

文部科学省次世代IT基盤構築のための研究開発
「革新的シミュレーションソフトウェアの研究開発」

RSS21 フリーソフトウェア

HEC ミドルウェア (HEC-MW)

PC クラスタ用ライブラリ型 HEC-MW
(hecmw-PC-cluster) バージョン 2.01

並列可視化ライブラリ マニュアル

本ソフトウェアは文部科学省次世代IT基盤構築のための研究開発「革新的シミュレーションソフトウェアの研究開発」プロジェクトによる成果物です。本ソフトウェアを無償でご使用になる場合「RSS21フリーソフトウェア使用許諾条件」をご了承頂くことが前提となります。営利目的の場合には別途契約の締結が必要です。これらの契約で明示されていない事項に関して、或いは、これらの契約が存在しない状況においては、本ソフトウェアは著作権法など、関係法令により、保護されています。

お問い合わせ先

(公開／契約窓口) (財)生産技術研究奨励会

〒153-8505 東京都目黒区駒場4-6-1

(ソフトウェア管理元) 東京大学生産技術研究所 計算科学技術連携研究センター

〒153-8505 東京都目黒区駒場4-6-1

Fax : 03-5452-6662

E-mail : software@rss21.iis.u-tokyo.ac.jp

目 次

1. 概要	1
1.1 はじめに	1
1.2 解析計算との並行処理	2
1.3 並列可視化技術について	4
1.4 計算機依存の最適化	9
参考文献	11
2. HEC-MW の並列可視化ライブラリの実行方法	12
2.1 ファイルバージョンとメモリバージョン	12
2.2 ファイルバージョンの実行方法	14
2.3 メモリバージョンの実行	16
3. HEC-MW 可視化パラメータ	17
3.1 書式とコントロールパラメータ	17
3.2 PSR モジュールのパラメータ設定	18
3.3 PVR モジュールのパラメータ設定	35
3.4 PSF と PVR におけるミクスチャーパラメータの設定例	44
3.5 PST モジュールのパラメータ設定	49
3.6 エラーメッセージ	58

1. 概要

1.1 はじめに

数値シミュレーションシステムは、通常以下の3つの構成要素からなる。①メッシュ生成、②基礎方程式の求解、③結果の可視化。最近のコンピュータハードウェアやソフトウェアの急速な発展により、詳細な解析を目的として②の求解部で大規模な計算が可能になってきている。それゆえグラフィックスによる結果表示なしでは現象の解析は非常に困難である。そこで強力な可視化プログラムが求められている。

現状の商用可視化プログラムは、比較的小規模なデータには有効に働くが、残念ながら大きなデータについては、メモリ量の制限や並列化が不十分などの理由により十分機能しないことがある。そこで今回、HEC-MW[1]プロジェクトにおいて可視化プログラムを重要な要素と位置づけ、並列可視化ライブラリの開発を行うことになった。これは大規模な解析計算の結果を見やすく効果的に図化し、それにより研究者や技術者が解析結果に潜む物理現象の把握の一助となることを目的とするものである。

本プロジェクトで開発された並列可視化ライブラリは以下の6つの特徴を有する。

- (1)同一の並列計算機内で解析計算と可視化処理を同時並行で行える。
- (2)グラフィックボードなど表示システムに依存しない。
- (3)スカラ、ベクトル及びテンソルなどのデータが持つ意味を詳細に表現するため、種々のテクニックを駆使している。
- (4)可視化自体も高速処理が行えるよう、さまざまな並列計算機（PC クラスタから地球シミュレータまで）に対して並列化を含む最適化を行っている。
- (5)すべてのモジュールは、複雑で大規模な非構造格子に対応している。
- (6)図の質的向上をねらった手法を取り入れている。

1.2 解析計算との並行処理

並列計算機を用いる場合は通常データ規模は非常に大きく数ペタバイトにまで及ぶことがあります。そうした大規模なデータを他のコンピュータに転送したり、ディスク間に保存したりするのは効率が悪いものである。更に、可視化のためにも大きなメモリが必要となりクライアントマシンでそれを行うのも困難である。そこで計算結果を即座に同じ並列計算機上で可視化処理してしまい、比較的小規模になった可視化データとして保持するようにした。この時下記の2つの実行モードが可能である。

① サーバー・クライアントモード

計算サーバーにてユーザーは可視化の内容（等値線、流線やそのパラメーター）を指定する。それに応じた幾何情報データをファイル出力し、実際の可視化はクライアントマシンにて行う。その際ユーザーは AVS などのプログラムを用いて、その段階でシェーディングや視点などのパラメータを指定できる。

この方法をとることにより当初巨大なデータもかなり縮小されるのでメモリ制限が緩和できる。クライアントマシンにて自由性のある可視化ができるなどのメリットがある。

②サーバーモード

このモードは、①と違って可視化処理をすべて計算サーバーで行ってしまうものである。時系列の解析結果データを描画したりアニメーションしたりした可視化データを直接生成し保存する。この可視化データは計算データサイズには依存せず一定である。このモードで生成された可視化データは一切の幾何情報を含んでいないため、単に視点の変更を行うためだけでも再計算が必要である。

極めて大規模なデータを扱うときにこのモードを用いると便利である。

図1に HEC-MW 可視化ライブラリの構成を示す。計算モジュールはメッシュデータと制御パラメータ(計算および可視化両方の)を読み込む。計算を始めから1タイムステップ終了した段階で可視化モジュールがスタートする。おのおののタイムステップ毎にユーザーは、等値線、流線、ボリュームレンダリングなどの描画内容を指定できる。更に描画のパラメータのそれぞれ指定できる。例えば、毎回視点を変えたボリュームレンダリングや異なる値での等値線図などを作成できる。多くのメッシュ点数のデータについて多くのタイムステップがあるデータについて数ヶ月にも及ぶ計算が必要なことがある。その際、計算結果のデータも巨大でそのままでは保存できない場合など、可視化データとして小規模なデータとして保存できることは便利である。更に、計算の途中で解析の様子を描画図で検証し、もし不具合（可視化のパラメータが好ましくないなどを含む）があれば計算を中断しパラメータ変更の後リスタートすることができる。こうして徒に計算資源を浪費する

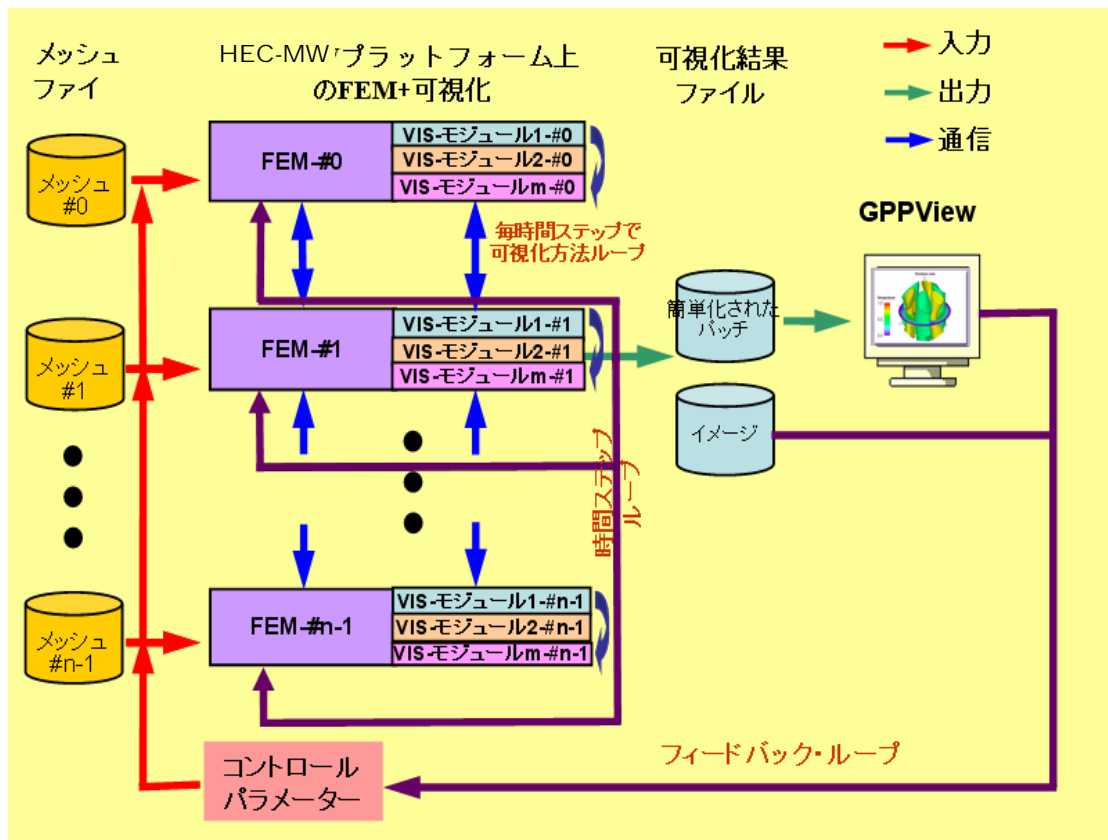


図 1: HEC-MW の並列可視化ライブラリの構成

ことを防げる。

サーバー・クライアントモードにて小規模な試計算を行った後、サーバーモードで本来の目的の計算を行うなどの方法は有効である。

1.3 並列可視化技術について

HEC-MW 並列可視化ライブラリでは、スカラー、ベクトル、テンソル表示のために様々なテクニックの導入を予定している。今回のリリース版では、スカラーデータの表示部が主なものになっている。以下では、初めにリリース版の中の並列ボリュームレンダリングと並列サーフェスレンダリングを紹介する。続いて現在開発中のモジュールについて述べる。

1.3.1 リリース版

(1) 並列サーフェスレンダリング(Parallel Surface Rendering:PSR)

サーフェスレンダリングは3次元可視化において非常に重要な技術である。これにより立体表面のデータ分布を詳細かつ明確に表示することができる。以下に3つの例を示す。

a. 境界表面の表示

境界表面の表示は自動的に行われる。しかし複数個の境界がある時は、それぞれに名前をつけることにより、以降名前指定で表示できる。図2に境界表面の例を示す。

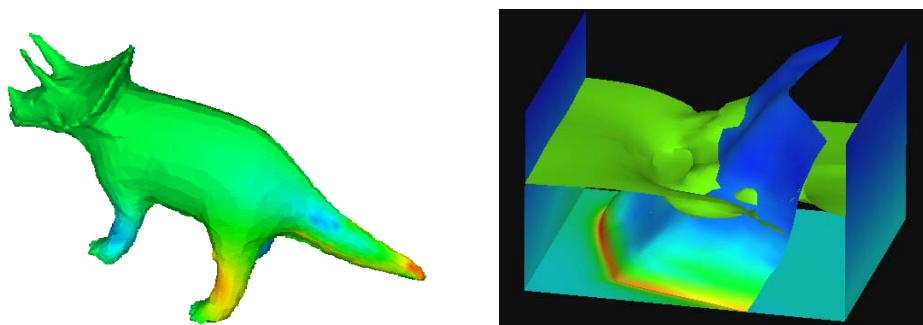


図 2: 境界表面の表示例 (データ提供: 東京大学 奥田氏、RIST 飯塚氏)

b. 等値面

データの値の分布を見るにはこの等値面表示を用いるのが便利である。本プログラムでは複数の等値面を並列に処理できる。図3は6つの等値面と平面の切り口でのデータ分布を示している。

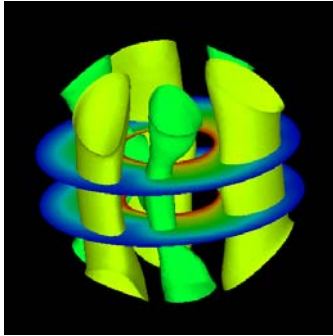


図 3: 6つの等値面と平面切り口でのデータ分布
(データ提供：松井氏).

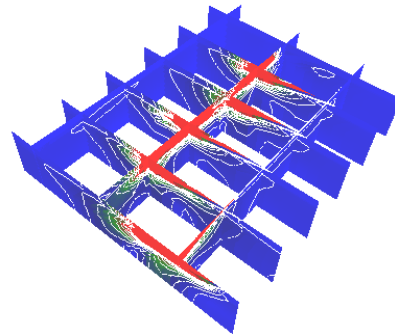


図 4: 西南日本の地震解析、応力分布図
(データ提供：RIST 飯塚氏).

c. 任意曲面の切り口

複数の任意曲面の切り口でのデータ分布を表示する。曲面は方程式で与えられる。図4 は西南日本の地震解析の例である。 .

(2) 並列ボリュームレンダリング (Parallel Volume Rendering:PVR)

ボリュームレンダリングは3次元可視化において非常に強力な手法である。これにより3次元立体の内部までの状況が観測できる[2]。ボリュームレンダリングは、時間的にもデータ容量的にも高コストであるため並列化の導入が極めて重要である。並列ボリュームレンダリング(PVR)は、並列マシンのタイプ、メッシュタイプ、射影の方法などによって分類できる[3]。効率アップが実現している例は参考文献[4-7]。いずれにしてもボリュームレンダリングのプログラムを設計する時には、複雑で大規模なデータを想定しなければならない。PVR ではスーパーボクセルによる領域分割を行いそれぞれで処理を行った後、描画データを統合して1つの描画データとしている。

● スーパーボクセルによる領域分割

HEC-MW で扱うメッシュは、複雑で規模が大きいことが多い。しかも扱う要素は非構造格子で、四面体、六面体、プリズムなどが混在し、その上、多層構造になることもある。こうした複雑なメッシュ構成は描画データ

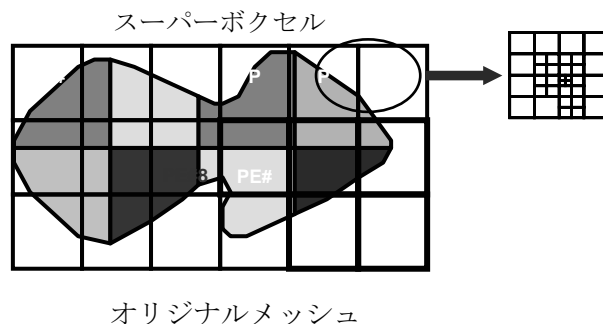


図. 5: Supervoxel-based partition in the PVR module.

