JAERI-Data/Code
2002-017

地球シミュレータ用可視化システム利用手引書

2002年8月

村松　一弘・齋　和憲

日本原子力研究所
Japan Atomic Energy Research Institute

本レポートは、日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは、日本原子力研究所研究情報部研究情報課（〒319-1195　茨城県那珂郡東海村）あて、お申し越し下さい。なお、このほかに財団法人原子力弘済会資料センター（〒319-1195　茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布を行っております。

This report is issued irregularly.
Inquiries about availability of the reports should be addressed to Research Information Division, Department of Intellectual Resources, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 〒319-1195, Japan.

編集兼発行　日本原子力研究所

JAERI－Data/Code 2002－017

地球シミュレータ用可視化システム利用手引書

日本原子力研究所計算科学技術推進センター
村松 一弘・齋 和憲

(2002年7月1日受理)

地球シミュレータ用の可視化システムを開発した。地球シミュレータ上でのシミュレーションと同時にクライアントにおいてその結果を視覚化することができ、計算を行っている最中に、その計算及び可視化の為のパラメータを変更することも可能である。グラフィカルユーザインターフェース (GUI) は Java applet で構築されており、そのためウェブブラウザさえあればよく、OS に非依存である。本システムはサーバ機能、ポストプロセッシング機能、クライアント機能で構成されている。サーバ機能とポストプロセッシング機能は地球シミュレータ上で動作し、クライアント機能はユーザ端末上で動作する。サーバ機能はライブラリ形式を採用しており、ユーザは容易に自分のコードに実時間可視化機能を組み込むことができる。ポストプロセッシング機能もライブラリ形式を採用しているが、ポストプロセッシング機能をもつロードモジュールも提供されている。本稿ではサーバ機能及びポストプロセッシング機能の使い方を中心に報告する。

日本原子力研究所: 〒110-0015 東京都台東区東上野 6-9-3

JAERI−Data/Code 2002−017

Visualization System on the Earth Simulator User's Guide

Kazuhiro MURAMATSU and Kazunori SAI

Center for Promotion of Computational Science and Engineering
Japan Atomic Energy Research Institute
Higashiueno, Taito-ku, Tokyo

(Received July 1, 2002)

A visualization system on the Earth Simulator is developped. The system enables users to see a graphic representation of simulation results on a client terminal simultaneously with them being computed on the Earth Simulator. Moreover, the system makes it possible to change parameters of the calculation and its visualization in the middle of calculation. The graphical user interface(GUI) of the system is constructed on a Java applet. Consequently, the client only needs a web browser, so it is independent of operating systems. The system consists of a server function, post-processing function and client function. The server and post-processing functions work on the Earth Simulator, and the client function works on the client terminal. The server function employs a library style format so that users can easily invoke real-time visualization functions by applying their code. The post-processing function employs a library style format and moreover provides a load module. This report describes mainly the usage of the server and post-processing functions.

Keywords: Earth Simulator, Real-time Visualization, Post-processing Visualization, Parallel Processing, Tracking and Steering, Java Applet, NetCDF, GrADS, AVS Field Data, RMV Format.

目 次

Contents

1 概要

実時間可視化システムは、流体解析や構造解析などの解析ソルバを実行しながら、解析結果をリアルタイムに可視化する。また、既存の計算結果ファイルを読み込んで可視化する。さらにGUI(Graphical User Interface) を通して、計算の途中で可視化のための各種パラメータを変更したり、解析ソルバ自体の制御を行うことができる。

以下では、実時間可視化システムの概要について説明する。

1.1 目的

計算機の高速化、数値解析技術の高度化とともに、計算結果をコンピュータグラフィックスにより可視化する技術の重要性が増している。従来可視化というと、解析計算終了後、計算結果ファイルを読み込んで可視化を行うポストプロセッシングを意味していた。これに対して、計算と同時に可視化 (実時間可視化) を行いたいというニーズが高まりつつある。その背景として、以下の点が挙げられる。

- 高速計算機の低価格化
- 解析の大型化による、大容量データ格納装置の必要性
- ネットワーク分散環境の普及
- デジタル動画の普及

以上のような背景の下、実時間可視化システムは、計算機上で様々な解析を行う人々に対し、以下の可視化機能を提供することを目的とする。

- 解析と可視化の同時実行 (トラッキング)
- 任意の解析コードからの可視化
- ネットワーク分散環境下での可視化
- デジタル圧縮動画による、アニメーション表示
- 可視化された計算結果を見ながらの、解析パラメータの制御 (ステアリング)

また、既存の計算結果ファイルを利用できるように、従来型のポストプロセッシング機能も提供する。

本システムの利用により、以下に代表される効果が期待できる。

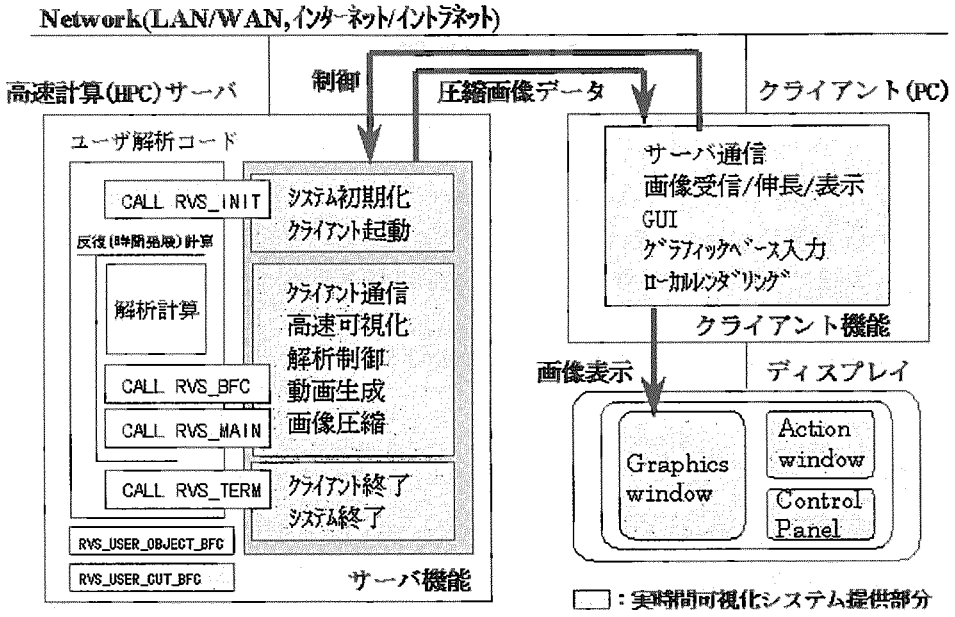


図 1-1: システム構成

- 定常解析における解の収束過程や非定常解析における解の時間変化などを、リアルタイムに確認できる。
- 詳細な評価や分析が必要な部分の解析結果のみファイル出力していくことにより、解析結果保存のためのディスク容量が削減できる。
- 解析結果に不具合が生じた場合は、計算を直ちに打ち切ることができる。
- 解析試行から評価のサイクルが効率化され、設計/開発コストが削減できる。

1.2 特徴

本システムはサーバ機能とポストプロセッシング機能とクライアント機能で構成される。本システムのサーバ機能とクライアント機能の構成を図 1-1 に示す。サーバ機能は解析プログラムが動作する高速計算サーバ上で、クライアント機能はユーザ端末上でそれぞれ動作する。ポストプロセッシング機能もサーバ機能と同様な構成であり、高速計算サーバ上で動作する。サーバ機能がユーザ解析コードに組み込まれるのに対し、ポストプロセッシング機能はポストプロセッシングプログラムの構成部品となる。

本システムの特徴は以下の通りである。

- トラッキング/ステアリング
  解析計算と同時に結果を可視化し、さらに可視化された解析結果を見ながら解析パラメータ

の制御ができる。

- 高速可視化処理
  解析コードのメモリ領域を直接参照するため、解析結果のファイルへの入出力が発生しない。また、ベクトル並列化された可視化アルゴリズムの採用により、ベクトル並列計算機を利用した場合はその性能を引き出すことができる。

- クライアントサーバ形式
  ネットワーク環境を利用し、端末から手軽に操作できる。

- 画像データの圧縮転送/保存
  サーバ機能は、可視化画像まで生成し、必要に応じて画像圧縮を行う。これにより、転送データ量が削減され、ネットワーク負荷が軽減される。また、画像保存に必要なファイル容量も削減される。

