は point-to-point 通信で，これは一対のプロセス間で，片方が送信者，もう片方が受信者として，それぞれ送受信のための関数を実行し，通信を行うものである．2 つ目は collective 通信と呼ばれるもので，これは 2 つ以上の多数のプロセスが関係し，それら全てを含んだ協調的な通信が行われる．

注意点としてすべてのメッセージは送受信の関係が成り立ち，論理的に矛盾がないようにしなければならないという点である．メッセージパッシングに基づく並列プログラミングにおいて最も重要な点であり，またバグが入りやすい部分である．

D.4 デッドロック

メッセージパッシング型のプログラムでは，デッドロック (deadlock) が生じやすい．これは，複数のプロセスが，相手の起こす何らかのイベントを互いに待ちあって，結果的に全

体の処理が停止してしまうことを言う．デッドロックはプログラマの不注意やプログラム全体の論理構造の見誤りによって生じる．

謝辞

本研究は，厚生労働科学研究費補助金 労働安全衛生総合研究事業として，平成 14 年度から 16 年度までの 3 年間にわたり，支援を受けて遂行されたものである．
厚生労働省労働基準局安全衛生部企画課の科学研究費担当の皆様には，事務連絡や報告書等の作成に際して，多くの御支援を受けた．ここに謝意を表す．
中央労働災害防止協会からは，平成 14 年度に若手研究者育成活用事業リサーチ・レジデント(名古屋大学大学院工学研究科助手(現在)斎藤寛泰)，平成 15 年度は，外国への日本人研究者派遣事業(産業安全研究所　大塚輝人)と外国人研究者招へい事業(カナダマクギル大学 J. H. Lee 教授)の支援を受けた．ここに謝意を表す．
又，本研究の開始時には研究グループに参加されたが，途中の人事異動に伴い，残念ながら，研究グループから離れられた方々にも謝意を表す．松井英憲氏(平成 14 年度産業技術安全研究所から参加，平成 15 年度から産業安全研究所理事）と村上桂一氏（平成 14 年度名古屋大学から参加，平成 15 年度から宇宙航空研究開発機構）には，多大な御尽力を頂いた．
又，岩佐樹氏（現在　産業安全技術協会）にも，測定電気回路の製作等で多くの支援を頂いた．謝意を表す．