の作成処理にとって非常にやっかいなもので、レイトレーシングによるボリュームレンダリングなどでは処理時間が大きくなってしまいます。その対策として並列 BSP という手法が研究開発されている[5][8]。しかしこの手法は中間データを用いるため大きなメモリやディスクスペースが必要となる。そこで極めて大きなデータに対しては、並列処理が効果的となる（高速化の意味だけでなくメモリ、ディスクスペースを大きく取れるため）。この時領域バランスよく分割するためにスーパーボクセルの手法を取り入れている（図5参照）。

スーパーボクセルの定義はデータの座標値から自動生成できるが、ユーザー指定により削除や細分も可能である。

● 分割した描画データの生成

スーパーボクセルによる領域分割後、各プロセスには均等にスーパーボクセルが割り振られる。それぞれのプロセスにおいて通常のレイトレーシング処理を行う。すなわち対象の描画対象となるピクセルを求める。次に視点から各ピクセルに光を当て、その光線とボクセルとの交点を求め、その点での対象データの値を求めカラーマップへの変換を行う。この処理を順次奥の方に進める。こうしてこのスーパーボクセルに対する分割描画データを作成する。

● 分割描画データの統合

それぞれのスーパーボクセルについてピクセル毎の処理をすべて行いその不透明度も合わせて記録する。こうして各プロセスにある分割描画データをその位置関係を考慮しながら統合する。統合に際してはやはり手前から奥のほうにレイトレーシングを進め書くピクセルでのカラーマップを求めていく。そして最終的に統合した描画データを得る。

図6にPVRによるボリュームレンダリングの例を示す。

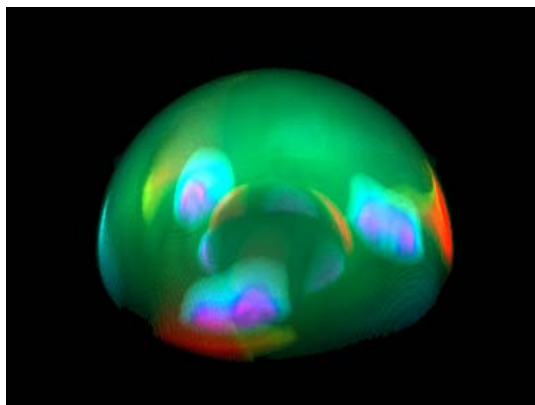


図 6: 半球の熱流体シミュレーションの
PVR による表示
(データ提供：松井氏)

1.3.2 開発中のモジュール

並列粒子追跡、及び並列ストリームラインモジュールを HEC-MW にて開発中である。これにより大規模データのベクトル場などの表示が可能となる。並列パーティクルトレース法によって粒子の挙動を様々なスタイルで表示できる。粒子の初期状態、指定の時刻の

状態、または特定の平面からの流れ出しの様子などを表示できる。図 7 は、地下水流れの

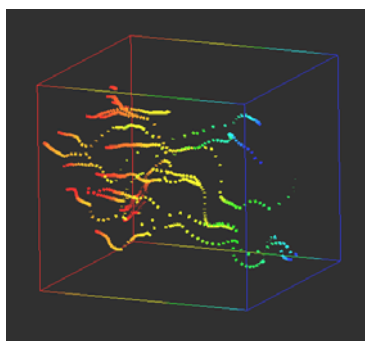


図 7: 並列粒子追跡法による地下水流の表示
(データ提供: RIST 中島氏).

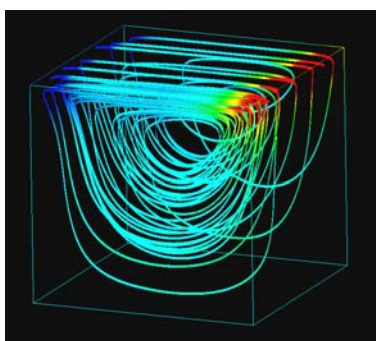


図 8: 照明つきストリームラインによる立方体内の流れ表示
(データ提供: 松井氏)

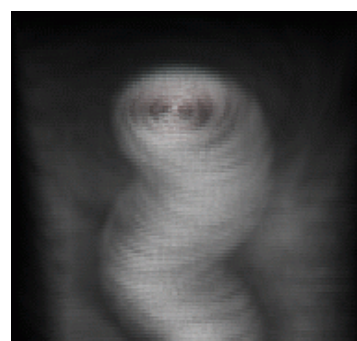


図 9: 重要度マップを考慮した流線畳み込み法による竜巻解析の表示
(データ提供: オハイオ大 Roger Crawfis 氏) .

様子を粒子追跡法にて表示したものである。ストリームラインは3つの表示法がある。(1) 従来の方法。(2)太さをもったストリームラインレンダリングをしたもの[9]。図 8 に例を示す。(3)ストリームラインの太さでその量を表現したもの。

テクスチャベース手法 はベクトル場の表示には強力な手法である[10]。しかし時として3次元の密なベクトル場に適用するとうまく動作しないことがある。そこでこの方法に重要度マップという概念を入れ改良を加えた[11]。ベクトル場での可視化では、渦のような流れの特徴となる領域を調べることがしばしば重要となるので、重要度マップにより選択的に強調描画することが効果的な可視化と考えられる。重要度マップの算定においては流体位相解析技法を用いている。更にこの位相解析技法により、切り口面での等値線の強調表示や不連続のある物理量の表示などが行える。現状 Zockler, et al. [9]のストリームライン照明モデルを用いている。これにより3次元流線畳み込み法の適用の場を増やしている。図 9 に竜巻の解析結果の例を示すが、周辺の地震解析の表示これにより多くの解析情報を得ることができる。

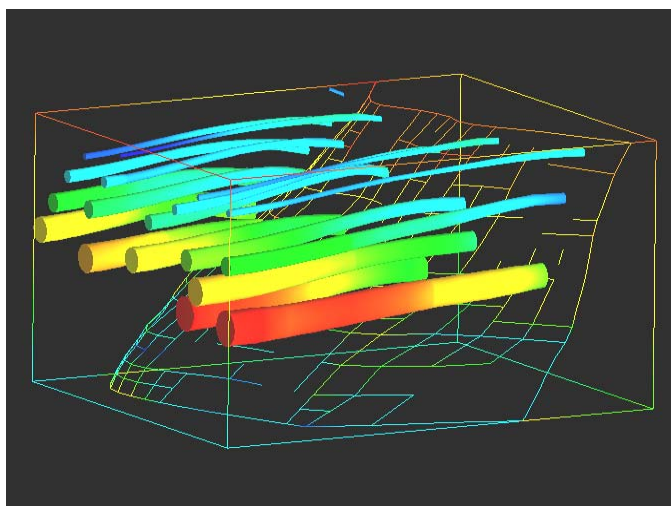


図 10: 並列ハイパーストリーム手法による日本列島

テンソル場の表示にはハイパーストリームライン技法を用いている[12]。これにより 3 次元空間の 2 次テンソル場をパスに沿って表示することができる。それと同時に 9 つのコンポーネント（3 つの固有ベクトル）も表示できる。あるスタートポイントから、最大主値に属する固有ベクトルに沿ったベクトルのトラジェクトリを描く。各ポイントでは楕円形の切り口であるが、その長径と短径が他の 2 つ主値の大きさと方向を示している。また色は最大主値の大きさを示している。図 10 に応力テンソルの例を示す。

1.4 計算機依存の最適化

一口に並列計算機といってもそのハードウェアアーキテクチャは、多種多様である。可視化においても高速化の手法は並列計算機のタイプに依存して手法を選択する必要がある。ある並列計算機にて効果的であっても他のタイプの計算機で効果的であるとは限らない。そこで HEC-MW プロジェクトでは可視化ライブラリについても計算機のタイプ毎の最適化コードの開発を行っている。以下に PVR を例に挙げて、各種のマシンにおける最適化手法を述べる。

1.4.1 スカラ計算機とベクトル計算機の違い

ベクトル計算機は通常スカラー計算機より高速である。しかしベクトル化率が性能に大きく影響する。コンパイラによる自動ベクトル化機能だけでは不十分で、往々にしてベクトル化に注意を払ってコード作成をしなければならない。

データ構造 — 8分木と一次元配列の比較

複雑なメッシュデータに対してPVR処理を高速で行うのは困難である。その理由は、①隣接情報を事前に作成しておく必要がある。②階層構造でデータを扱うのは効果的であるが、非構造格子について階層構造を作ることは難しい。③不規則な形状ではPVRの合成処理が複雑になる。④点の同定が構造化メッシュに比べて非常に困難である。そこで効率よい処理のために直交メッシュへの変換が必要である。しかしながら、同等の精度を維持してこの処理を行うと随分大きなメモリスペースが必要となる。またボリュームレンダリングの処理に時間がかかってしまう。そこで通常は8分木手法による階層型データ構造を採用する。これにより、トータルのボクセルの数を減らしても、内部情報の詳しくさは維持できる（図11(a)）。

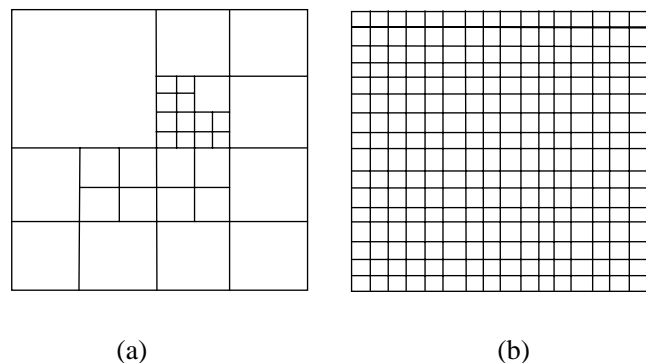


図 11: スカラ計算機、ベクトル計算機での違い

(a) スカラ計算機における 8 分木探索

(b) ベクトル計算機における独立探索

高速化のために更に Branch-on-need Octree 手法を採用している (BONO) [13]。従来型の 8 分木法は基本的にバイセクション法である。そのため空のボクセルが多数生成されてしまうことがある。反面、BONO では分割が必要になった段階で行うので、徒に分割がなされる

ことはない。加えてBONOでは空のボクセルや透明なボクセルを効率的にスキップしながら必要な点での最大値と最小値を求めることができる。この方法においてはまずそれぞれのスーパーボクセルを直交メッシュの中で再同定を行う。続けて隣接するボクセルでの値の差を見て、一定以上大きければボクセルを2分割する作業を再帰的に行う。スカラー計算機においてはこの手法が効果的である。

一方、ベクトル計算機ではBONOはベクトル化されないため効率的ではない。そのため従来型の直交メッシュへの分割を行い1次元配列での処理となるようにしている。これによりベクトル化が実現できる（図11(b)）。

参考文献

- (1) <http://www.fsis.iis.u-tokyo.ac.jp/>
- (2) Levoy, M.: “Display of surfaces from volume data”, *IEEE CG&A*, Vol. 8, No. 3, pp.29-37,1988.
- (3) Wittenbrink, C. M.: “Survey of parallel volume rendering algorithms”, *Proceedings of International Conference on Parallel distributed Processing Techniques and Applications*, Las Vegas, Nevada, pp. 1329-1336, 1998.
- (4) Yagel, R.: “Towards real time volume rendering”, In *Proceedings of GRAPHICON*, Saint-Petersburg, Russia, volume 1, pp. 230-241, 1996.
- (5) Ma, K. L., Painter, J., Hansen, C. D. and M. F. Krogh: “Parallel volume rendering using binary-swap compositing”, *IEEE CG&A*, vol. 14, No. 4, pp. 59-67, 1994.
- (6) Wittenbrink, C. M. and Somani, A. K.: “Time and space optimal parallel volume rendering using permutation warping”, *Journal of Parallel and Distributed Computing*, vol. 46, No. 2, 148-164, 1997.
- (7) Silva, C.: Parallel volume rendering of irregular Grids, *Ph.D. thesis*, State University of New York at Stony Brook, 1996.
- (8) Ramakrishnan, C.R. and Silva, C: “Optimal processor allocation for sort-last compositing under BSP-tree ordering”, *SPIE Electronic Imaging*, pp. 73-80, 1999.
- (9) Zoeckler, M., Stalling, D., and Hege, H.-C.: “Interactive visualization of 3D vector fields using illuminated streamlines”, In *Proceedings of IEEE Visualization’96*, pp. 107-113, 1996.
- (10) Cabral, B. and Leedom, C.: “Image vector field using line integral convolution”, *Computer Graphics Proceedings*, ACM SIGGRAPH, pp. 263-272, 1993.
- (11) Chen, L., Fujishiro, I. and Suzuki, Y.: “Comprehensible volume LIC rendering based on 3D significance map”, *Proceedings of SPIE Conference on Visualization and Data Analysis 2002* (San Jose) pp. 142-153, 2002.
- (12) Delmarcelle, T. and L. Hesselink: “Visualizing second-order tensor fields with hyper-streamlines”, *IEEE CG&A*, 13(4), 25-33, 1993.
- (13) Wilhelms, J. and Gelder, von A.: “Octree for faster isosurface generation”, *ACM Trans. on Graphics*, Vol. 11, No. 3, pp. 201-227, 1992.
- (14) Rabenseifner, R.: “Communication Bandwidth of Parallel Programming Models on Hybrid Architectures”, *Lecture Notes in Computer Science* 2327, pp.401-412, 2002
- (15) OpenMP Web Site: <http://www.openmp.org>
- (16) Nakajima, K. and Okuda, H.: “Parallel iterative solvers for unstructured grids using directive/MPI hybrid programming model for GeoFEM platform on SMP cluster architectures”, *Journal of Concurrency and Computation: Practice and Experience*, Vol.14, No.6-7, pp.411-430, 2002.

2. HEC-MW の並列可視化ライブラリの実行方法

2.1 ファイルバージョンとメモリバージョン

図 12,図 13 に概要を示す。.

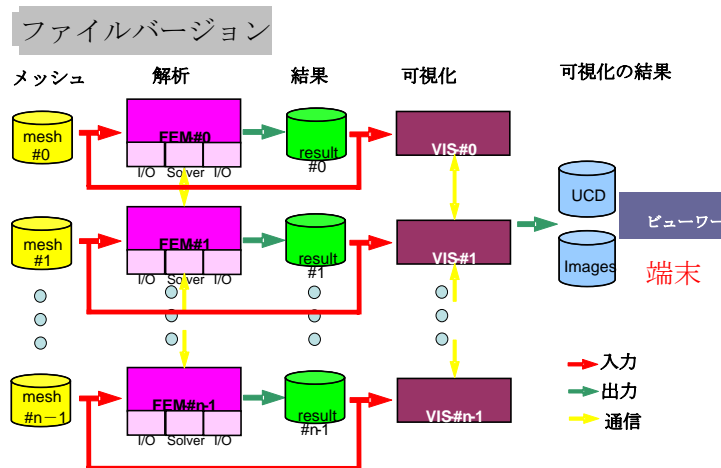


図 12: ファイルバージョンの概要

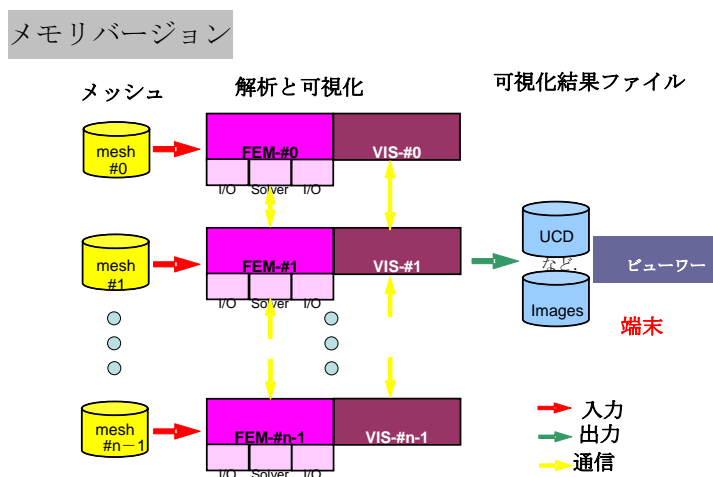


図 13: 並列可視化 (メモリバージョン)

ファイルバージョンにおいては解析と可視化の実行は独立である。シミュレーションの結果データはファイル出力され、その後、可視化プログラムがこれらのデータを読み込み並列可視化モジュールが走り始める。可視化プログラムの出力（幾何データまたはイメージデータ）はクライアントマシンに送られる。一方メモリバージョンでは、ディスクスペースを費やすことなく、メモリを介して直接にデータ転送を行い、可視化データを作成する。

可視化モジュールをインストールのためにはまず、**HEC-MW** をインストールする（**HEC-MW** インストールマニュアルを参照）。その後以下の2つの実行形式をインストールする。