- ライブラリ形式
  サーバ機能はライブラリ形式により提供される。これを呼び出すことにより、様々な解析コードから手軽に利用することができる。

- グラフィカルユーザインタフェース (GUI)
  クライアント機能からメニュー形式による操作ができる。さらにマウスを用いて、表示画像の操作や可視化パラメータの値入力ができる。

1.3 機能

本システムの機能概要を以下に記述する。

1.3.1 表示図の種類

本システムでは、トレーサ、等高線図、オブジェクト、ベクトル図、流線図、格子図、等値面図、地形図およびそれらの重ね合わせ (等高線図、ベクトル図は複数格子面/断面) が表示できる。ただし、地形図はポストプロセッシング機能でのみ使用できる。各図種の表示/非表示、表示の対象となる物理量の選択等のセットアップおよび各図種固有のパラメータ値の指定は、GUI のメニューにより行う。ボリュームレンダリングは現在のところ未実装である。

1.3.2 共通パラメータ

各図種に共通なパラメータとして、カラーモード、カラーテーブル、背景色、視点、光源が設定できる。

1.3.3 システムモード

本システムには以下に示す4つのシステムモードがある。

- リアルタイムモード
  解析ソルバの演算と可視化処理の両方が実行される。可視化処理が行われるステップ間隔も変更可能である。
- 可視化表示中断モード
  可視化処理は中断され、解析ソルバの演算だけが実行される。
- 解析ソルバ中断モード
  解析ソルバの演算が中断され、可視化処理だけが実行される。
- 解析ソルバ・可視化表示中断モード
  解析ソルバの演算と可視化処理の両方が中断される。

1.3.4 可視化画像の操作

可視化表示された計算結果をマウスなどでポイントすることにより、表示図の回転、移動、サイズ変更等が行える。さらに、GUI メニュー上の可視化パラメータの一部は、可視化された計算結果をポイントすることにより入力することが可能である。

1.3.5 パラメータ設定ファイル

ユーザは、本システムのパラメータをファイルに保存したのち、そのファイルを読み込ませることで、パラメータ保存時点での状態を再現できる。このパラメータ設定ファイルは、本システム起動の際に読み込ませることができる。

1.3.6 画像データの保存

画像データを各解析ステップごとに静止画または動画としてファイルへ保存しておくことができる。さらに解析途中でも、それまでに作成された動画の再生ができる。

1.3.7 ポストプロセッシング

NetCFD 形式、GrADS 形式、AVS Field Data のファイルを読み込み可視化処理を行うことができる。また、本機能は実行モジュールとライブラリ形式の2つの形式で提供される。実行モジュールで簡単に本機能を利用可能であると同時に、ライブラリを利用したカスタマイズも行うことができる。

1.3.8 処理方式

処理方式として、ネットワーク上の2台の計算機で演算と表示・操作の処理を分割して担当する分散処理モードと、ユニバーサルシナリオファイルに従った可視化処理をバッチ的に行うバッチ処理モードのいずれかを選択してシステムを起動できる。分散処理モードでは、一方の計算機上で解析ソルバおよび可視化処理の実行を、もう一方の計算機(端末)上では画像データの表示およびパラメータ操作を行う。バッチ処理モードでは、解析ソルバおよび可視化処理の実行のみが一つのマシン上で行われ、画像データがファイル出力される。

2　ライブラリの組み込み

2.1　ライブラリ組み込みの概要

実時間可視化システムのサーバ機能は、ライブラリ形式で提供される。またポストプロセッシング機能は実行モジュールとライブラリ形式で提供される。このライブラリは、本システムが提供する実時間可視化機能を、一般の解析ソルバから利用できるようにするため、あるいは実行モジュールとしても提供されているポストプロッセッシング機能を独自に作成できるようにするために用意されているものである。ライブラリは7個の組み込み用サブルーチンで構成されている。また、可視化に必要なパラメータの一部をソルバのパラメータ内容に応じて与えるためにユーザ関数とユーティリティ関数を用意している。

一般の解析ソルバから本システムを利用するためには、解析プログラムの中にライブラリ呼び出し部分を設けるとともに、ライブラリへ渡すデータのフォーマット変換、ユーザ関数のコーディング、ユーティリティ関数の呼び出しを必要に応じて行う必要がある。

また、ポストプロセッシング機能を独自に作成する場合、ライブラリ呼び出し部分を含むプログラムを作成する必要がある。ユーザ関数のコーディング、ユーティリティ関数の呼び出しもサーバ機能利用時と同様に必要に応じて行う必要がある。

本章では、一般の解析ソルバから本システムを利用するための方法やポストプロセッシングを独自に作成するための方法について、ライブラリおよびユーザ関数のインタフェース仕様とともに説明する。

2.2　ライブラリ仕様

実時間可視化システムのサーバ機能を利用するためには、必要な組み込み用サブルーチンを解析プログラムから呼び出し、必要に応じてユーザ関数のコーディングとユーティリティ関数の呼び出しを行わなければならない。またポストプロセッシング機能を作成するためには、必要な組み込み用サブルーチンを用いてプログラムを作成しなければならない。本節では組み込み用サブルーチン、ユーザ関数、およびユーティリティ関数の仕様について述べる。

本システムのサーバ機能は、以下のデータを可視化の対象とする。

- 3次元 BFC(Boundary Fitted Coordinate) 格子上に定義された数値データ
- 非構造四面体格子上に定義された数値データ
- 非構造六面体格子上に定義された数値データ

また本システムのポストプロセッシング機能は、以下のデータファイルを可視化の対象とする。

- NetCDF(Network Common Data Form) 形式のデータファイル
- GrADS(Grid Analysis and Display System) 形式のデータファイル
- AVS Field Data のデータファイル

NetCDF 形式やポストプロセッシング機能での対応については、付録 D を参照。
GrADS 形式やポストプロセッシング機能での対応については、付録 E を参照。
AVS Field Data やポストプロセッシング機能での対応については、付録 F を参照。
以下の説明では、3 次元 BFC 格子については 3 つの主軸方向を、I、J、K 方向とする。また、各格子が定義された物理空間の主軸方向を、X、Y、Z 方向とする。
なお、組み込み用サブルーチン、ユーザ関数、およびユーティリティ関数の引数一覧表の見方は以下の通りである。

引数名 ライブラリ引数の名称が記載される。

型 引数のデータ型を示す。I4 は整数型を、R4 は単精度実数型を、R8 は倍精度実数型を、A は文字型を、A*n (n は整数) は長さ n の文字列をそれぞれ表す。

サイズ 引数の大きさを表す。サイズが 1 の場合は変数である。サイズが 1 でない場合は配列であり、配列の次元数と各次元の寸法を表している。例えばサイズが n(>1) の場合、寸法が n の 1 次元配列であることを示す。また、サイズが (n1,n2) の場合 (n1、n2 は整数)、第 1 次元の寸法が n1、第 2 次元の寸法が n2 の 2 次元配列であることを示す。

入出力 引数が、組み込み用サブルーチン、ユーザ関数、またはユーティリティ関数へ何らかのデータを入力するために用いられるのか出力するために用いられるのかを示す。

内容 引数が保持すべき情報について説明している。

2.2.1 組み込み用サブルーチン

サーバ機能の組み込み用サブルーチンには、初期化用、各データタイプに対するデータ配列 (格子データ、解析結果など) のアドレス設定用、メイン処理 (可視化、制御) 用、終了用の 6 つがある。例えば 3 次元 BFC 格子上に定義された数値データの可視化を行う場合、解析ソルバの処理と組み込み用サブルーチン呼び出し位置の関係は、概ね図 2-1 のようになる。
他のデータタイプを可視化の対象とする場合、これに応じたアドレス設定用のサブルーチンを RVS_BFC の代わりに呼び出す。
ポストプロセッシング機能の組み込み用サブルーチンを用いてポストプロセッシングモジュールを作成する場合、以下のように組み込み用サブルーチンを呼び出すプログラムを作成すればよい。

```
#include<stdio.h>
#include<stdlib.h>
```