- ①ファイル名： **hecmw_vis**
 ディレクトリ： **hecmw/bin/**
 説明： ファイルバージョンの実行プログラム
- ②ファイル名： **hecmwvis.a**
 ディレクトリ： **hecmw/bin/**
 説明： メモリバージョンの実行プログラム

2.2 ファイルバージョンの実行方法

(1) 入力データ

ファイルバージョンでは3つの入力ファイルが必要となる。分割されたメッシュデータ、分割された解析結果データ、可視化コントロールデータ。そしてそれぞれのファイル名は `hecmw_ctr.dat` に記述されていなければならない。

- メッシュデータの指定

(ア) 非分割メッシュ

```
!MESH, NAME=mesh, TYPE=HECMW-ENTIRE  
filename
```

ファイル名

(イ) 分割メッシュ

```
!MESH, NAME=mesh, TYPE=HECMW-DIST, IO=IN  
filename
```

ファイル名

分割メッシュの場合実際のファイル名は上記 *filename* にプロセス番号を加えたものである。:

filename . *pe#* (pe# は0から始まる昇順の数字)

例: もし4プロセスで実行させたなら,

filename.0 *filename.1* *filename.2* *filename.3*

- シミュレーションの結果ファイル名の指定

```
!RESULT, NAME=result, IO=IN  
filename
```

ファイル名

実際のファイル名はプロセス番号とタイムステップ番号に付随したものである。

filename . *pe#* . *timestep#* (pe#は0から始まる昇順の数字, timestep# は1から始まる昇順の数字)

例: '*filename.0.1*', '*filename.1.1*', '*filename.2.1*', '*filename.3.1*'

- 可視化コントロールファイルの指定

```
!CONTROL, NAME=vis_ctrl  
filename
```


HEC-MW の可視化システムにおいては hecmw_vis.ini がデフォルト値である。もしこの名のファイルがない時は上記で指定したファイルが使われる。

(2) 出力データ

可視化結果データのファイル名は下記のように指定する。

```
!RESULT,NAME=vis_out, IO=OUT  
filename
```

このファイル名は可視化の内容によってそれぞれ接尾語が加えられる。

例えば、各タイムステップで複数の PSR ファイルが存在するなら

filename_psf(PSR_id)[(.frame_id).(timestep)]. bmp

各タイムステップで1つの PSR ファイルが存在するなら

(output_filename)_psf[(.frame_id).(timestep)]. bmp

例えば、各タイムステップで複数の PVR ファイルが存在するなら

filename_pvr(PVR_id) [(.frame_id).(timestep)]. bmp

各タイムステップで1つの PVR ファイルが存在するなら

(output_filename)_pvr[(.frame_id).(timestep)]. bmp

(3) 実行コマンド

```
mpirun -np (total PE#) (path)/hecmw_vis
```

2.3 メモリバージョンの実行

メモリバージョンでは、メモリ間で直接データがやり取りされるので、そのためのファイルは使われない。したがってメッシュデータや結果データのファイル名の指定は必要ない。そして

a. call `hecmw_visualize_init()`

このファンクションによりシミュレーションコントロールデータから可視化のためのコントロールデータを読み込む。

b. call `hecmw_visualize (mesh, result, step, max_step, is_force)`

このファンクションによりタイムステップ毎の PSR または PVR の可視化モジュールを実行し可視化ライブラリに応じて、BMP, FEAMAP や AVS のファイルを結果ファイルとして出力する。

パラメータ

- result: 解析結果データへのポインタ
- step: 実行中のタイムステップ番号
- max_step: 最終タイムステップの番号
- is_force: このタイムステップにおいて可視化を実行するかどうかのフラグ

c. call `hecmw_visualize_finalize()`

3 HEC-MW 可視化パラメータ

3.1 書式とコントロールパラメータ

(1) 可視化手法の指定

!VISUAL, method=**PSR**, visual_start_step=2, visual_interval_step=5, visual_end_step=20

!VISUAL をキーワードとして新規可視化手法を指定する。

パラメーター:

- method: 可視化手法の選択
 - PSR – 並列サーフェスレンダリング
 - PVR – 並列ボリュームレンダリング
 - PST – 並列ストリームライン
- visual_start_step: 可視化処理を始めるタイムステップ番号を指定する。
省略値: -1
- visual_end_step: 可視化処理を終了するタイムステップ番号を指定する。
省略値: = visual_start_step
- visual_interval_step: 可視化処理を行うタイムステップの間隔を指定する。
省略値: 1

(2) 可視化コントロールファイルの書式

形式は以下の通り。

- a. 1ファイル内に複数の可視化手法を指定できる。
- b. 可視化開始、終了、間隔は可視化手法毎に個別に設定できる。
- c. 1つの可視化手法に対して何回でも異なったパラメータでの指定ができる。
- d. パラメータ指定は順序不同である。
- e. ほとんどのパラメータには省略値が用意されている。
- f. !! 又は # で始まる行はコメント行である。

3.2 PSR モジュールのパラメータ設定

PSR では 1 枚の画像の中に複数種類の画像を取り込むことができる。例えば複数の等値面や、複数の平面の切り口での等値線を描ける。キーワード `!SURFACE` は 1 つの属性の面を指定できる。そしていくつの属性を使うかを初めに指定しなければならない。.

例) 2 つの等値線と 2 つの平面との切り口を描くには以下のように指定する。

```
!surface_num =4
```

に続いてそれぞれの属性を 4 回繰り返して指定する。

```
!SURFACE . . .
```

```
!SURFACE . . .
```

```
!SURFACE . . .
```

```
!SURFACE . . .
```

PSR のコントロールパラメータは以下の通り。.

3.2.1 PSR モジュールのパラメータ

(1) 共通データ

キーワード	型	内容
surface_style	integer	表面タイプの指定 (省略値: 1) 1: 境界表面 2: 等値面 3: 方程式によるユーザ定義の曲面
display_method	integer	表示方法 (省略値: 1) 1. 色コードの表示 2. 境界線表示 3. 色コード及び境界線表示 4. 指定色一色の表示 5. 色分けにによる等値線表示
color_comp_name	character(100)	変数名とカラーマップとの対応 (省略値: 第一変数名)
color_subcomp_name	character(4)	変数がベクトルの時、表示するコンポーネントを指定する。 (省略値: x) norm: ベクトルのノルム x: x 成分 y: y 成分 z: z 成分
color_comp	integer	変数名に識別番号をつける (省略値: 0)
color_subcomp	integer	変数の自由度が 1 以上の時、表示される自由度番号を指定する。 0: ノルム (省略値: 1)
Iso_number	integer	等値線数を指定する。 (省略値: 5)
specified_color	real	display_method = 4 の時のカラーを指定する。 $0.0 < \text{specified_color} < 1.0$
deform_display_on	integer	変形の有無を指定する。 1: on 0: off (省略値: 0)
deform_comp_name	character(100)	変形を指定する際の採用する属性を指定する。 (省略値: DISPLACEMENT という名の変数)
deform_comp	integer	変形を指定する際の変数の識別番号 (省略値: 0)
deform_scale	real	変形を表示する際の変位スケールを指定する。 Default: 自動 $\text{standard_scale} = 0.1 \cdot \frac{\sqrt{x_range^2 + y_range^2 + z_range^2}}{\max_deform} *$

		user_defined: real_scale= standard_scale * deform_scale
Initial_style	integer	変形表示のタイプを指定する(省略値: 1) 0: 無 1: 実線メッシュ(指定がなければ青で表示) 2: グレー塗りつぶし 3: シェーディング (物理属性をカラー対応させる) 4: 点線メッシュ(指定がなければ青で表示)
deform_style	integer	初期、変形後の形状表示スタイルを指定する (省略値: 4) 0: 無 1: 実線メッシュ(指定がなければ青で表示) 2: グレー塗りつぶし 3: シェーディング (物理属性をカラー対応させる) 4: 点線メッシュ(指定がなければ青で表示)
Initial_line_color	real (3)	初期メッシュを表示する際のカラーを指定する。これは 実線、点線両者を含む。 (省略値: 青 (0.0, 0.0, 1.0))
deform_line_color	real (3)	変形メッシュを表示する際のカラーを指定する。これは 実線、点線両者を含む。 (黄色 (1.0, 1.0, 0.0))
deform_num_of _frames	integer	Deformation アニメーションのサイクルを指定する。 (deform_display_on=1) (省略値: 8)
Output_type	character(20)	出力ファイルの型を指定する。 (省略値: AVS) AVS: AVS 用 UCD データ(物体表面上のみ) COMPLETE_AVS: AVS 用 UCD データ BMP: イメージデータ(BMP フォーマット) FSTR_FEMAP_NEUTRAL: FEMAP 用ニュートラルフ ァイル VIS_FEMAP_NEUTRAL: FEMAP 用ニュートラルフ ァイル(物体表面上のみ)

(2) 等値面 (surface_style=2) の場合

キーワード	型	内容
Data_comp_name	character(100)	等値面の属性に名前をつける。
Data_subcomp_name	character(4)	変数がベクトルの時、表示するコンポーネントを指定する。(省略値: x) norm: ベクトルのノルム x: x 成分 y: y 成分 z: z 成分
Data_comp	integer	変数名に識別番号をつける (省略値: 0)
Data_subcomp	integer	変数の自由度が 1 以上の時、表示される自由度番号を指定する。 0: ノルム (省略値: 1)
iso_value	real	等値面の値を指定する。

(3) ユーザーの方程式指定による曲面の場合 (surface_style = 3)

キーワード	型	内容
method	integer	曲面の属性を指定する。(省略値: 5) 1. 球面 2. 楕円曲面 3. 双曲面 4. 方物面 5. 一般的な 2 次曲面
Point	real(3)	method = 1, 2, 3, or 4 の時の中心の座標を指定する。 (省略値: 0.0, 0.0, 0.0)
Radius	real	method = 1 の時の半径を指定する。(省略値: 1.0)
Length	real	method = 2, 3, 又は 4)の時の径の長さを指定する。 注意:楕円曲面の場合一つの径の長さは 1.0 である。
coef	real	method=5 の時、2 次曲面の係数を指定する。 $\text{coef}[1]x^2 + \text{coef}[2]y^2 + \text{coef}[3]z^2 + \text{coef}[4]xy + \text{coef}[5]xz + \text{coef}[6]yz + \text{coef}[7]x + \text{coef}[8]y + \text{coef}[9]z + \text{coef}[10]=0$ 例: coef=0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, -10.0 これは $y=10.0$ という平面を意味する。

(4) レンダリングのためのパラメータを指定する。 (output_type = BMP の時のみ有効)

キーワード	型	内容
x_resolution	integer	最終図の幅を指定する。 (省略値: 512)
y_resolution	integer	最終図の高さを指定する。 (省略値: 512)
num_of_lights	integer	照明の個数を指定する。 (省略値: 1)
position_of_lights	real(:)	照明の位置を座標で指定する。 (省略値: 正面真上) 指定方法 !position_of_lights= x, y, z, x, y, z, ... 例) !position_of_lights=100.0, 200.0, 0.0
viewpoint	real(3)	視点の位置を座標で指定する。 (省略値: $x = (x_{\min} + x_{\max})/2.0$ $y = y_{\min} + 1.5 * (y_{\max} - y_{\min})$ $z = z_{\min} + 1.5 * (z_{\max} - z_{\min})$)
look_at_point	real(3)	視線の位置を指定する。 (省略値: データの中心)
up_direction	real(3)	Viewpoint, look_at_point and up_direction にてビューフレーム を定義する。 (省略値: 0.0, 0.0, 1.0)
ambient_coef	real	周囲の明るさを指定する。 (省略値: 0.3)
diffuse_coef	real	乱反射光の強さを係数にて指定する。 (省略値 0.7)
specular_coef	real	鏡面反射の強さを係数にて指定する。 (省略値 0.6)
color_mapping_style	integer	カラーマップの方法を指定する。 (省略値: 1) 1: 完全線形マップ (全色をRGBに線形に写像する) 2: クリップ線形マップ (mincolor から maxcolor)を RGBカラスペースに写像する。 3: 非線形カラーマップ (全領域を複数の区間に分割し、区間ごとに線形マップを行う) 4. 最適自動調整 (データの分布を統計処理してカラーマップを決定する)
interval_mapping_num	integer	color_mapping_style = 3 の時の区間の数を指定する。
interval_mapping	real(:)	color_mapping_style = 2 or 3 の時の区間位置とカラー番号を指定する。 color_mapping_style = 2 の場合 !interval_mapping = [minimum color], [maximum color] If color_mapping_style = 3 の場合 !interval_mapping= [区間,対応するカラー値],...指定回繰り返し 注意: 1 行内に記述すること。
rotate_style	integer	アニメーションの回転軸を指定する。 1: x軸で回転する。 2: y軸で回転する。 3: z軸で回転する。

		4: 特に視点を指定してアニメーションする。(8 フレーム)
rotate_num_of_frames	integer	Rotation アニメーションのサイクルを指定する。(rotate_style = 1, 2, 3) (省略値: 8)
color_mapping_bar_on	integer	カラーマップバーの有無を指定する。 0: off 1: on 省略値:0
scale_marking_on	integer	カラーマップバーに値の表示の有無を指定する。 0: off 1: on 省略値:0
num_of_scale	integer	カラーバーのメモリの数指定する。(省略値:3)
font_size	real	カラーマップバーの値表示の際のフォントサイズを指定する。 範囲: 1.0~4.0. (省略値:1.0)
font_color	real(3)	カラーマップバーの値表示の際の表示色を指定する。 (省略値: 1.0, 1.0, 1.0 (白))
background_color	real(3)	背景色を指定する。(省略値: 0.0, 0.0, 0.0 (黒))
isoline_color	read (3)	等値線の色を指定する。(省略値: その値と同じ色)
boundary_line_on	integer	データの地域を表示の有無を指定する。 0: off 1: on 省略値:0
color_system_type	integer	カラーマップのスタイルを指定する(省略値: 1) 1: (青ー赤) (昇順に) 2: レインボーマップ (赤から紫へ昇順に) 3: (黒ー白) (昇順に).
fixed_range_on	integer	カラーマップの方法を他のタイムステップに対して保持するか否かを指定する。0: off 1: on (省略値 0)
Range_value	real (2)	区間を指定する。

3.2.2 パラメータの設定例

(1) !surface_num

1つのサーフェスレンダリング内のサーフェス数

例：図18は4つのサーフェスがあり、2つは等値面で $\text{pressure}=1000.0$ と $\text{pressure}=-1000.0$ 2つは平面の切り口で $z=-1.0$ と $z=1.0$

`!surface_num = 4`

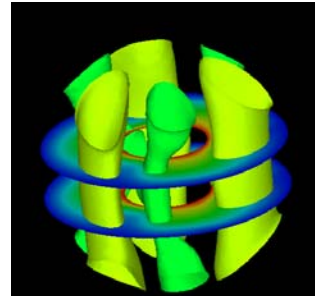


図. 18: surface_num の設定例

(2) !surface

サーフェスの内容を設定する。

例：図19は4つのサーフェスがありその内容は以下の通りである。

```
!surface_num = 2
!SURFACE
!surface_style=2
!data_comp_name =   press
!iso_value   =  1000.0
!display_method = 4
!specified_color = 0.45
!output_type   =  BMP
!SURFACE
!surface_style=2
!data_comp_name =   press
!iso_value   = -1000.0
!display_method = 4
!specified_color = 0.67
```