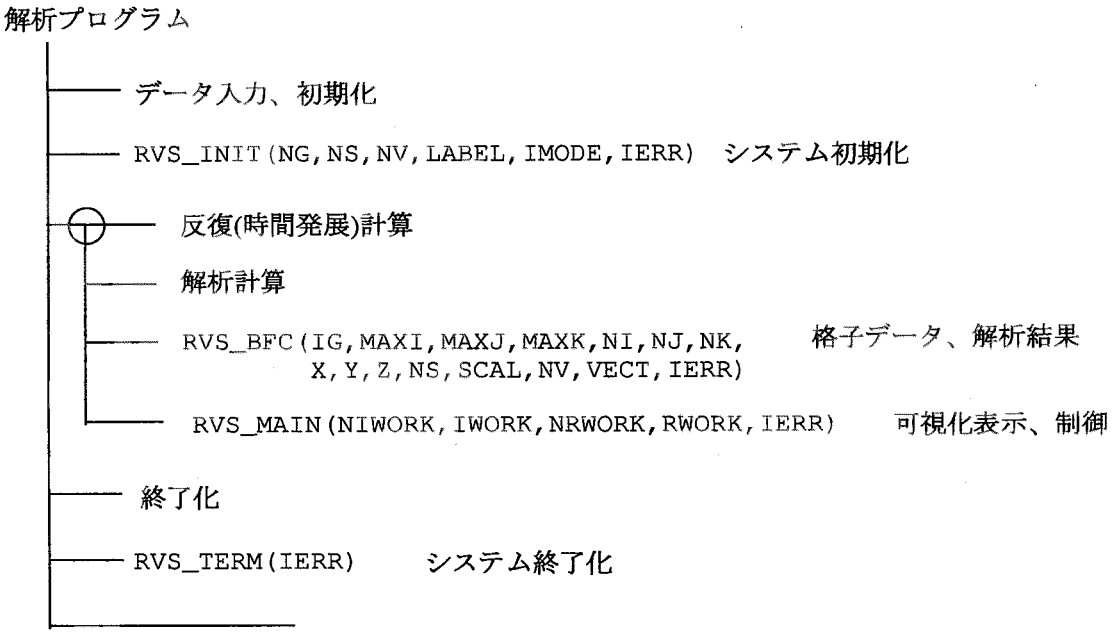


図 2-1: 組み込み用サブルーチン呼び出し構造

```
main(argc,argv)
int argc;
char *argv[];
{
   char *postfilewk ;
   char *parafilewk ;
   char postfile[256]  ;
   char parafile[256]  ;
   int imode ;
   int ierr = 0 ;
   if (argc != 4)
   {printf("Argment Error\n");
    exit(-1);
   }
   imode = atoi(argv[1]);
   postfilewk = argv[2];
   parafilewk = argv[3];
   strcpy(postfile,postfilewk);
   strcpy(parafile,parafilewk);
   rvs_postlib_(&imode,&postfile,&parafile,&ierr);
```

```
    rvs_term_(&ierr);
}
```

本プログラムは３つの引数を持ち、その引数の内容をポストプロセッシング機能の組み込み用サブルーチンに渡している。第一引数は実時間可視化システムの実行モード。第二引数はNetCDFの入力ファイル、GrADSのデータ記述ファイル、AVS Field Dataファイルなどのファイル名。第三引数は可視化に必要なパラメータ設定ファイルのファイル名である。この順番に引数を定義しておくと、クライアント機能からファイル名を選択して自動起動ができるようになっている。

ポストプロセッシング機能の組み込み用サブルーチンは、内部的にサーバ機能と同様のサブルーチンを利用しているため、最後にrvs_term_を呼び出す必要がある。

また、前バージョンにあったNetCDF専用サブルーチンrvs_netcdf_、GrADS専用サブルーチンrvs_grads_も利用可能である。いずれの場合も、サブルーチン引数は同じである。

2.2.1.1 RVS_INIT

(1) 機能

実時間可視化システムのサーバ機能を初期化する。

(2) 呼び出し形式

```
CALL RVS_INIT(NG,NS,NV,LABEL,IMODE,IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
NG	I4	1	入力	全ブロック数
NS	I4	1	入力	スカラデータ数
NV	I4	1	入力	ベクトルデータ数
LABEL	A*15	NS+NV	入力	物理量種別名
IMODE	I4	1	入力	実行モード
IERR	I4	1	出力	エラーコード

(4) 補足説明

- NG
  計算格子としてマルチブロック格子を用いる場合、NG にはブロック数を指定する。シングルブロック格子を用いる場合は、1 を指定する。本システムではマルチブロック格子をサポートしていないため、必ず 1 を指定する。
- LABEL
  LABEL には、各スカラデータおよびベクトルデータの意味内容を表すラベルを 15 文字以下の文字列で指定する。なお、ラベルは空白であってはならない。スカラデータおよびベクトルデータに対するラベルを、RVS_MAIN へデータを渡す順序で、(NS+NV) 個分入力する。本システムの GUI 上で、どのデータを可視化の対象とするかを、ここで指定したラベルの中から選択できるようになる。
- IMODE
  IMODE が 0 の場合、バッチ処理モードで起動される。ユニバーサルシナリオファイルに従って可視化処理が行われ、画像ファイルが作成される。IMODE が 1 の場合、分散処理モードで起動される。クライアントサーバ型の処理が行われ、クライアント側から随時パラメータを変更することができる。
- IERR
  エラーコードの詳細については、2.2.6 節参照。

2.2.1.2 RVS_BFC

(1) 機能

3次元BFC格子上に定義された数値データを可視化の対象とする場合の、格子データおよび解析結果の配列のアドレスを引き渡す。

(2) 呼び出し形式

```
CALL RVS_BFC(IG,MAXI,MAXJ,MAXK,NI,NJ,NK,X,Y,Z,NS,SCAL,NV,VECT,IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IG	I4	1	入力	ブロック番号
MAXI	I4	1	入力	I軸方向の整合寸法
MAXJ	I4	1	入力	J軸方向の整合寸法
MAXK	I4	1	入力	K軸方向の整合寸法
NI	I4	1	入力	I軸方向の格子点数
NJ	I4	1	入力	J軸方向の格子点数
NK	I4	1	入力	K軸方向の格子点数
X	R8	(MAXI,MAXJ,MAXK)	入力	格子点のX座標値
Y	R8	(MAXI,MAXJ,MAXK)	入力	格子点のY座標値
Z	R8	(MAXI,MAXJ,MAXK)	入力	格子点のZ座標値
NS	I4	1	入力	スカラデータ数
SCAL	R8	(MAXI,MAXJ,MAXK,NS)	入力	スカラデータ
NV	I4	1	入力	ベクトルデータ数
VECT	R8	(MAXI,MAXJ,MAXK,3,NV)	入力	ベクトルデータ
IERR	I4	1	出力	エラーコード

(4) 補足説明

- IG
  各格子ブロックに対して、固有の番号付けを行う。この番号は、1から全格子ブロック数NGの範囲で指定する。本システムではマルチブロック格子をサポートしていないため、IGとして必ず1を指定する。
- MAXI、MAXJ、MAXK
  格子点の物理座標値および格子点上での解析結果(スカラ/ベクトルデータ)を格納するための配列(X、Y、Z、SCAL、VECT)の、I、J、K各方向の整合寸法を指定する。
- NI、NJ、NK
  I、J、K各方向の格子点数を指定する。

- X、Y、Z、SCAL、VECT
  格子点の物理座標値および格子点上での解析結果(スカラ/ベクトルデータ)を格納する。これらは整合配列である。X(i,j,k)、Y(i,j,k)、Z(i,j,k)には、格子点(i,j,k)の物理座標値を格納する。SCAL(i,j,k,n)には、格子点(i,j,k)におけるn番目のスカラデータのデータ値を格納する。VECT$(i,j,k,1,n)$、VECT$(i,j,k,2,n)$、VECT$(i,j,k,3,n)$には、格子点(i,j,k)におけるn番目のベクトルデータのX成分、Y成分、Z成分を格納する。
- IERR
  エラーコードの詳細については、2.2.6節参照。
- 解析ソルバの格子点座標値およびスカラ/ベクトルデータ格納形式が本システムの形式と異なる場合、RVS_BFCを呼ぶ前に値の並び換えを行う必要がある。
- 本サブルーチンは、RVS_TET1、またはRVS_HEX1と同時に呼び出すことはできない。

2.2.1.3 RVS_TET1

(1) 機能

非構造四面体格子上に定義された数値データを可視化の対象とする場合の、格子データおよび解析結果の配列のアドレスを引き渡す。

(2) 呼び出し形式

```
CALL RVS_TET1(IG,MNOD,MELM,NNOD,NELM,CORD,IELM,NS,SCAL,NV,VECT,IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IG	I4	1	入力	ブロック番号
MNOD	I4	1	入力	節点数の整合寸法
MELM	I4	1	入力	要素数の整合寸法
NNOD	I4	1	入力	全節点数
NELM	I4	1	入力	全要素数
CORD	R8	(3,MNOD)	入力	節点のX、Y、Z座標値
IELM	I4	(4,MELM)	入力	要素内構成節点番号
NS	I4	1	入力	スカラデータ数
SCAL	R8	(MNOD,NS)	入力	スカラデータ
NV	I4	1	入力	ベクトルデータ数
VECT	R8	(MNOD,3,NV)	入力	ベクトルデータ
IERR	I4	1	出力	エラーコード

(4) 補足説明

- IG
  各格子ブロックに対して、固有の番号付けを行う。この番号は、1から全格子ブロック数NGの範囲で指定する。本システムではマルチブロック格子をサポートしていないため、IGとして必ず1を指定する。
- MNOD,MELM
  節点の物理座標値、要素内構成節点番号、および節点上での解析結果(スカラ/ベクトルデータ)を格納するための配列(CORD、IELM、SCAL、VECT)の整合寸法を指定する。
- NNOD、NELM
  それぞれ、全節点数、全要素数を指定する。
- CORD、IELM、SCAL、VECT
  節点の物理座標値、要素内構成節点番号、および節点上での解析結果(スカラ/ベクトルデータ)を格納する。これらは整合配列である。CORD$(1,i)$、CORD$(2,i)$、CORD$(3,i)$

には、i 番目の節点の物理座標値を格納する。IELM$(1,i)$、IELM$(2,i)$、IELM$(3,i)$、IELM$(4,i)$には、i番目の四面体要素を構成する4つの節点の番号を格納する。SCAL(i,n)には、i番目の節点上でのn番目のスカラデータのデータ値を格納する。VECT$(i,1,n)$、VECT$(i,2,n)$、VECT$(i,3,n)$には、i番目の節点上でのn番目のベクトルデータのX成分、Y成分、Z成分を格納する。

- IERR
  エラーコードの詳細については、2.2.6節参照。
- 解析ソルバの節点座標値、要素内構成節点番号、およびスカラ/ベクトルデータ格納形式が本システムの形式と異なる場合、RVS_TET1を呼ぶ前に値の並び換えを行う必要がある。
- 本サブルーチンは、RVS_BFC、またはRVS_HEX1と同時に呼び出すことはできない。

2.2.1.4 RVS_HEX1

(1) 機能

非構造六面体格子上に定義された数値データを可視化の対象とする場合の、格子データおよび解析結果の配列のアドレスを引き渡す。

(2) 呼び出し形式

```
CALL RVS_HEX1(IG,MNOD,MELM,NNOD,NELM,CORD,IELM,NS,SCAL,NV,VECT,IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IG	I4	1	入力	ブロック番号
MNOD	I4	1	入力	節点数の整合寸法
MELM	I4	1	入力	要素数の整合寸法
NNOD	I4	1	入力	全節点数
NELM	I4	1	入力	全要素数
CORD	R8	(3,MNOD)	入力	節点のX、Y、Z座標値
IELM	I4	(8,MELM)	入力	要素内構成節点番号
NS	I4	1	入力	スカラデータ数
SCAL	R8	(MNOD,NS)	入力	スカラデータ
NV	I4	1	入力	ベクトルデータ数
VECT	R8	(MNOD,3,NV)	入力	ベクトルデータ
IERR	I4	1	出力	エラーコード

(4) 補足説明

- IG
  各格子ブロックに対して、固有の番号付けを行う。この番号は、1から全格子ブロック数NGの範囲で指定する。本システムではマルチブロック格子をサポートしていないため、IGとして必ず1を指定する。
- MNOD,MELM
  節点の物理座標値、要素内構成節点番号、および節点上での解析結果(スカラ/ベクトルデータ)を格納するための配列(CORD、IELM、SCAL、VECT)の整合寸法を指定する。
- NNOD、NELM
  それぞれ、全節点数、全要素数を指定する。
- CORD、IELM、SCAL、VECT
  節点の物理座標値、要素内構成節点番号、および節点上での解析結果(スカラ/ベクトルデータ)を格納する。これらは整合配列である。CORD$(1,i)$、CORD$(2,i)$、CORD$(3,i)$

には、i番目の節点の物理座標値を格納する。IELM$(1,i)$、IELM$(2,i)$、IELM$(3,i)$、…、IELM$(8,i)$には、i番目の六面体要素を構成する8つの節点の番号を格納する。SCAL(i,n)には、i番目の節点上でのn番目のスカラデータのデータ値を格納する。VECT$(i,1,n)$、VECT$(i,2,n)$、VECT$(i,3,n)$には、i番目の節点上でのn番目のベクトルデータのX成分、Y成分、Z成分を格納する。

- IERR
  エラーコードの詳細については、2.2.6節参照。
- 解析ソルバの節点座標値、要素内構成節点番号、およびスカラ/ベクトルデータ格納形式が本システムの形式と異なる場合、RVS_HEX1を呼ぶ前に値の並び換えを行う必要がある。
- 本サブルーチンは、RVS_BFC、またはRVS_TET1と同時に呼び出すことはできない。