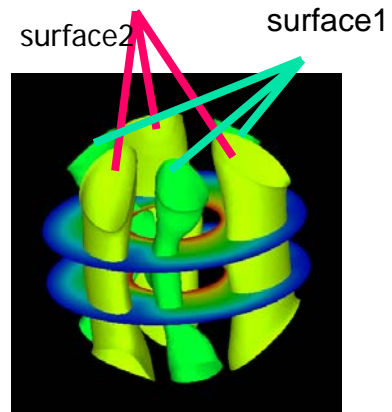


図. 19: surface の設定例

(3) !surface_style

サーフェスのスタイルを指定する。

- 1: 境界面
- 2: 等値面
- 3: 任意の2次曲面

$$\begin{aligned} & \text{coef}[1]x^2 + \text{coef}[2]y^2 + \text{coef}[3]z^2 + \text{coef}[4]xy + \text{coef}[5]xz \\ & + \text{coef}[6]yz + \text{coef}[7]x + \text{coef}[8]y + \text{coef}[9]z + \text{coef}[10]=0 \end{aligned}$$



図. 20: surface_style の設定例

(2) **!color_comp_name !color_comp !color_subcomp**

物理量からカラーマップへの対応を指定する。HECMW_IO にてまず、必要な物理量やその自由度番号に名前をつける。これにより HECMW の結果データの構造体 node_label(:)や nn_dof(:)に名前がはいる。

Then you can define which one you hope to map into color by

!color_comp_name (文字列、省略値：初めの変数)

例： **!color_comp_name = pressure**

!color_comp (整数、省略値：0)

物理量の識別番号 (0 以上の整数)

例： **!color_comp = 2**

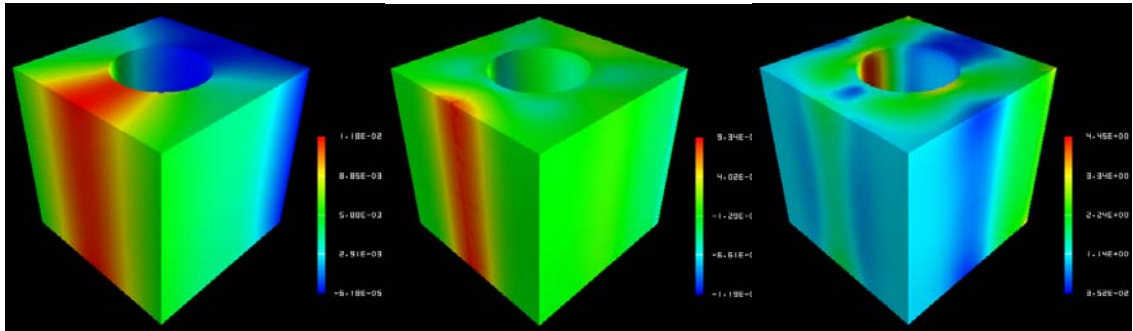
!color_subcomp (整数、省略値：1)

物理量がベクトル量のような自由度数 1 以上の時、その自由度番号

例： **!color_subcomp = 0**

構造解析において例えば

物理量	変位	ひずみ	応力
自由度数	3	6	7

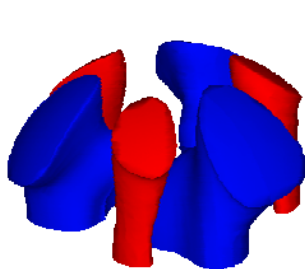


!color_comp_name=displacement !color_comp_name=strain !color_comp = 3
!color_subcomp = 1 !color_subcomp_name = 1 !color_subcomp = 7

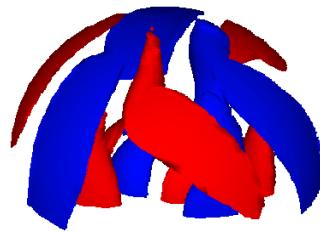
図 22: color_comp, color_subcomp および color_comp_name の設定例

(3) !data_comp_name !data_comp !data_subcomp

surface_style=2 の時、可視化する等値面の物理量を指定する。



!data_comp_name=pressure



!data_comp_name=vorticity
!data_subcomp=3

図 23: data_comp, data_subcomp 及び data_comp_name の設定例

(4) !method

面との切り口を指定する際、その面の設定方法を指定する。

```

!surface_num =2
!surface
!surface_style = 3
!method=5
!coef=0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, -0.35
!color_comp_name = temperature
!surface
!surface_style = 3
!method=5
!coef=0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.35
!color_comp_name = temperature

```

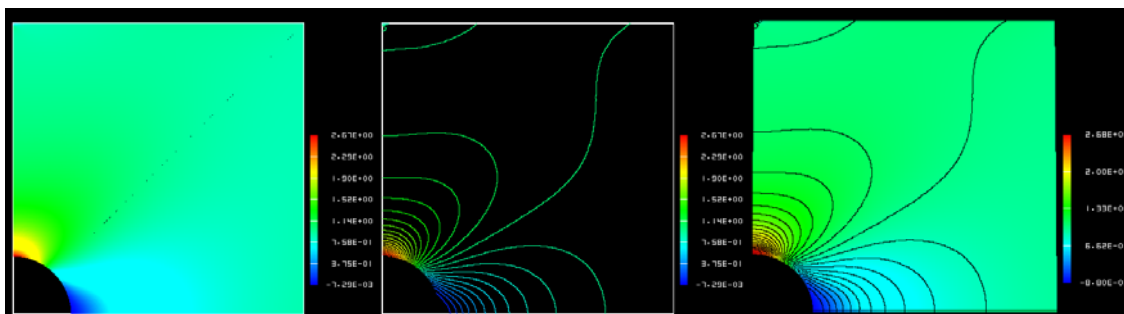
図 24: method の設定例

これにより平面 $z=0.35$ と $z=-0.35$ の断面が可視化される。

(5) !display_method

表示方法 (省略値: 1)

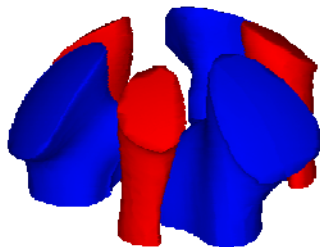
1. 色コードの表示
2. 境界線表示
3. 色コード及び境界線表示
4. 指定色一色の表示
5. 色分けによる等値線表示



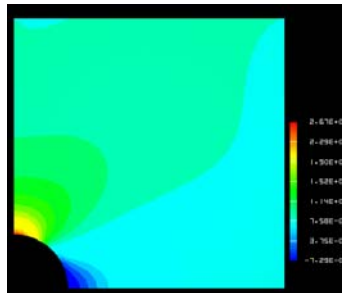
!display_method=1

!display_method=2

!display_method=3



`!display method=4`

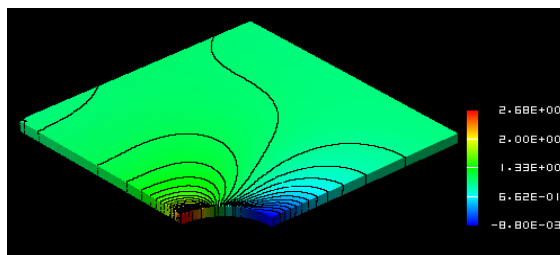


`!display method=5`

図 25: `display_method` の設定例

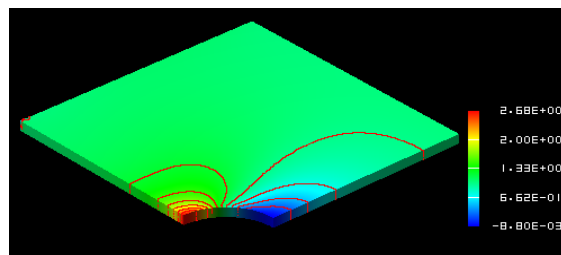
(6) `!isoline_number` and `!isoline_color`

`display_method=2,3` 又は 5 の時



`!isoline_number = 30`

`!isoline_color = 0.0, 0.0, 0.0`



`!isoline_number = 10`

`!isoline_color = 1.0, 0.0, 0.0`

図. 26: `isoline_number` と `isoline_color` の設定例

(10) `!initial_style` and `!deform_style`

初期の形状、変形後の形状の表示スタイルを指定する。

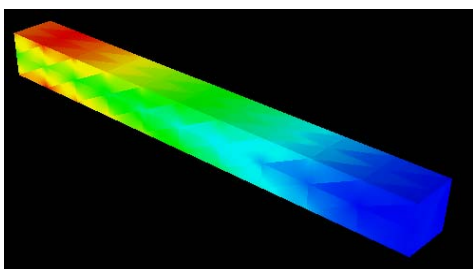
- 0: 無
- 1: 実線メッシュ(指定がなければ青で表示)
- 2: グレー塗りつぶし
- 3: シェーディング
(物理属性をカラー対応させる)
- 4: 点線メッシュ(指定がなければ青で表示)

(11) `!deform_scale`

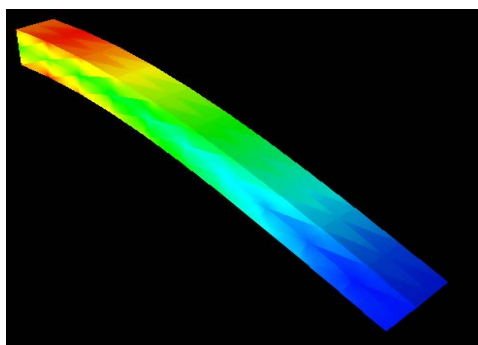
変形を表示する際の変位スケールを指定する。

Default: 自動

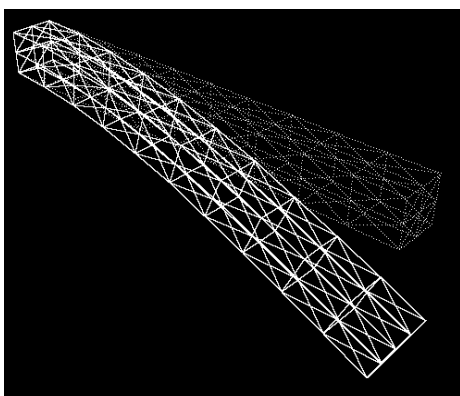
$$\text{standard_scale} = 0.1 * \sqrt{x_range^2 + y_range^2 + z_range^2} / \text{max_deform}$$



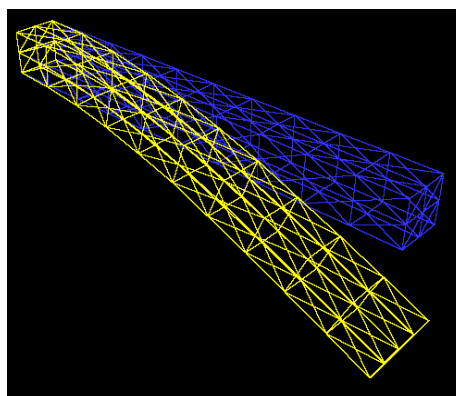
!initial_style=2
!deform_style = 0



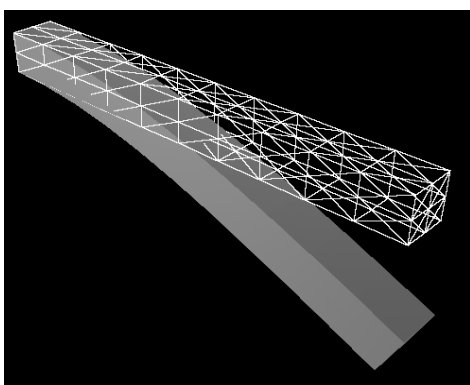
!initial_style=0
!deform_style = 2



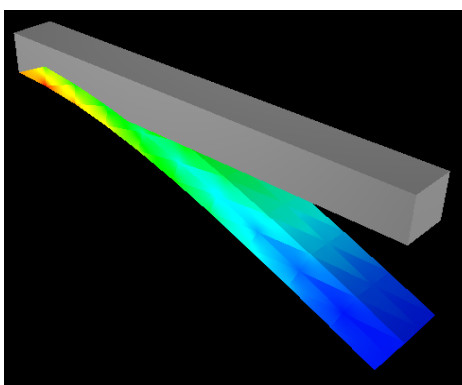
!initial_style=4
!deform_style = 1
!initial_line_color = 1.0, 1.0, 1.0



!initial_style=1
!deform_style = 1 **NASTRAN style**
!initial_line_color = default

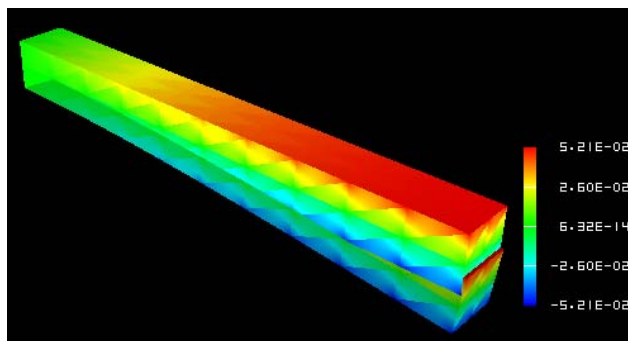


!initial_style=1
!deform style = 2

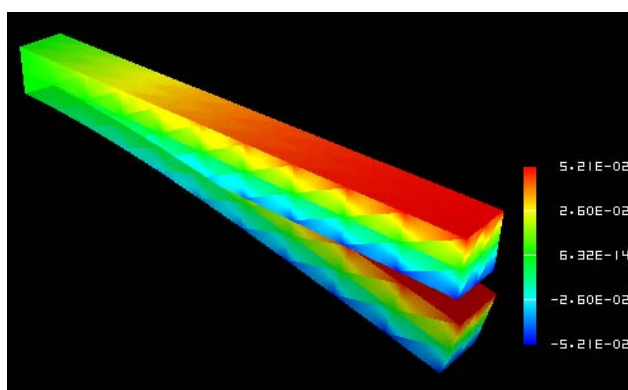


!initial_style=2
!deform style = 3

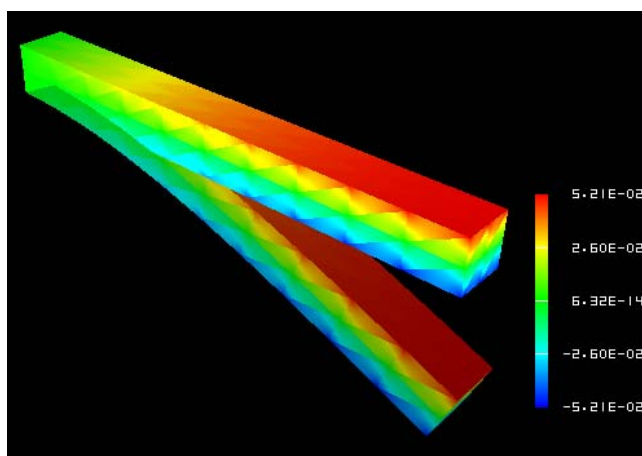
図 27: display styles の設定例



!deform_scale=1.0



!deform_scale=2.0

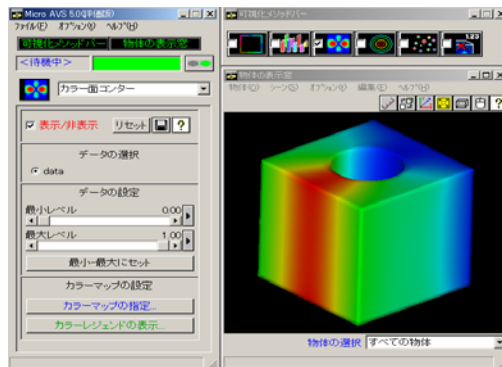


!deform_scale=4.0

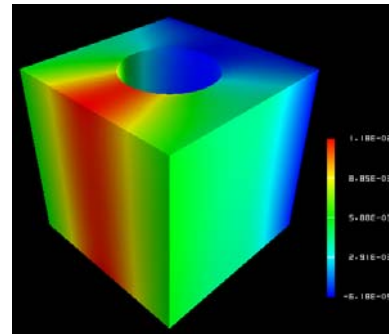
図 28:deform_scale の設定例

(12) !output_type

- =AVS: AVS UCD 形式で可視化パッチを出力する。
このファイルを AVS 環境にて、自由に可視化できる。
- =BMP: BMP イメージ形式に出力する。これを Windows 環境にて Microsoft Photo Editor, ACDSee や Photoshop などにて表示できる。
- =FSTR_FEMAP_NEUTRAL: FEMAP ニュートラルファイルを出力する。このファイルを FEMAP にて自由に可視化できる。
- =VIS_FEMAP_NEUTRAL: FEMAP ニュートラル形式で可視化パッチを出力する。このファイルを FEMAP にて自由に可視化できる。