2.2.1.5 RVS_MAIN

(1) 機能

可視化および表示の制御を行う。

(2) 呼び出し形式

```
CALL RVS_MAIN(NIWORK,IWORK,NRWORK,RWORK,IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
NIWORK	I4	1	入力	整数作業領域配列サイズ
IWORK	I4	NIWORK	出力	整数作業領域
NRWORK	I4	1	入力	実数作業領域配列サイズ
RWORK	R8	NRWORK	出力	実数作業領域
IERR	I4	1	出力	エラーコード

(4) 補足説明

- NIWORK、NRWORK
  本システムの整数作業領域 (IWORK) および実数作業領域 (RWORK) のサイズである。確保すべきサイズについては、IWORK、RWORK の説明を参照のこと。これらの値として 0 を指定した場合、必要な整数または実数作業領域のサイズをシステムが自動的に計算し、作業領域を動的に確保する。
- IWORK、RWORK
  本システムの整数作業領域および実数作業領域である。本領域の入力値および出力値は意味をもたないため、解析ソルバの作業領域の併用も可能である。ただし、NIWORK または NRWORK に 0 を指定した場合、解析ソルバの作業領域の併用はできない。
  作業領域として、以下の大きさ以上の領域が必要である。

作業領域	必要な大きさ
IWORK	$Viewport\_size^2 \times$ (同時表示図種数 + 2) + 10 + int_size
RWORK	$Viewport\_size^2 \times$ (同時表示図種数 + 2) + $real\_size$

  ここで、*Viewport_size* はビューポートサイズ、*int_size* は各表示機能で使用する整数作業領域のサイズ、*real_size* は各表示機能で使用する実数作業領域のサイズである。
  各表示機能で使用する作業領域のサイズは、以下の通りである。

(a) 等高線図

サーバ機能

作業領域	表示条件	必要な大きさ
整数作業領域	カラーフリンジ表示	$Viewport\_size^2$
	等高線表示	$(Viewport\_size+1)^2 + Viewport\_size^2$
実数作業領域	座標面表示	$4 \times \max(NI \times NJ, NJ \times NK, NK \times NI)$
	任意断面表示	$6 \times Viewport\_size$

ポストプロセッシング機能

作業領域	表示条件	必要な大きさ
整数作業領域	座標面かつ カラーフリンジ表示	$Viewport\_size^2$
	断面かつ カラーフリンジ表示	$\max(Viewport\_size^2, 4 \times NI \times NJ)$
	座標面かつ 等高線表示	$(Viewport\_size+1)^2 + Viewport\_size^2$
	断面かつ 等高線表示	$\max((Viewport\_size+1)^2 + Viewport\_size^2, 4 \times NI \times NJ)$
実数作業領域	座標面表示	$4 \times \max(NI \times NJ, NJ \times NK, NK \times NI)$

(b) オブジェクト

作業領域	必要な大きさ
整数作業領域	$\max(isheet \times 80, max\_pol \times 12) + 782$
実数作業領域	$\max(isheet \times 200, max\_pol \times 45) + 57$

ここで、*isheet*は、$DI = LO(2,*) - LO(1,*) + 3$、$DJ = LO(4,*) - LO(3,*) + 3$、$DJ = LO(6,*) - LO(5,*) + 3$としたとき、$DI \times DJ + DJ \times DJ + DJ \times DI$の最大値である。LOについては、ユーザ関数RVS_USER_OBJECTの説明を参照のこと。*max_pol*は、ユーティリティ関数RVS_UTIL_OBJECT_POLYGON(ポストプロセッシング機能ではユーザ関数RVS_USER_OBJECT_POLYGON)で定義した各マテリアル上の三角形ポリゴンの最大数である。なお、ユーザ関数RVS_USER_OBJECTを利用しなかった場合は$isheet = 0$、ユーティリティ関数RVS_UTIL_OBJECT_POLYGON(ポスト可視化版ではユーザ関数RVS_USER_OBJECT_POLYGON)を利用しなかった場合は$max_pol = 0$と考える。

(c) トレーサ

作業領域	格子条件	必要な大きさ
整数作業領域	BFC 格子	最大トレーサ発生数
	非構造格子 (非並列版)	最大トレーサ発生数 ＋max(3 × 最大トレーサ発生数, (NELM − 1)/32 + 1)
	非構造格子 (並列版)	最大トレーサ発生数 ＋4 × NPROC + 30000
実数作業領域	BFC 格子	3 × 最大トレーサ発生数
	非構造格子 (非並列版)	6 × 最大トレーサ発生数
	非構造格子 (並列版)	14 × 最大トレーサ発生数

ここで、NPROC は非構造格子ブロックを処理する PE 群の PE 数である。

(d) ベクトル図

作業領域	表示条件	格子条件	必要な大きさ
整数作業領域	座標面表示	BFC 格子	max(NI × NJ, NJ × NK, NK × NI)
	任意断面表示	BFC 格子	0
		非構造格子	$Viewport\_size^2$ ＋4 × *Viewport_size*
実数作業領域	座標面表示	BFC 格子	6 × max(NI × NJ, NJ × NK, NK × NI)
	任意断面表示		0

(e) 格子図

作業領域	格子条件	必要な大きさ
整数作業領域	BFC 格子	0
実数作業領域	BFC 格子	6 × *max_num*

ここで、*max_num* は、格子を描く領域における IJK 各方向の格子線描画本数をそれぞれ *num_i*、*num_j*、*num_k* とするとき、max(*num_i*× *num_j*, *num_j*× *num_k*, *num_k*× *num_i*) の各領域における値のうちの最大値である。

(f) 流線図

作業領域	格子条件	必要な大きさ
整数作業領域	BFC 格子	4 × 流線の数の最大値
	非構造格子 (非並列版)	2 × 流線の数の最大値 ＋max(3 × 流線の数の最大値, (NELM − 1)/32 + 1)
	非構造格子 (並列版)	6 × 流線の数の最大値 + 4 × NPROC
実数作業領域	BFC 格子	12 × 流線の数の最大値
	非構造格子 (非並列版)	9 × 流線の数の最大値
	非構造格子 (並列版)	17 × 流線の数の最大値

ここで、NPROC は非構造格子ブロックを処理する PE 群の PE 数である。

(g) 等値面図

サーバ機能

作業領域	格子条件	必要な大きさ
整数作業領域	BFC 格子	$\max(105 \times (\mathrm{NI}-1), 6 \times \mathrm{NI} \times \mathrm{NJ})$
	非構造四面体格子	$(\mathrm{NELM}-1)/32$
	非構造六面体格子	$(6 \times \mathrm{NELM}-1)/32$
実数作業領域		0

ポストプロセッシング機能

作業領域	格子条件	必要な大きさ
整数作業領域	BFC 格子	$\max(105 \times (\mathrm{NI}-1), 6 \times \mathrm{NI} \times \mathrm{NJ})$
	非構造四面体格子	$(\mathrm{NELM}-1)/32+32768$
	非構造六面体格子	$(6 \times \mathrm{NELM}-1)/32+32768$
実数作業領域	BFC 格子	0
	非構造格子	98304

(h) 地形図

作業領域	格子条件	必要な大きさ
整数作業領域	BFC 格子	$(\mathit{Viewport\_size}+1)^2+\mathit{Viewport\_size}^2$
実数作業領域	BFC 格子	$4 \times \mathrm{NI} \times \mathrm{NJ}$

ここで、NI、NJ は、地形図固有の値である。NI は経度方向、NJ は緯度方向の範囲と刻み幅から求められる。

- IERR

  エラーコードの詳細については、2.2.6 節参照

(5) 注意事項

上記に加え、以下の領域が内部的に確保される。

- 非構造六面体格子上に定義された数値データの可視化を行う場合、$24 \times \mathrm{NELM}$ の大きさの整数作業領域が動的確保され、常時使用される。さらに、$4 \times \mathrm{NELM}+125000$ の大きさの整数作業領域が一時的に動的確保される。

2.2.1.6 RVS_TERM

(1) 機能

実時間可視化システムの実行を終了する。

(2) 呼び出し形式

```
CALL RVS_TERM(IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IERR	I4	1	出力	エラーコード

(4) 補足説明

- IERR
  エラーコードの詳細については、2.2.6 節参照。

2.2.1.7 RVS_POSTLIB

(1) 機能

実時間可視化システムのポストプロセッシング機能ルーチン。指定されたデータファイルを読み込み、実時間可視化システムのサーバ機能で可視化を行う。

(2) 呼び出し形式

```
CALL RVS_POSTLIB(IMODE,DATAFILE,PARAFILE,IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IMODE	I4	1	入力	実行モード
DATAFILE	A*255	1	入力	入力データファイル名
PARAFILE	A*255	1	入力	実時間可視化システムのパラメータ設定ファイル名
IERR	I4	1	出力	エラーコード

(4) 補足説明

- IMODE
  IMODE が 0 の場合、バッチ処理が行われる。ユニバーサルシナリオファイルに従って可視化処理が行われ、画像ファイルが生成される。IMODE が 1 の場合、分散処理モードで起動される。クライアントサーバ型の処理が行われ、クライアント側から随時パラメータを変更することができる。
- DATAFILE
  DATAFILE には、入力データファイルのファイル名を 255 文字以内の文字列として指定する。本システムでは、入力データファイルとして NetCDF 入力ファイル、GrADS のデータ記述ファイル、AVS Field Data ファイルが指定可能である。本ファイルの拡張子は、NetCDF 入力ファイルの場合は ".nc"、GrADS のデータ記述ファイルの場合は ".ctl"、AVS Field Data ファイルの場合は ".fld" としなければならない。
- PARAFILE
  PARAFILE には、上記 DATAFILE の可視化で用いる実時間可視化システムのパラメータ設定ファイルのファイル名を 255 文字以内の文字列として指定する。クライアント機能を用いて GUI からファイル名を指定できるが、その場合本ファイルの拡張子は ".para" としなければならない。
- IERR
  エラーコードの詳細については、2.2.6 節参照。

(5) 備考

前バージョンにあった NetCDF 専用サブルーチン rvs_netcdf_、GrADS 専用サブルーチン rvs_grads_も利用可能である。いずれの場合も、サブルーチン引数は同じである。

2.2.2 ユーザ関数 (サーバ機能)

ユーザ関数は、本システムが必要とするパラメータの一部を、解析ソルバのパラメータ内容に応じて与えるために用意されている。本システム利用の際には、必要に応じてユーザ関数をコーディングする。組み込み用サブルーチンはこれらの関数を内部的に呼び出し、コーディング内容に従ったパラメータ値を獲得する。

ユーザ関数を利用するためには、コーディングしたユーザ関数を含むファイルをコンパイルしておき、組み込みロードモジュールを作成する際に本システムのアーカイブファイルとともにリンクすればよい (第 3 章参照)。

図 2-2 にユーザ関数の役割を模式的に示す。一般にユーザ関数は、コモン変数を通して解析ソルバのパラメータ値を獲得し、これに応じて適切な値を関数の引数に設定する。組み込み用サブルーチンは、ユーザ関数を内部的に呼び出し、引数に設定された値を参照して可視化処理を行う。RVS_USER_INIT は RVS_INIT から、これ以外のユーザ関数は RVS_MAIN から呼び出される。

以下では、RVS_USER_INIT は C 言語で、これ以外の関数は FORTRAN で記述されることを前提として、各関数の仕様を説明する。

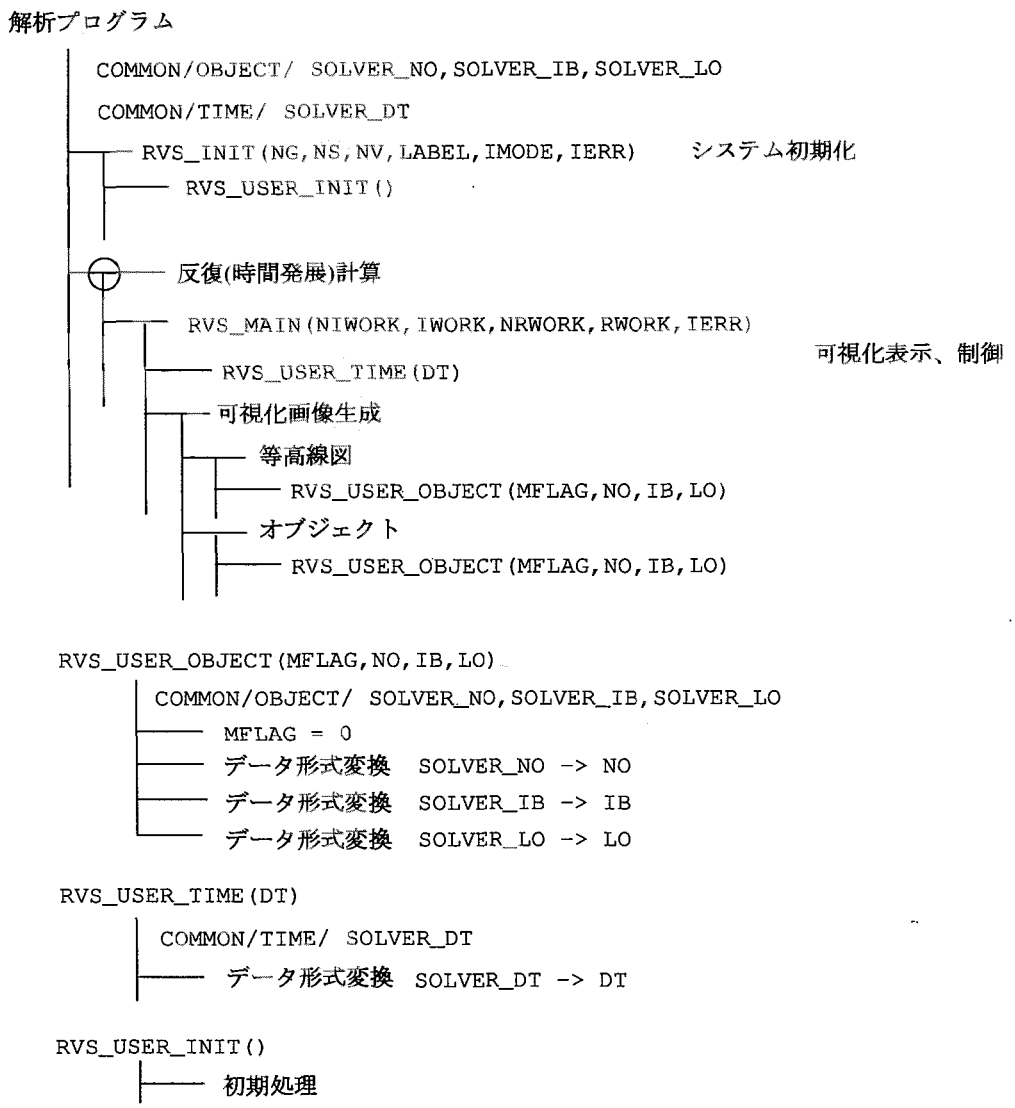


図 2-2: ユーザ関数の役割

2.2.2.1 RVS_USER_OBJECT

(1) 機能

BFC 格子内のオブジェクト (障害物) を表示するためのオブジェクトデータを、本システムに供給する。

(2) 関数インターフェース