!output_type = AVS

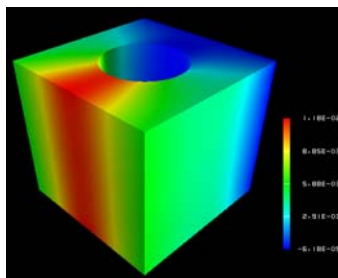


!output_type=BMP

図 29: output_type の例

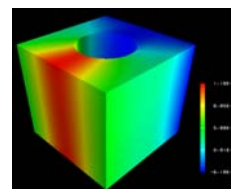
(13) !x_resolution and !y_resolution

output_type=BMP の時、解像度を指定する。



!x_resolution=500

!y_resolution=500



!x_resolution=300

!y_resolution=300

図 30: x_resolution と y_resolution の設定例

(14) !viewpoint, !look_at_point, !up_direction

viewpoint: 視点の位置を座標で指定する。

省略値: $x = (xmin + xmax)/2.0$,
 $y = ymin + 1.5 * (ymax - ymin)$,
 $z = zmin + 1.5 * (zmax - zmin)$

look_at_point: 視線の位置を指定する。
(省略値: データの中心)

up_direction: Viewpoint, look_at_point と
up_direction にてビューフレーム
を指定する。
default: 0.0 0.0 1.0

View coordinate frame:

原点: look_at_point
z 軸: viewpoint - look_at_point
x 軸: up × z axis
y 軸: z axis × x axis

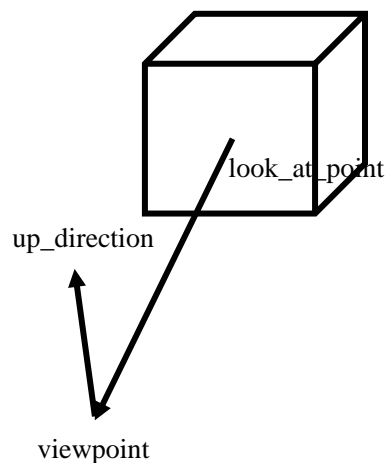


図 31: ビューフレームの決定法

Ex:

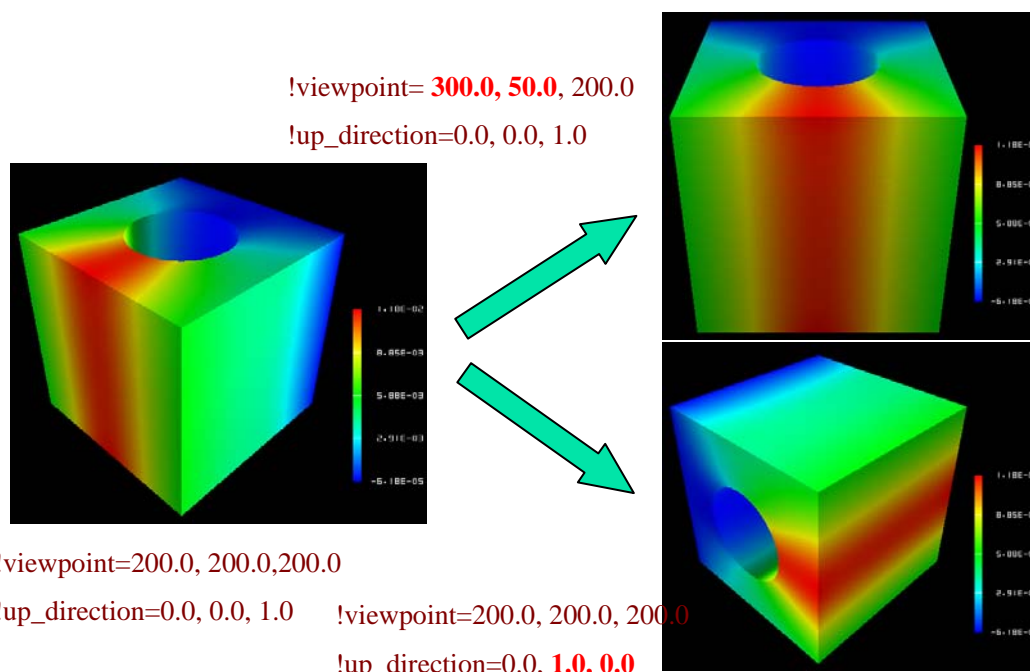
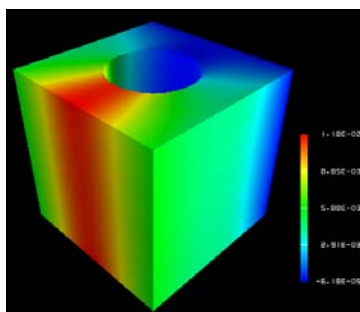


図 32: !viewpoint, !look_at_point と up_direction の設定例

(15) !ambient_coef, !diffuse_coef, !specular_coef

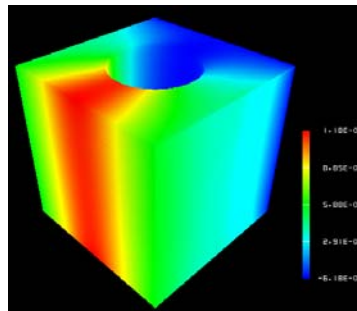
照明モデルの係数設定

ambient_coef,を増加すると 3 次元の奥行き方向の情報が損なわれる。



!ambient_coef = 0.3

!diffuse_coef = 0.7



!ambient_coef = 0.9

!diffuse_coef = 0.1

図 33: 照明モデルパラメータの設定例

(16) !color_mapping_bar_on, !scale_marking_on, !num_of_scales

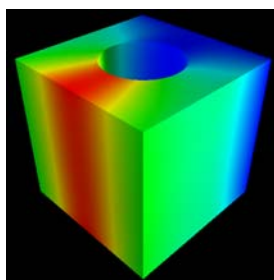
!color_mapping_bar_on: color mapping bar の表示有無を指定する。

0: off 1: on (省略値: 0)

!scale_marking_on: color mapping bar のメモリの有無を指定する

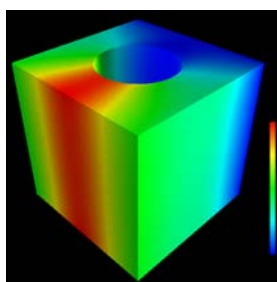
0: off 1: on (省略値: 0)

!num_of_scales: メモリ の数 を指定する。 (省略値: 3)



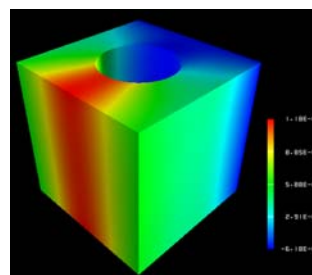
!color_mapping_bar_on=0

!scale_marking_on=0



!color_mapping_bar_on=1

!scale_markig_on=0



!color_mapping_bar_on=1

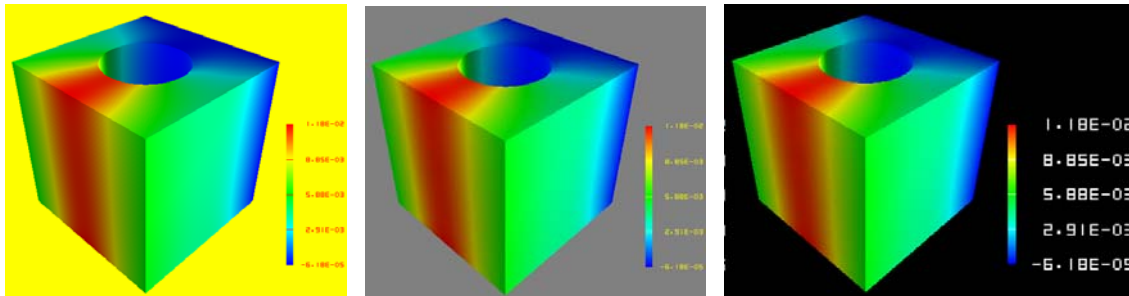
!scale_marking_on=1

!num_of_scale=5

図 34: color mapping bar の表示の例

(16) ! background_color, !font_color, !font_size

背景色や文字フォントを指定する。



```
!background_color =1.0,1.0,0.0 !background_color =0.5, 0.5, 0.5 !background_color =0.0, 0.0,0.0  
!font_color=1.0, 0.0, 0.0      !font_color=1.0, 1.0, 0.0      !font_color=1.0, 1.0, 1.0  
!font_size=1.5                 !font_size =1.5                !font_size=2.5
```

図 35: background と font の設定例

3.3 PVR モジュールのパラメータ

3.3.1 PVR モジュールのパラメータ説明

(1) 分割の細かさの指定

Name	Type	Description
maximum_refinement	integer	分割の細かさのレベルを指定する。(省略値:100)更にプログラムは、メッシュの最小長さから自動調整を行う。ユーザーはメモリサイズも考慮してこの値を選ぶ必要がある。
n_voxel_x	integer	X 軸方向のスーパーボクセルの分割数を指定する。 (省略値:プロセス数)
n_voxel_y	integer	Y 軸方向のスーパーボクセルの分割数を指定する。 (省略値: 1)
n_voxel_z	integer	Z 軸方向のスーパーボクセルの分割数を指定する。 (省略値: 1)
voxel_filename	character(100)	スーパーボクセルを定義するファイルの名前を指定する。 (省略値:なし) 特定の領域などをきれいに表示するにはユーザー指定のボクセルファイルを作るべきである。
x_specified_level	integer	X 軸方向の細かさのレベル (省略値:100)
y_specified_level	integer	Y 軸方向の細かさのレベル (省略値:100)
z_specified_level	integer	Z 軸方向の細かさのレベル (省略値:100)

(2) 画面を指定するパラメータ

Name	Type	Description
x_resolution	integer	幅を指定する(default: 512)
y_resolution	integer	高さを指定する (default: 512)
num_of_lights	integer	照明の個数を指定する (default: 1)
position_of_lights	real(:)	照明の位置を座標で指定する。(省略値: 正面真上) 指定方法 !position_of_lights= x, y, z, x, y, z, ... 例) !position_of_lights=100.0, 200,0, 0.0
viewpoint	real(3)	視点の位置を座標で指定する。 (省略値: $x = (x_{min} + x_{max})/2.0$ $y = y_{min} + 1.5 * (y_{max} - y_{min})$ $z = z_{min} + 1.5 * (z_{max} - z_{min})$)
look_at_point	real(3)	視線の位置を指定する。

		(省略値: データの中心)
up_direction	real(3)	Viewpoint, look_at_point と up_direction にてビューフレームを定義する。(省略値: 0.0, 0.0, 1.0)

(3) 照明についてのパラメータ

Name	Type	Description
ambient_coef	Real	周囲の明るさを指定する。(省略値: 0.3)
diffuse_coef	Real	乱反射光の強さを係数にて指定する。(省略値 0.7)
specular_coef	Real	鏡面反射の強さを係数にて指定する。(省略値 0.6)

(4) カラーマップを指定するパラメータ

Name	Type	Description
color_mapping_style	integer	カラーマップの方法を指定する。(省略値: 1) 1: 完全線形マップ (全色をRGBに線形に写像する) 2: クリップ線形マップ (mincolor から maxcolor)を RGB カラー空間に写像する。 3: 非線形カラーマップ (全領域を複数の区間に分割し、区間ごとには線形マップを行う) 4: 最適自動調整 (データの分布を統計処理してカラーマップを決定する)
interval_mapping_num	integer	color_mapping_style = 3 の時の区間の数を指定する。
interval_mapping	real(:)	color_mapping_style = 2 or 3 の時の区間位置とカラー番号を指定する。 color_mapping_style = 2 の場合 !interval_mapping = [minimum color], [maximum color] If color_mapping_style = 3 の場合 !interval_mapping= [区間, 対応するカラー値], ...指定回繰り返し 注意: 1 行内に記述すること。

(5) 不透明度を指定するパラメータ

Name	Type	Description
opacity_mapping_style	integer	不透明度のタイプ (省略値: 1) 1: 一定 2: 変化の激しい領域をハイライトする。 3: 特定の領域をハイライトする。 4: 区間内の値をハイライトする。

		5: 不透明度を視点からの距離に比例させる。 6: 不透明度を視点からの距離に反比例させる。 7: 特定のデータ特性に比例させる。 8: 不透明度のテーブルを参照する。
opacity_value	real	不透明度の値 (opacity_mapping_style = 1 の時有効) (省略値:0.02)
num_of_features	integer	opacity_mapping_style = 3 の時、点の数を opacity_mapping_style = 4 の時、区間の数を指定する。
feature_points	real(:)	点または区間を指定する。 (opacity_mapping_style = 3 又は 4) 入力方法 opacity_mapping_style = 3 の時 !feature_points= [点 1], [減少区間 1], [不透明度 1], [点 2], [減少区間 2], [不透明度 2], ... opacity_mapping_style = 4 の時 !feature_points= [区間の下限 1], [区間の上限 1], [不透明度 1], [区間の下限 2], [区間の上限 2], [不透明度 2] ... 1 行で記述すること。
name_lookup	character(100)	参照テーブルのファイル名 (opacity_mapping_style = 8 の時)

(6) その他のパラメータ

Name	Type	Description
rotate_style	integer	アニメーションの回転軸を指定する。 1: x軸で回転する。 2: y軸で回転する。 3: z軸で回転する。 4: 特に視点を指定してアニメーションする。(8 フレーム)
num_of_frames	integer	rotation アニメーションのサイクルを指定する。(rotate_style = 1, 2, 3) (省略値: 8)
color_mapping_bar_on	integer	カラーマップバーの有無を指定する。 0: off 1: on 省略値:0
scale_marking_on	integer	カラーマップバーに値の表示の有無を指定する。 0: off 1: on 省略値:0
num_of_scales	integer	カラーバーのメモリの数を指定する。(省略値:3)
font_size	real	カラーマップバーの値表示の際のフォントサイズを指定する。 範囲: 1.0~4.0. (省略値:1.0)
font_color	real(3)	カラーマップバーの値表示の際の表示色を指定する。 (省略値: 1.0, 1.0, 1.0 (白))

background _color	real(3)	背景色を指定する。(省略値: 0.0, 0.0, 0.0 (黒))
color_system _type	integer	カラーマップのスタイルを指定する(省略値: 1) 1: (青ー赤) (昇順に) 2: レインボーマップ (赤から紫へ昇順に) 3: (黒ー白) (昇順に).
fixed_range _on	integer	カラーマップの方法を他のタイムステップに対して保持するか否かを指定する。0: off 1: on (省略値 0)
range_value	real (2)	区間を指定する。
histogram_on	integer	Output histogram image of data distribution or not 1: データの分布を数値で出力 2: データの分布をグラフで出力
display_range	real(6)	モデルの中の特定の部位だけを表示する。