```
RVS_USER_OBJECT(MFLAG,NO,IB,LO)
```

(3) 引数

引数名	型	サイズ	入出力	内容
MFLAG	I4	1	出力	データ変更フラグ (0:変更なし,1:変更あり)
NO	I4	1	出力	オブジェクト数
IB	I4	NO	出力	オブジェクトが属するブロックの番号
LO	I4	(6,NO)	出力	オブジェクトを表す格子点の格子インデックス

(4) 利用方法

(a) 解析ソルバ側でオブジェクトに関する変数を COMMON 化する。

(b) RVS_USER_OBJECT 内でも上記 COMMON 変数を宣言する。

(c) RVS_USER_OBJECT 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- MFLAG

  オブジェクト (障害物) データ (NO、LO) の内容が前表示ステップ (解析プログラムから組み込み用サブルーチン RVS_MAIN が呼び出された時点) での内容と異なる場合、MFLAG に 1 を指定する。データの内容に変更が無い場合、MFLAG に 0 を指定する。

- LO

  一つのオブジェクト (障害物) は、計算格子内でボックスとして表現する。

  $LO(1,*)$: I 方向の格子インデックスの最小値
  $LO(2,*)$: I 方向の格子インデックスの最大値
  $LO(3,*)$: J 方向の格子インデックスの最小値
  $LO(4,*)$: J 方向の格子インデックスの最大値
  $LO(5,*)$: K 方向の格子インデックスの最小値
  $LO(6,*)$: K 方向の格子インデックスの最大値

  例えば、$LO(1,*) = 1$、$LO(2,*) = 10$、$LO(3,*) = 2$、$LO(4,*) = 20$、$LO(5,*) = 3$、$LO(6,*) = 30$ とした場合、格子インデックスの (1,2,3)～(10,20,30) の範囲がオブジェクトとして認識される。

- IB

  IB(n) には、n 番目のオブジェクトを含む格子ブロック固有のブロック番号を指定する。格子ブロック固有のブロック番号については、組み込み用サブルーチンの説明を参照。本システムではマルチブロック格子をサポートしていないため、これらに必ず１を指定する。

- 複数のオブジェクトが重なっていても問題はない。
- オブジェクトを表示しない場合、本ユーザ関数をコーディングする必要はない。

2.2.2.2 RVS_USER_TRACER

(1) 機能

O 型および C 型の BFC 格子におけるトポロジー上の切断面の位置データを、本システムに供給する。

(2) 関数インターフェース

```
RVS_USER_TRACER(AREA1,AREA2)
```

(3) 引数

引数名	型	サイズ	入出力	内容
AREA1	I4	6	出力	切断面の位置 (IS1,JS1,KS1,IE1,JE1,KE1)
AREA2	I4	6	出力	切断面の位置 (IS2,JS2,KS2,IE2,JE2,KE2)

(4) 利用方法

(a) 解析ソルバ側で切断面の位置に関する変数を COMMON 化する。

(b) RVS_USER_TRACER 内でも上記 COMMON 変数を宣言する。

(c) RVS_USER_TRACER 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- O 型および C 型の BFC 格子においては、BFC 座標系で異なる 2 つの面に対応する物理空間内の面が 1 つ存在する。これをトポロジー上の切断面という。この切断面に対応する BFC 座標系内の 2 つの面の各々を、これを囲む 2 つの格子点の格子インデックスを用いて指定する。
- 配列 AREA1 では、一方の面を、IS1 ≤ IE1、JS1 ≤ JE1、KS1 ≤ KE1 を満たすように指定する。
- 配列 AREA2 では、他方の面を、(IS1,JS1,KS1) と (IS2,JS2,KS2) が物理空間で一致するように指定する。同様に、(IE1,JE1,KE1) と (IE2,JE2,KE2) も物理空間で一致するように指定する。
- 本ユーザ関数は、O 型および C 型の BFC 格子内でトレーサまたは流線を表示する場合にコーディングする必要がある。ただし切断面が存在しない場合は、コーディングの必要がない。なお、本システムでは切断面が存在する格子ブロックでのトレーサまたは流線表示はサポートしない。

2.2.2.3 RVS_USER_TIME

(1) 機能

時間発展計算における時間刻み幅を、本システムに供給する。

(2) 関数インターフェース

```
RVS_USER_TIME(DT)
```

(3) 引数

引数名	型	サイズ	入出力	内容
DT	R8	1	出力	時間刻み幅 Δt

(4) 利用方法

(a) 解析ソルバ側で時間刻み幅に関する変数を COMMON 化する

(b) RVS_USER_TIME 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_TIME 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

本ユーザ関数は、本システムのメインウィンドウのテキスト表示領域に時間刻み幅を表示させたい場合や、トレーサや流線の表示を行う場合にコーディングする。

2.2.2.4 RVS_USER_INIT

(1) 機能

本システム起動時に行いたい処理内容を記述する。

(2) 関数インターフェース

```
RVS_USER_INIT()
```

(3) 引数

なし。

(4) 利用方法

本ユーザ関数は、本システム初期化時(解析プログラムから組み込み用サブルーチンRVS_INITが呼び出されたとき)に内部的に呼び出される。

(5) 補足説明

なし。

2.2.2.5 RVS_USER_DECOMPOSE_BFC

(1) 機能

並列化における BFC 格子の領域分割情報を供給する。

(2) 関数インターフェース

```
RVS_USER_DECOMPOSE_BFC(IB,NPI,NPJ,NPK,NCOMM,NIS,NJS,NKS)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
NPI	I4	1	出力	I 軸方向の PE 数
NPJ	I4	1	出力	J 軸方向の PE 数
NPK	I4	1	出力	K 軸方向の PE 数
NCOMM	I4	1	出力	PE 群のコミュニケータ
NIS	I4	1	出力	I 軸方向の最初の格子点の格子インデックス (グローバル)
NJS	I4	1	出力	J 軸方向の最初の格子点の格子インデックス (グローバル)
NKS	I4	1	出力	K 軸方向の最初の格子点の格子インデックス (グローバル)

(4) 利用方法

(a) 解析ソルバ側で並列化における BFC 格子の領域分割情報に関する変数を COMMON 化する

(b) RVS_USER_DECOMPOSE_BFC 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_DECOMPOSE_BFC 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB

  ここで対象となる BFC 格子ブロック固有のブロック番号を指定する。格子ブロック固有のブロック番号については、組み込み用サブルーチンの説明を参照。本システムではマルチブロック格子をサポートしていないため、必ず 1 を指定する。

- NPI、NPJ、NPK

  I、J、K 各方向の PE 数を指定する。

- NCOMM

  ここで対象となる BFC 格子ブロックを処理する PE 群のコミュニケータの値を指定する。

- NIS、NJS、NKS

  本関数を呼び出したPEが担当する部分領域に関して、I、J、K各方向の最初の格子点のグローバルな格子インデックスを指定する。

- 本関数は、BFC格子を用いた解析結果を並列処理により可視化する場合にコーディングする必要がある。

2.2.2.6 RVS_USER_DECOMPOSE_SIZE_FEM

(1) 機能

並列化における非構造格子の領域分割情報格納に必要な配列のサイズを供給する。

(2) 関数インターフェース

```
RVS_USER_DECOMPOSE_SIZE_FEM(IB,MAXBEL,MAXPE)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
MAXBEL	I4	1	出力	整合寸法 (IBEL、NPE 用)
MAXPE	I4	1	出力	整合寸法 (IPE 用)

(4) 利用方法

(a) 解析ソルバ側で並列化における非構造格子の領域分割情報格納に必要な配列サイズに関する変数を COMMON 化する。

(b) RVS_USER_DECOMPOSE_SIZE_FEM 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_DECOMPOSE_SIZE_FEM 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB
  ここで対象となる非構造格子ブロック固有のブロック番号を指定する。格子ブロック固有のブロック番号については、組み込み用サブルーチンの説明を参照。本システムではマルチブロック格子をサポートしていないため、必ず 1 を指定する。
- MAXBEL
  RVS_USER_DECOMPOSE_FEM で指定する境界要素番号 (ローカル) および共有 PE 数を格納するのに必要な配列のサイズを指定する。MAXBEL は NBEL 以上である必要がある。NBEL は RVS_USER_DECOMPOSE_FEM で指定する。
- MAXPE
  RVS_USER_DECOMPOSE_FEM で指定する共有 PE 番号を格納するのに必要な配列のサイズを指定する。本関数を呼び出した PE が担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する各要素に対する隣接部分領域数 (PE 数) の総和以上である必要がある。
- 本関数は、非構造格子を用いた解析結果を並列処理により可視化する場合にコーディングする必要がある。

2.2.2.7 RVS_USER_DECOMPOSE_FEM

(1) 機能

並列化における非構造格子の領域分割情報を供給する。

(2) 関数インターフェース

```
RVS_USER_DECOMPOSE_FEM(IB,NCOMM,NBEL,IBEL,NPE,IPE)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
NCOMM	I4	1	出力	PE 群のコミュニケータ
NBEL	I4	1	出力	境界要素数
IBEL	I4	MAXBEL	出力	境界要素番号 (ローカル)
NPE	I4	MAXBEL	出力	共有 PE 数
IPE	I4	MAXPE	出力	共有 PE 番号

(4) 利用方法

(a) 解析ソルバ側で並列化における非構造格子の領域分割情報に関する変数を COMMON 化する

(b) RVS_USER_DECOMPOSE_FEM 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_DECOMPOSE_FEM 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB

  ここで対象となる非構造格子ブロック固有のブロック番号を指定する。格子ブロック固有のブロック番号については、組み込み用サブルーチンの説明を参照。本システムではマルチブロック格子をサポートしていないため、必ず 1 を指定する。

- NCOMM

  ここで対象となる非構造格子ブロックを処理する PE 群のコミュニケータの値を指定する。

- NBEL

  本関数を呼び出した PE が担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する要素の数を指定する。

- IBEL

  本関数を呼び出した PE が担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する要素の要素番号 (ローカル) を指定する。

- NPE

  本関数を呼び出したPEが担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する各要素について、隣接部分領域数(PE数)を指定する。

- IPE

  本関数を呼び出したPEが担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する各要素について、隣接部分領域を処理するPEの番号を指定する。

- 本関数は、非構造格子を用いた解析結果を並列処理により可視化する場合にコーディングする必要がある。

2.2.3 ユーザ関数(ポストプロセッシング機能)

ユーザ関数は、本システムが必要とするパラメータの一部を、解析ソルバのパラメータ内容に応じて与えるために用意されている。本システム利用の際には、必要に応じてユーザ関数をコーディングする。組み込み用サブルーチンはこれらの関数を内部的に呼び出し、コーディング内容に従ったパラメータ値を獲得する。

ユーザ関数を利用するためには、コーディングしたユーザ関数を含むファイルをコンパイルしておき、組み込みロードモジュールを作成する際に本システムのアーカイブファイルとともにリンクすればよい(第3章 参照)。

図2-3にユーザ関数の役割を模式的に示す。一般にユーザ関数は、コモン変数を通して解析ソルバのパラメータ値を獲得し、これに応じて適切な値を関数の引数に設定する。組み込み用サブルーチンは、ユーザ関数を内部的に呼び出し、引数に設定された値を参照して可視化処理を行う。RVS_USER_INITはRVS_INITから、これ以外のユーザ関数はRVS_MAINから呼び出される。