3.3.2 詳しい説明

(1) !color_mapping_style

カラーマップの方法を指定する。(省略値: 1)

- 1: 完全線形マップ (全色をRGBに線形に写像する)
- 2: クリップ線形マップ (mincolor から maxcolor)を RGBカラー空間に写像する。
- 3: 非線形カラーマップ (全領域を複数の区間に分割し、区間ごとには線形マップを行う)
- 4: 最適自動調整 (データの分布を統計処理してカラーマップを決定する)

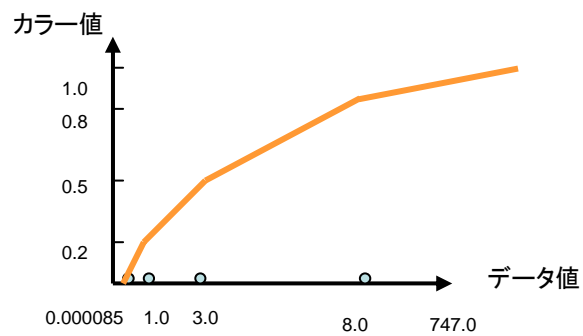
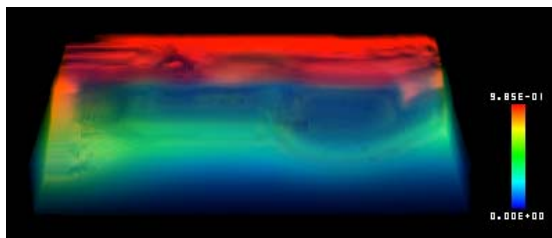
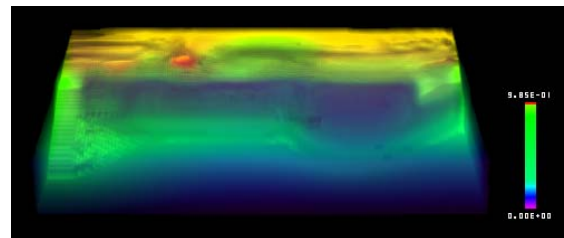


図 36: color mapping style=3 の時の例



!color_mapping_style=1

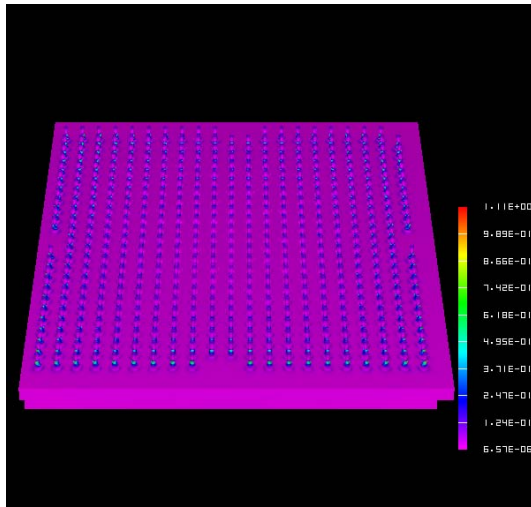


!color_mapping_style=3

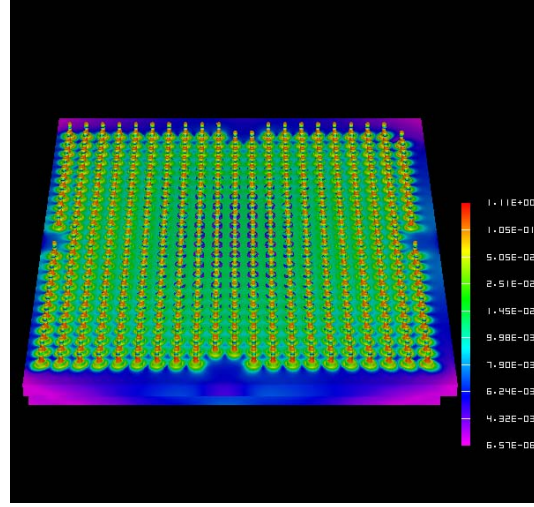
!interval_mapping_num=4

!interval_mapping=0.0,0.0,0.25,0.5,
0.85,0.6,0.955,0.7,0.965,1.0

図 37 : !color_mapping_style の種々の例



!color_mapping_style=1



!color_mapping_style=4

図 38 : !color_mapping_style の種々の例

(2) !histogram_on

データの分布を表示する

!histogram_on

- 1: データの分布を数値で出力
- 2: データの分布をグラフで出力

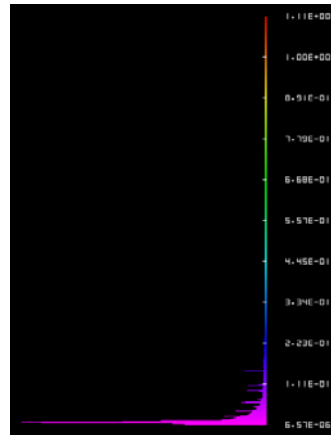
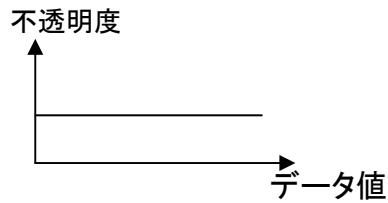


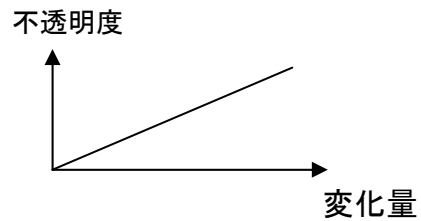
図 39: An example of histogram graph output

(3) !opacity_mapping_style

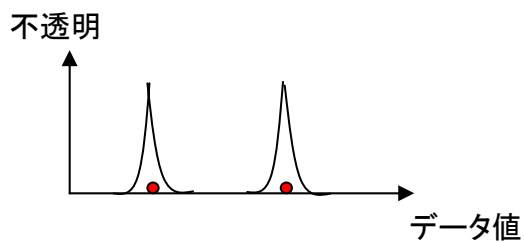
3次元空間を2次元の平面で表示するため、時として表示が複雑になり混乱することがある。そこで、重点ポイントを絞って表示することが有用なことがある。ユーザーが不透明度を指定して、重点ポイントを鮮明に表示できることを狙っている。



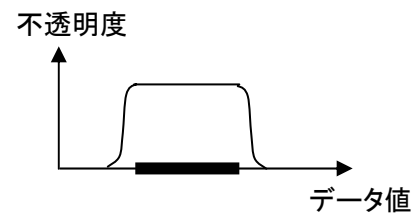
1: 一定値



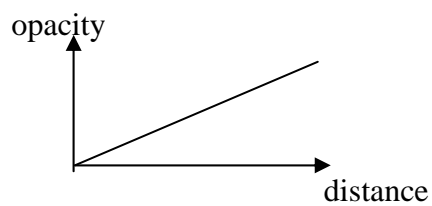
2: 変化に応じてハイライトする



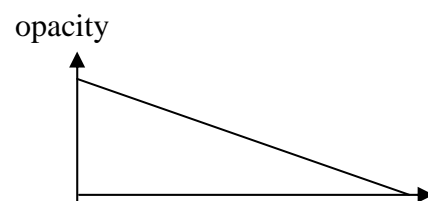
3: 特定の値域をハイライト



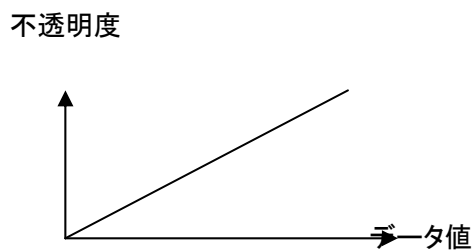
4: 特定の区間をハイライト



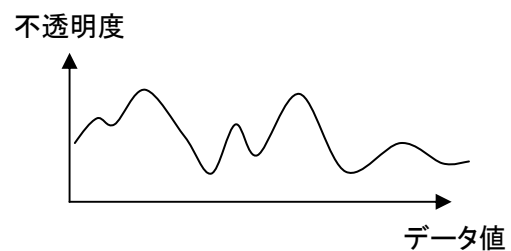
5: 距離に比例した不透明度



6: 距離に逆比例した透明度



7: 特定のデータに比例した不透明度



8: 参照テーブル

(4) !rotate_style

アニメーションの際の回転方向を指定する。

- 1: x 軸回転
- 2: y 軸回転
- 3: z 軸回転
- 4: 視点を変えてのアニメーション (8 フレーム)

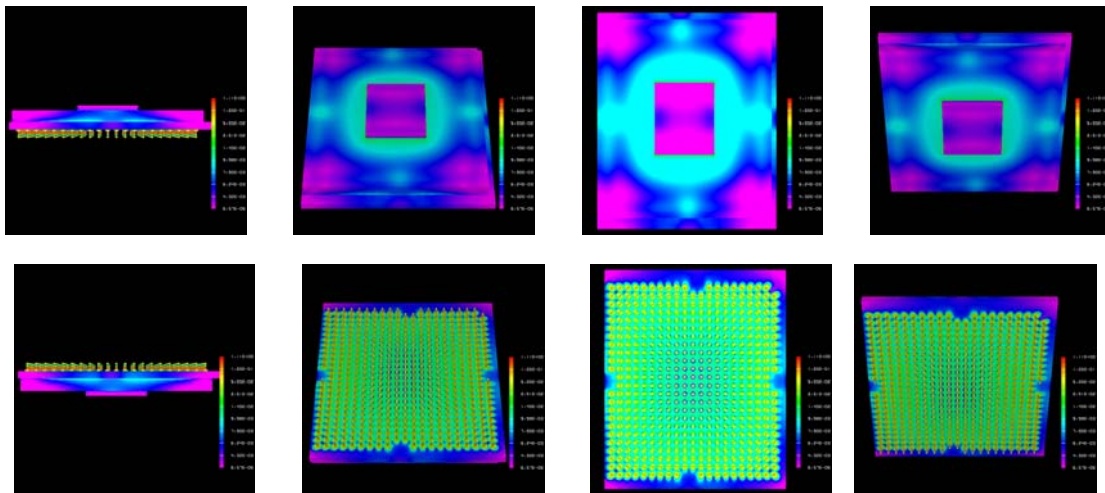


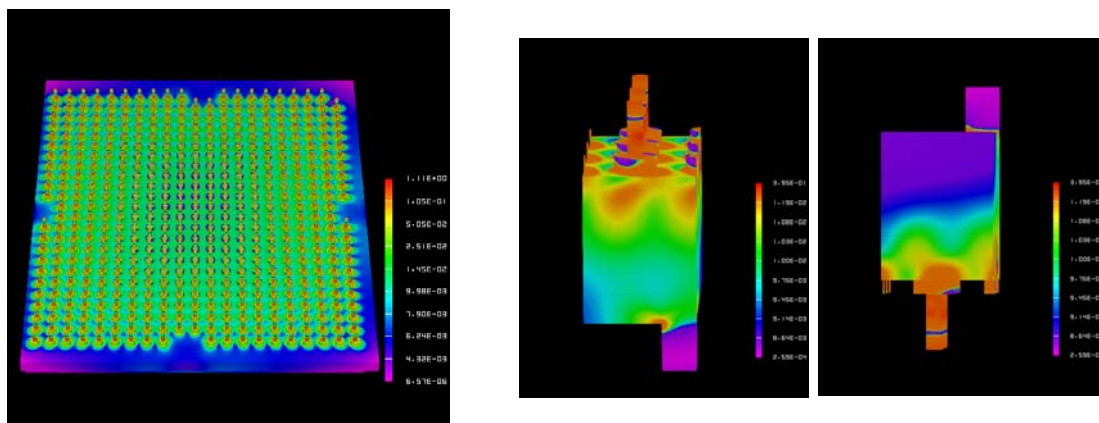
図 40 : rotate_style=1 の例

(5) !display_range

モデルの中の特定の部位だけを表示する。

形式は、

`!display_range=minx, maxx, miny, maxy, minz, maxz`



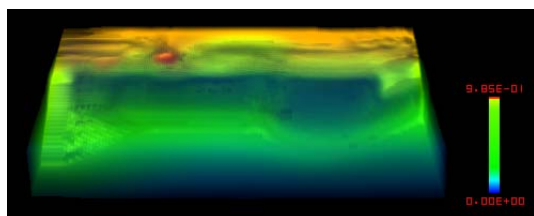
`!display_range=-6, -4, -9, -5, -1.25, 3.53`

図 41: display_range の設定例

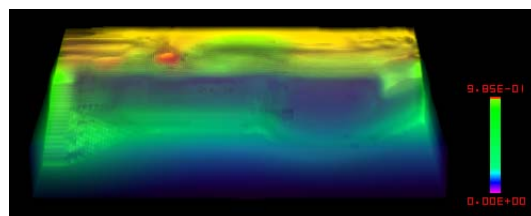
(6) !color_system_type

カラーマップのスタイルを指定する(省略値: 1)

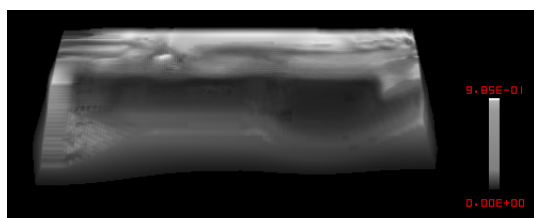
- 1: (青ー赤) (昇順に)
- 2: レインボーマップ (赤から紫へ昇順に)
- 3: (黒ー白) (昇順に)



!color_system_type=1



!color_system_type=2



!color_system_type=3

図 42: color_system_type の設定例

3.4 PSF と PVR におけるミクスチャーパラメータの設定例

時間進行のある 解析結果の可視化の例を示す。

- (1) timestep10における圧力の等値面
 - (2) timesteps from 1 から 500のすべてにわたってボリュームレンダリングを行う。
 - (3) ベクトル量の等値面を 2 つtimestep 10において出力する。
- この時以下のようにパラメータを設定する。

!! The first visualization output is made by parallel surface rendering module at just timestep 10

!VISUAL, method=**PSR**, visual_start_step=**10**, visual_end_step=**10**

!! The first PSR visualization consists of two iso-surfaces, generated at timestep=10

!surface_num = 2

!SURFACE

!surface_style=2

!data_comp_name = pressure

!iso_value = -1000.0

!display_method = 4

!specified_color = 0.0

!output_type = BMP

!SURFACE

!surface_style=2

!data_comp_name = pressure

!iso_value = 1000.0

!display_method = 4

!specified_color = 1.0

!x_resolution = 500

!y_resolution = 500

!viewpoint = 10.0, 10.0, 10.0

!ambient_coef= 0.3

!diffuse_coef= 0.7

!specular_coef= 0.5

!font_color = 1.0, 1.0, 1.0

(==to be continue)

(==continued ==)

**!! The second visualization output is made by parallel volume rendering module for all
!! timesteps from 1 to 500**

!VISUAL, method=PVR, visual_start_step=1, visual_end_step=500

!color_comp_name = vorticity

!color_subcomp_name = z

!x_resolution = 300

!y_resolution = 300

!num_of_lights = 1

!position_of_lights = 0.0, 10.0, 18.0

!viewpoint = 10.0, 10.0, 10.0

!color_mapping_style= 3

!interval_mapping_num= 3

!interval_mapping= -150.0, 0.0, 0.0, 0.5, 50.0, 0.7, 110.0, 1.0

!opacity_mapping_style = 3

!opacity_value = 0.001

!num_of_features= 4

!feature_points= 0.0, 20.0, 0.0, -120.0, 100.0, 0.008, 120.0, 100.0, 0.008, -170.0, 30.0, 0.01

!color_mapping_bar_on = 1

!scale_marking_on = 1

!color_system_type= 1

!font_size = 1.5

!color_bar_style = 2

!num_of_scale = 5

!fixed_range_on = 1

!range_value= -200.0, 200.0

(==to be continue)

(==continued ==)

!! The third visualization output is made by parallel surface rendering module

!! at just timestep 10

!VISUAL, method=PSR, visual_start_step=10, visual_end_step=10

!surface_num = 2

!SURFACE

!surface_style=2

!data_comp_name = vorticity

!data_subcomp_name = z

!iso_value = -100.0

!display_method = 4

!specified_color = 0.0

!output_type = BMP

!SURFACE

!surface_style=2

!data_comp_name = vorticity

!data_subcomp_name=z

!iso_value = 80.0

!display_method = 4

!specified_color = 1.0

!x_resolution = 500

!y_resolution = 500

!viewpoint = 10.0, 10.0, 10.0

!ambient_coef= 0.3

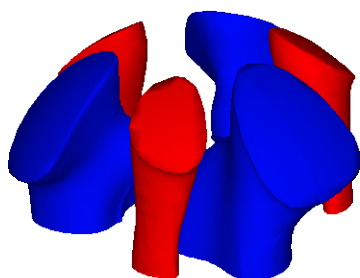
!diffuse_coef= 0.7

!specular_coef= 0.5

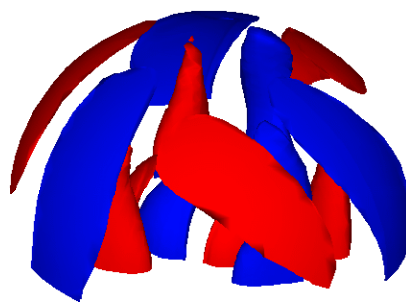
!font_color = 1.0, 1.0, 1.0

以下のような図が出力される。

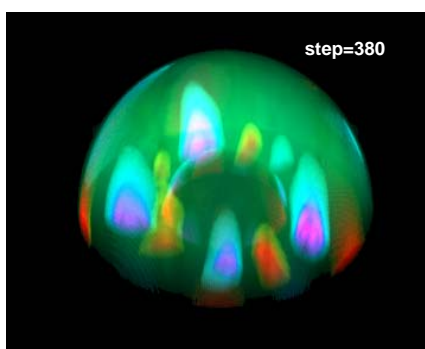
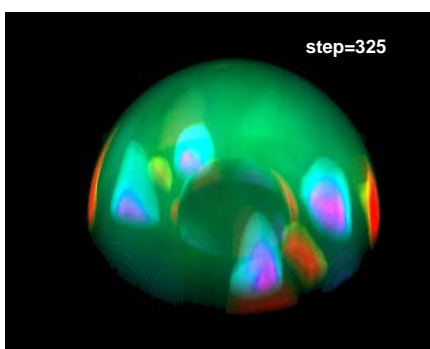
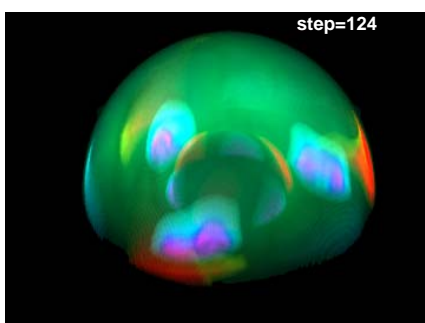
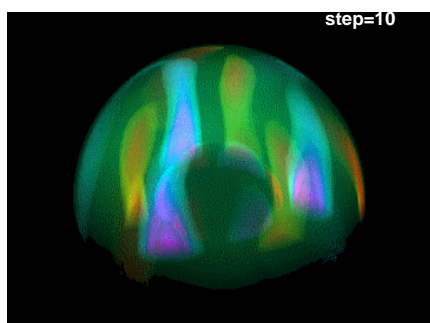
- (1) 圧力の等値面が出力される。
- (2) 500 枚のボリュームレンダリングデータ
- (3) ベクトル量の z 成分の大きさの等値面



(1) 圧力の等値面
result_psf1.10.bmp



(3) ベクトルの z 成分の等値面
result_psf2.10.bmp



変

数名

(2) 500 枚の図
図 43: ミクスチャーパラメータの設定例

3.5 PST モジュールのパラメータ

3.5.1 PST モジュールのパラメータ説明

(1) ストリームラインパラメータの指定

Name	Type	Description
tracing_ direction	integer	トレース方向 -1: 後方へ 1: 前方へ 2: 両方の前後方向
vector_comp _name	character(100)	ベクトル属性に名前をつける。
vector_comp	integer	ベクトル変数名に識別番号をつける (省略値: 0)
color_comp _name	character(100)	変数名とカラーマップとの対応 (省略値: 第一変数名)
color_subcomp _name	character(4)	変数がベクトルの時、表示するコンポーネントを指定する。(省略値: x) norm: ベクトルのノルム x: x 成分 y: y 成分 z: z 成分
color_comp	integer	変数名に識別番号をつける (省略値: 0)
color_subcomp	integer	変数の自由度が 1 以上の時、表示される自由度番号を指定する。 0: ノルム (省略値: 1)
seed_style	integer	種まきのスタイル 1: defined by users directly 2: defined in some file 3: semi-automatic by YZ plane 4: semi-automatic by XZ plane 5: semi-automatic by XY plane
seed_plane _position	real	種子平面の位置 if seed_style=3, then z=seed_plane_position if seed_style=4, then x=seed_plane_position if seed_style=5, then y=seed_plane_position
seed_filename	character(100)	種子ポイント座標のファイル
seed_num	integer	種子ポイントの数 (省略値: 25)
seed_point	real (:)	種子ポイントの座標
output_type	character(4)	出力ファイルのタイプ AVS: AVS UCD format BMP

(2) レンダリングのためのパラメータを指定する。 (output_type = BMP の時のみ有効)

キーワード	型	内容
x_resolution	integer	最終図の幅を指定する。 (省略値: 512)
y_resolution	integer	最終図の高さを指定する。 (省略値: 512)
num_of_lights	integer	照明の個数を指定する。 (省略値: 1)
position_of_lights	real(:)	照明の位置を座標で指定する。 (省略値: 正面真上) 指定方法 !position_of_lights= x, y, z, x, y, z, ... 例) !position_of_lights=100.0, 200.0, 0.0
viewpoint	real(3)	視点の位置を座標で指定する。 (省略値: $x = (x_{\min} + x_{\max})/2.0$ $y = y_{\min} + 1.5 * (y_{\max} - y_{\min})$ $z = z_{\min} + 1.5 * (z_{\max} - z_{\min})$)
look_at_point	real(3)	視線の位置を指定する。 (省略値: データの中心)
up_direction	real(3)	Viewpoint, look_at_point and up_direction にてビューフレーム を定義する。 (省略値: 0.0, 0.0, 1.0)
ambient_coef	real	周囲の明るさを指定する。 (省略値: 0.3)
diffuse_coef	real	乱反射光の強さを係数にて指定する。 (省略値 0.7)
specular_coef	real	鏡面反射の強さを係数にて指定する。 (省略値 0.6)
color_mapping_style	integer	カラーマップの方法を指定する。 (省略値: 1) 1: 完全線形マップ (全色をRGBに線形に写像する) 2: クリップ線形マップ (mincolor から maxcolor)を RGBカラスペースに写像する。 3: 非線形カラーマップ (全領域を複数の区間に分割し、区間ごとに線形マップを行う) 4. 最適自動調整 (データの分布を統計処理してカラーマップを決定する)
interval_mapping_num	integer	color_mapping_style = 3 の時の区間の数を指定する。
interval_mapping	real(:)	color_mapping_style = 2 or 3 の時の区間位置とカラー番号を指定する。 color_mapping_style = 2 の場合 !interval_mapping = [minimum color], [maximum color] If color_mapping_style = 3 の場合 !interval_mapping= [区間,対応するカラー値],...指定回繰り返し 注意: 1 行内に記述すること。
rotate_style	integer	アニメーションの回転軸を指定する。 1: x軸で回転する。 2: y軸で回転する。 3: z軸で回転する。

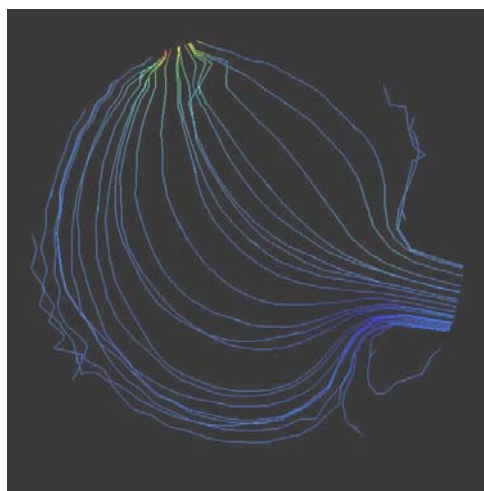
		4: 特に視点を指定してアニメーションする。(8 フレーム)
num_of_frames	integer	アニメーションのサイクルを指定する。(rotate_style = 1, 2, 3) (省略値: 8)
color_mapping_bar_on	integer	カラーマップバーの有無を指定する。 0: off 1: on 省略値:0
scale_marking_on	integer	カラーマップバーに値の表示の有無を指定する。 0: off 1: on 省略値:0
num_of_scale	integer	カラーバーのメモリの数指定する。(省略値:3)
font_size	real	カラーマップバーの値表示の際のフォントサイズを指定する。 範囲: 1.0~4.0. (省略値:1.0)
font_color	real(3)	カラーマップバーの値表示の際の表示色を指定する。 (省略値: 1.0, 1.0, 1.0 (白))
background_color	real(3)	背景色を指定する。(省略値: 0.0, 0.0, 0.0 (黒))
isoline_color	read (3)	等値線の色を指定する。(省略値:その値と同じ色)
boundary_line_on	integer	データの地域を表示の有無を指定する。 0: off 1: on 省略値:0
color_system_type	integer	カラーマップのスタイルを指定する(省略値: 1) 1: (青ー赤) (昇順に) 2: レインボーマップ (赤から紫へ昇順に) 3: (黒ー白) (昇順に).
fixed_range_on	integer	カラーマップの方法を他のタイムステップに対して保持するか否かを指定する。0: off 1: on (省略値 0)
range_value	real (2)	区間を指定する。

3.5.2 パラメータの設定例

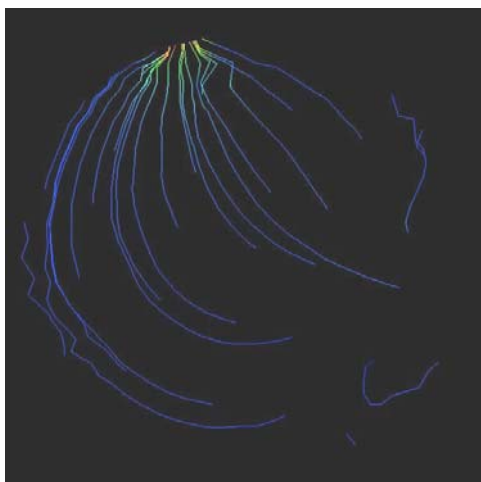
(1) !tracing_direction

トレース方向 (省略値: 1)

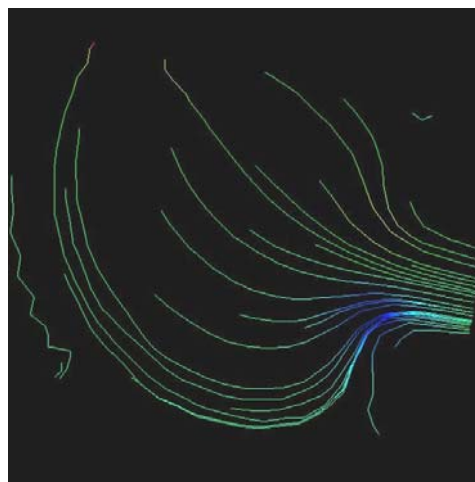
- 1: 後方へ
- 1: 前方へ
- 2: 両方の前後方向



!tracing_direction=2



!tracing_direction=1



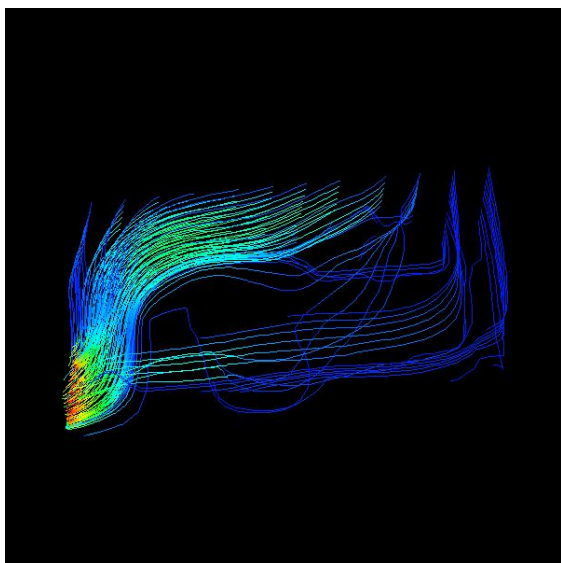
!tracing_direction=-1

図 44: tracing_direction の設定例

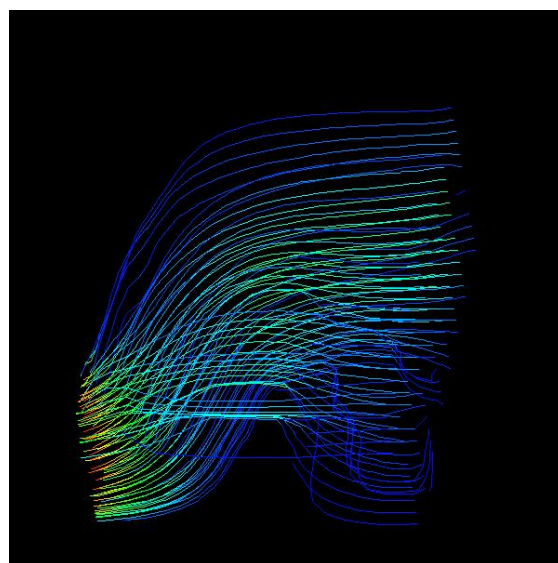
(2) !seed_style

種まきのスタイル

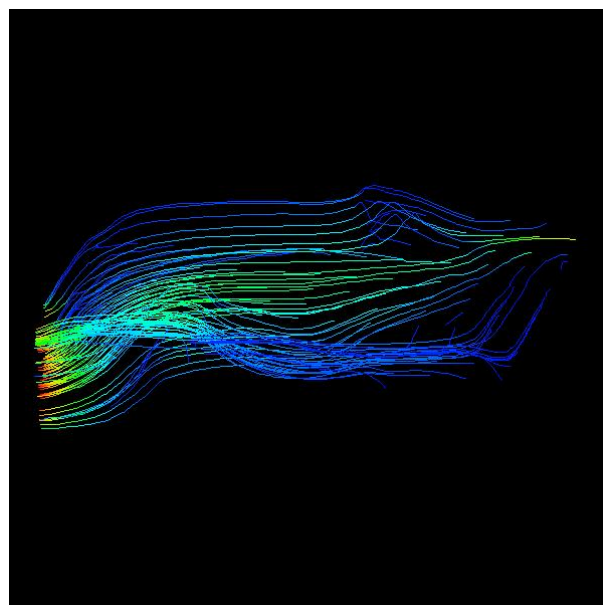
- 1: defined by users directly
- 2: defined in some file
- 3: semi-automatic by YZ plane
- 4: semi-automatic by XZ plane
- 5: semi-automatic by XY plane



!seed_style =3



!seed_style= 4



!seed_style =5

図 45: seed_style の設定例

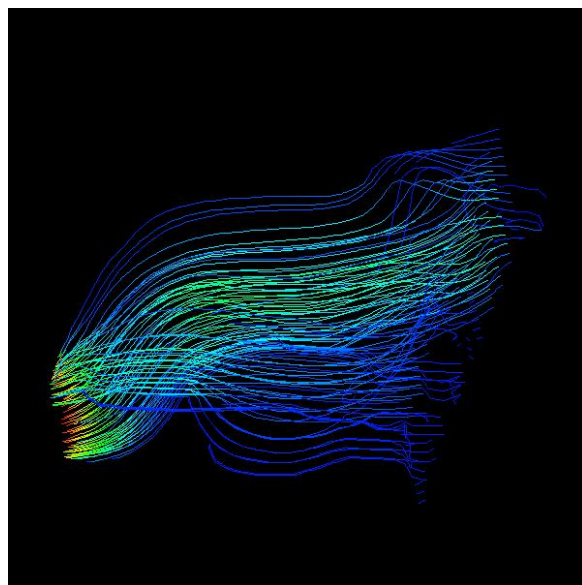
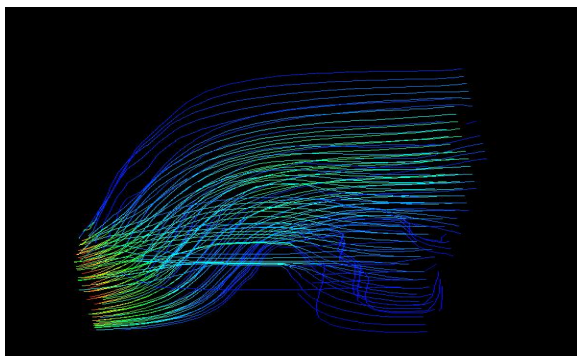
(3) !seed_plane_position

種子平面の位置

if seed_style=3, then z=seed_plane_position

if seed_style=4, then x=seed_plane_position

if seed_style=5, then y=seed_plane_position



!seed_plane_position=default

(in the middle)

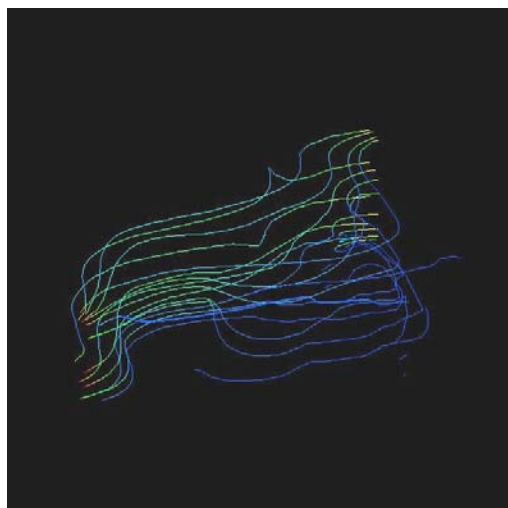
!seed_plane_position=1.5

図 46: seed_plane_position の設定例

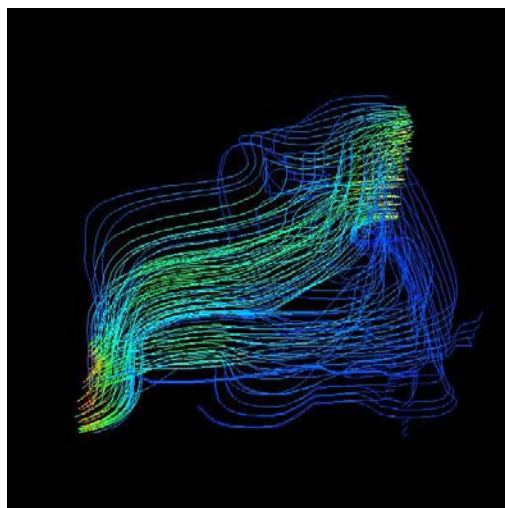
(4) !seed_plane_position

種子ポイントの数

(省略値:25)



!seed_num=default



!seed_num=144

図 47: seed_num の設定例

3.6 エラーメッセージ

HEC-MW-VIS-E0001: There is no enough memory allocated for variable “...”

変数 “...” のアロケーションエラー

HEC-MW-VIS-E0002: The control parameter format error: should start from !

コントロールパラメータ “!” の書式エラー

HEC-MW-VIS-E0003: The control parameter format error: no integer value for variable “...”

コントロールパラメータの書式エラー、“...” に整数値を設定する。

HEC-MW-VIS-E0004: Variable “...” should be an integer

変数 “...” が整数でない

HEC-MW-VIS-E0005: Variable “...” should be a real

変数 “...” が数値でない

HEC-MW-VIS-E0006: The control parameter format error: visual ID

識別番号のコントロールパラメータエラー

HEC-MW-VIS-E0007: The control parameter format error: method only can be PSR or PVR

コントロールパラメータエラー：method で PSR 又は PVR 以外が指定された

HEC-MW-VIS-E0008: The control parameter format error: surface_num should be defined

コントロールパラメータエラー、surface_num が設定されていない

HEC-MW-VIS-E0009: Cannot open output file

出力ファイルが開けない

HEC-MW-VIS-E0010: Cannot open voxel file

ボクセルファイルが開けない

HEC-MW-VIS-E0011: Cannot open control file

コントロールファイルが開けない

HEC-MW-VIS-E1001: output_type should be defined as AVS or BMP

出力形式が AVS 、BMP 以外である。

HEC-MW-VIS-E1002: iso_value should be defined for isosurfaces

等値面の値が設定されていない

HEC-MW-VIS-E1003: The 10 coefficients of the equation should be defined

2 次曲面の 10 個の係数が設定されていない

HEC-MW-VIS-E1004:display_method is not in the reasonable range

表示画面の範囲が妥当でない

HEC-MW-VIS-E1005: The x_resolution should be greater than 20

x 方向の解像度が 20 以下である

HEC-MW-VIS-E1006: The y_resolution should be greater than 20

y 方向の解像度が 20 以下である

HEC-MW-VIS-E1007: The number of light sources should be greater than 0

照明の個数が正しく設定されていない

HEC-MW-VIS-E1008: The ambient_coef is not correct. Should be ≥ 0.0

周囲の明るさの指定が正しくない

HEC-MW-VIS-E1009: The diffuse_coef is not correct. Should be ≥ 0.0

乱反射の強さの指定が正しくない

HEC-MW-VIS-E1010: The specular_coef is not correct. Should be ≥ 0.0

鏡面反射の強さの指定が正しくない

HEC-MW-VIS-E1011: color_mapping_style should be between 1 and 4

color_mapping_style の指定が 1 ～4 以外である

HEC-MW-VIS-E1012: For color_mapping_style 3, interval_mapping_num is required

color_mapping_style=3なのにinterval_mapping_numが指定されていない

HEC-MW-VIS-E1013: For color_mapping_style 3, the interval_mapping_num should be greater than 0

color_mapping_style=3なのにinterval_mapping_numが正しく設定されていない

HEC-MW-VIS-E1014: For color_mapping_style =2 or 3, the interval_mapping should be defined

color_mapping_style =2 or 3なのにinterval_mappingが設定されていない

HEC-MW-VIS-E1015: The parameter num_of_frames cannot be less than 1

num_of_frames が1より小さい

HEC-MW-VIS-E1016: color_mapping_bar_on parameter only can be defined as 0 or 1

color_mapping_bar_on で0又は1以外が指定されている

HEC-MW-VIS-E1017: scale_marking_on parameter only can be defined as 0 or 1

scale_marking_on で0又は1以外が指定されている

HEC-MW-VIS-E1018: x resolution should be larger than 40 for adding color mapping bar case

カラーバー表示の場合はx方向の解像度は40以上でなければならない

HEC-MW-VIS-E1019: x resolution should be larger than 65 for adding color mapping bar and scale marking case

カラーバーと目盛表示の場合はx方向の解像度は65以上でなければならない

HEC-MW-VIS-E1020: color_system_type should be between 1 and 3

color_system_type 1 ~3 以外である

HEC-MW-VIS-E1021: background color should be in the interval of (0.0, 1.0)

背景色の色が区間 (0.0, 1.0)以外である

HEC-MW-VIS-E1022: The font color should be in the interval of (0.0, 1.0)

フォントの色が区間 (0.0, 1.0)以外である

HEC-MW-VIS-E1023: font_size should be between 1.0 and 4.0

フォントサイズが区間 (1.0,4.0)以外である

HEC-MW-VIS-E1024: color_bar_style only can be 1 or 2

color_bar_style 1 又は 2 以外である

HEC-MW-VIS-E1025: num_of_scale only can be greater than 0

num_of_scale が正しく設定されていない

HEC-MW-VIS-E1026: range_value is required for fixed_range_on =1

fixed_range_on =1 なのに range_value が設定されていない

HEC-MW-VIS-E1027: mark_0_on only can be 0 or 1

mark_0_on 0 又は 1 以外である

HEC-MW-VIS-E1028: transfer_function_style should be between 1 and 8

伝達関数形式が 1 ~8 以外である

HEC-MW-VIS-E1029: opacity_value cannot be less than 0.0

opacity_value が正しく設定されていない

HEC-MW-VIS-E1030: When transfer_function_style =3 or 4, num_of_features should be defined

transfer_function_style =3 又は 4 なのに num_of_features が設定されていない

HEC-MW-VIS-E1031: When transfer_function_style =3 or 4, num_of_features should be greater than 0

transfer_function_style =3 又は 4 なのに num_of_features が正しく設定されていない

HEC-MW-VIS-E1032: For transfer_function_style =3 or 4, fea_point should be defined

transfer_function_style =3 又は 4 なのに fea_point が設定されていない

HEC-MW-VIS-E1033: For transfer_function_style=8, lookup_filename should be specified

transfer_function_style=8なのに lookup_filename が指定されていない

HEC-MW-VIS-E1034: histogram_on value should be 0,1, or 2

histogram_on で 0,1, 又は、2 以外が指定されている。

HEC-MW-VIS-E1036: time_mark_on value should be 0 or 1

time_mark_on が 0 又は 1 以外である

HEC-MW-VIS-E1037: fixed_scale_on value should be 0 or 1

fixed_scale_on が 0 又は 1 以外である

HEC-MW-VIS-E1038: maximum_refinement parameter should be greater than 0

maximum_refinement が正しく設定されていない

HEC-MW-VIS-E1039: rotate_style parameter is wrong. Please input one within (0, 4)

rotate_style parameter が 0~4 以外である

HEC-MW-VIS-E1040: please define all the n_voxel_x, n_voxel_y, n_voxel_z parameters

n_voxel_x, n_voxel_y 又は n_voxel_z が設定されていない

HEC-MW-VIS-E1041: n_voxel_x*n_voxel_y*n_voxel_z should be equal to the total number of PEs

n_voxel_x*n_voxel_y*n_voxel_z がプロセス数になっていない

HEC-MW-VIS-E1042: n_voxel_x, n_voxel_y, and n_voxel_z cannot be less or equal to 0

n_voxel_x, n_voxel_y, and n_voxel_z が正しく設定されていない

HEC-MW-VIS-E1043: fixed_range_on only can be 0 or 1

fixed_range_on 0 又は 1 以外である

HEC-MW-VIS-E1045: remove_0_display_on only can be 0 or 1

remove_0_display_on 0 または 1 以外である

HEC-MW-VIS-E1046: x_specified_level should be greater than 0

x_specified_level が正しく設定されていない

HEC-MW-VIS-E1047: y_specified_level should be greater than 0

y_specified_level が正しく設定されていない

HEC-MW-VIS-E1048: z_specified_level should be greater than 0

z_specified_level が正しく設定されていない

HEC-MW-VIS-E1051: The name for data component is not correct

データコンポーネント名が正しくない

HEC-MW-VIS-E1052: The subcomponent name is not correct, it must be norm, x, y, or z

サブコンポーネント名が norm, x, y, or z 以外である

HEC-MW-VIS-E1053: The name for color component is not correct

カラーコンポーネント名が正しくない

HEC-MW-VIS-E1054: color_comp is wrong: >nn_component

color_comp が正しくない

HEC-MW-VIS-E1055: color_subcomp is wrong: >dof

color_subcomp が自由度数より大きい

HEC-MW-VIS-E1055: data component number is wrong: >nn_component or <0

データコンポーネント番号が正しくない

HEC-MW-VIS-E1057: data_subcomp is wrong: >dof

data_subcomp が自由度数より大きい

HEC-MW-VIS-E1058: the number of method must be between 1 and 5

method が 1～5 以外である

HEC-MW-VIS-E1059: the number of isolines should be ≥ 0

isolines が正しく設定されていない

HEC-MW-VIS-E1060: deform_display_on only can be 0 and 1

deform_display_on が 0 又は 1 以外である

HEC-MW-VIS-E1061: The component name for deformation display is not correct

変位表示のためのコンポーネント名が正しくない

HEC-MW-VIS-E1062: deform_comp is wrong: $>nn_component$

deform_comp が自由度数以上である

HEC-MW-VIS-E1063: deform_comp is wrong: $>nn_component$ or <0

deform_comp が正しく設定されていない

HEC-MW-VIS-E1064: deform_scale should be greater than 0

deform_scale が正しく設定されていない

HEC-MW-VIS-E1065: initial_style for deformation display only can be 0-4

initial_style で 0~4 以外が設定されている

HEC-MW-VIS-E1066: deform_style for deformation display only can be 0-4

deform_style で 0~4 以外が設定されている

HEC-MW-VIS-E1067: The initial_line_color in deformation display should be in the interval of (0.0, 1.0)

変位表示において initial_line_color が区間(0.0, 1.0)以外である

HEC-MW-VIS-E1068: The deform_line_color in deformation display should be in the interval of (0.0, 1.0)

変位表示において deform_line_color が区間(0.0, 1.0)以外である

HEC-MW-VIS-E2001: There is something wrong with transform matrix, invers =0

transform 行列が非正則である

HEC-MW-VIS-E2002: The viewpoint position is not correct

視点の設定が正しくない

HEC-MW-VIS-E2003: Cannot generate histogram graph. The number of voxels is 0

ボクセル数が 0 のため柱状グラフが作れない

HEC-MW-VIS-E2004: Voxel search error in voxel refinement in PVR

PVR においてボクセルサーチでエラーが発生した

HEC-MW-VIS-E2005: Data communication error in refinement subroutine in PVR

PVR において通信エラーが発生した

HEC-MW-VIS-E2006: There is something wrong on finding the first intersection point in PVR

PVR において初めの交点が見つからない

HEC-MW-VIS-E2007: There is some problem in finding intersection points in PVR

PVR において交点が見つからない

HEC-MW-VIS-E3001:streamline_style only can be 1,2 or 3

streamline_style で 1~3 以外が設定されている

HEC-MW-VIS-E3002:tracing_direction only can be -1, 1 or 2

tracing_direction で-1,1,2 以外が設定されている

HEC-MW-VIS-E3003:seed_style should be within 1-- 6

seed_style で 1~6 以外が設定されている

HEC-MW-VIS-E3004:seed_num should be >=1

seed_num で<1 が設定されている