以下では、RVS_USER_INITはC言語で、これ以外の関数はFORTRANで記述されることを前提として、各関数の仕様を説明する。

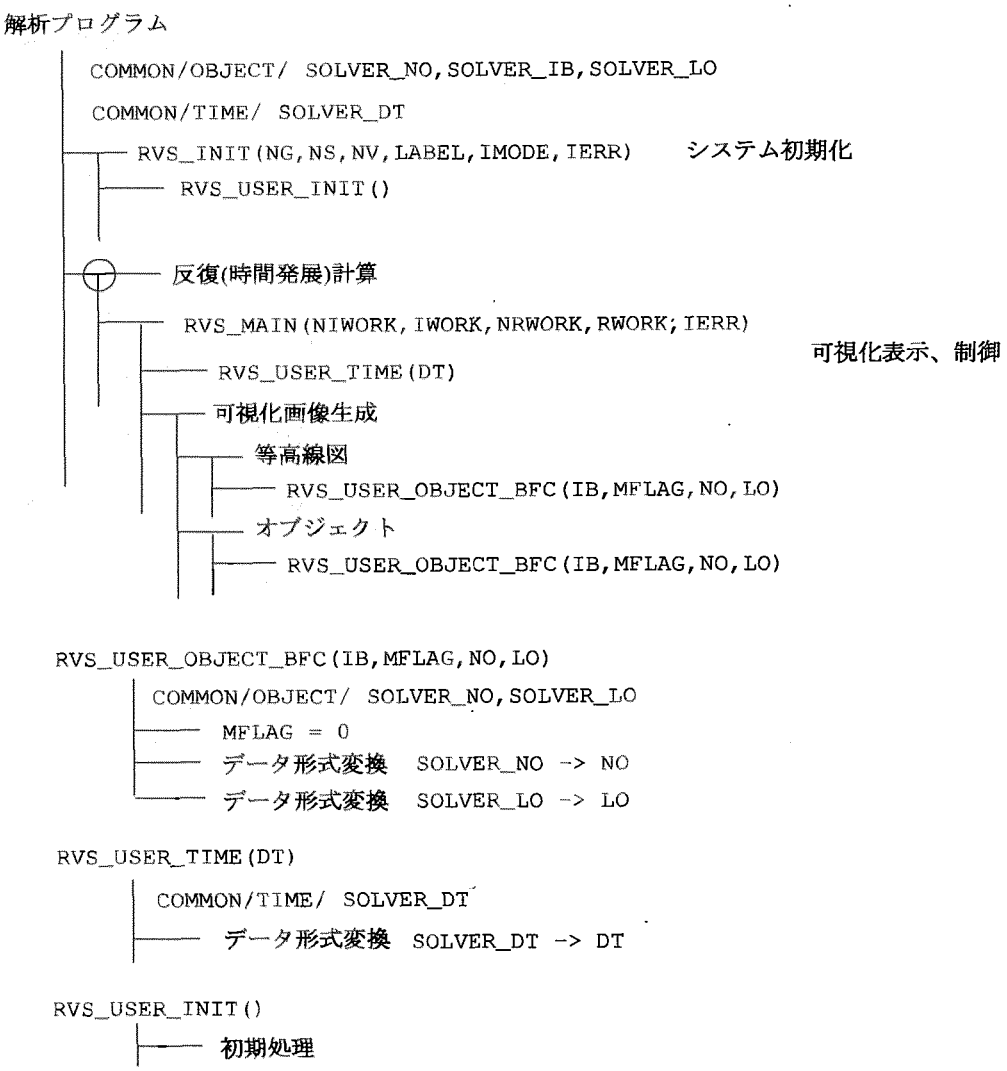


図 2-3: ユーザ関数の役割

2.2.3.1 RVS_USER_OBJECT_BFC

(1) 機能

BFC 格子内のオブジェクト (障害物) を表示するためのオブジェクトデータを、本システムに供給する。

(2) 関数インターフェース

```
RVS_USER_OBJECT_BFC(IB,MFLAG,NO,LO)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
MFLAG	I4	1	出力	データ変更フラグ (0:変更なし,1:変更あり)
NO	I4	1	出力	オブジェクト数
LO	I4	(6,NO)	出力	オブジェクトを表す格子点の格子インデックス

(4) 利用方法

(a) 解析ソルバ側でオブジェクトに関する変数を COMMON 化する。

(b) RVS_USER_OBJECT_BFC 内でも上記 COMMON 変数を宣言する。

(c) RVS_USER_OBJECT_BFC 内に、入力変数 (IB) と COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB

  IB には、可視化処理の対象となっている格子ブロックの番号が本システムから渡される。以下では、渡された番号をもつ格子ブロック内のオブジェクトについてのデータを指定する。本システムではマルチブロック格子をサポートしていないため、IB には必ず 1 が渡される。

- MFLAG

  オブジェクト (障害物) データ (NO、LO) の内容が前表示ステップ (解析プログラムから組み込み用サブルーチン RVS_MAIN が呼び出された時点) での内容と異なる場合、MFLAG に 1 を指定する。データの内容に変更が無い場合、MFLAG に 0 を指定する。

- NO

  格子ブロック内に存在するオブジェクトの数を指定する。

- LO

  一つのオブジェクト (障害物) は、計算格子内でボックスとして表現する。ここでは、解析領域全体におけるグローバルな格子インデックスで指定する。

LO(1,*) ：I 方向の格子インデックスの最小値
LO(2,*) ：I 方向の格子インデックスの最大値
LO(3,*) ：J 方向の格子インデックスの最小値
LO(4,*) ：J 方向の格子インデックスの最大値
LO(5,*) ：K 方向の格子インデックスの最小値
LO(6,*) ：K 方向の格子インデックスの最大値

例えば、LO(1,*) = 1、LO(2,*) = 10、LO(3,*) = 2、LO(4,*) = 20、LO(5,*) = 3、LO(6,*) = 30 とした場合、格子インデックスの (1,2,3)～(10,20,30) の範囲がオブジェクトとして認識される。

- 複数のオブジェクトが重なっていても問題はない。
- オブジェクトを表示しない場合、本ユーザ関数をコーディングする必要はない。

2.2.3.2 RVS_USER_OBJECT_POLYGON_SIZE

(1) 機能

ポリゴンで定義されたオブジェクト (障害物) 情報の格納に必要な配列のサイズを供給する。

(2) 関数インターフェース

```
RVS_USER_OBJECT_POLYGON_SIZE(MAXTRIA)
```

(3) 引数

引数名	型	サイズ	入出力	内容
MAXTRIA	I4	1	出力	整合寸法

(4) 利用方法

(a) 解析ソルバ側でポリゴンで定義されたオブジェクト情報格納に必要な配列サイズに関する変数を COMMON 化する。

(b) RVS_USER_OBJECT_POLYGON_SIZE 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_OBJECT_POLYGON_SIZE 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- MAXTRIA
  RVS_USER_OBJECT_POLYGON で指定するオブジェクト情報を格納するのに必要な配列のサイズを指定する。MAXTRIS は NTRIA 以上である必要がある。NTRIA は RVS_USER_OBJECT_POLYGON で指定する。
- オブジェクトを表示しない場合、またはポリゴンで定義されたオブジェクトが存在しない場合は、本ユーザ関数をコーディングする必要はない。

2.2.3.3 RVS_USER_OBJECT_POLYGON

(1) 機能

ポリゴンで定義されたオブジェクト (障害物) 情報を供給する。

(2) 関数インターフェース

```
RVS_USER_OBJECT_POLYGON(NTRIA,MAT_TRIA,CORD_TRIA,VEC_TRIA)
```

(3) 引数

引数名	型	サイズ	入出力	内容
NTRIA	I4	1	出力	三角形ポリゴンの数
MAT_TRIA	I4	NTRIA	出力	三角形ポリゴンの材質番号
CORD_TRIA	R8	(3,3,MAXTRIA)	出力	三角形ポリゴンの頂点の物理座標値
VEC_TRIA	R8	(3,3,MAXTRIA)	出力	三角形ポリゴンの頂点における単位法線ベクトル

(4) 利用方法

(a) 解析ソルバ側でポリゴンで定義されたオブジェクト情報に関する変数を COMMON 化する

(b) RVS_USER_OBJECT_POLYGON 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_OBJECT_POLYGON 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- NTRIA
  三角形ポリゴンの数として、1 以上の値を指定する。
- MAT_TRIA
  オブジェクトの色を用いた表示を行う場合、同じ材質番号をもつ三角形ポリゴンは同じ色で表示される。つまり、材質番号 n をもつ三角形ポリゴンは n 番目のオブジェクト色で表示される。MAT_TRIA(i) には、i 番目の三角形ポリゴンの材質番号を、1 から 256 の範囲で指定する。
- CORD_TRIA
  各三角形ポリゴンを構成する 3 つの頂点の物理座標値を指定する。

  CORD_TRIA(1, 1, *) : 1 番目の頂点の X 座標
  CORD_TRIA(2, 1, *) : 1 番目の頂点の Y 座標
  CORD_TRIA(3, 1, *) : 1 番目の頂点の Z 座標
  CORD_TRIA(1, 2, *) : 2 番目の頂点の X 座標

CORD_TRIA(2, 2, *) ：2番目の頂点のY座標
CORD_TRIA(3, 2, *) ：2番目の頂点のZ座標
CORD_TRIA(1, 3, *) ：3番目の頂点のX座標
CORD_TRIA(2, 3, *) ：3番目の頂点のY座標
CORD_TRIA(3, 3, *) ：3番目の頂点のZ座標

- VEC_TRIA

  各三角形ポリゴンを構成する3つの頂点における単位法線ベクトルを指定する。ここで指定された法線ベクトルが、オブジェクト表示の際の輝度計算に利用される。

  VEC_TRIA(1, 1, *) ：1番目の頂点における単位法線ベクトルのX成分
  VEC_TRIA(2, 1, *) ：1番目の頂点における単位法線ベクトルのY成分
  VEC_TRIA(3, 1, *) ：1番目の頂点における単位法線ベクトルのZ成分
  VEC_TRIA(1, 2, *) ：2番目の頂点における単位法線ベクトルのX成分
  VEC_TRIA(2, 2, *) ：2番目の頂点における単位法線ベクトルのY成分
  VEC_TRIA(3, 2, *) ：2番目の頂点における単位法線ベクトルのZ成分
  VEC_TRIA(1, 3, *) ：3番目の頂点における単位法線ベクトルのX成分
  VEC_TRIA(2, 3, *) ：3番目の頂点における単位法線ベクトルのY成分
  VEC_TRIA(3, 3, *) ：3番目の頂点における単位法線ベクトルのZ成分

  各法線ベクトルが単位ベクトルに正規化されていない場合、正確な輝度計算が行われない。

- 複数のオブジェクトが重なっていても問題はない。
- オブジェクトを表示しない場合、またはポリゴンで定義されたオブジェクトが存在しない場合は、本ユーティリティ関数を呼び出す必要はない。

2.2.3.4 RVS_USER_CUT_BFC

(1) 機能

O 型および C 型の BFC 格子におけるトポロジー上の切断面の位置データを、本システムに供給する。

(2) 関数インターフェース

```
RVS_USER_CUT_BFC(IB,NAREA,AREA1,AREA2)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
NAREA	I4	1	出力	切断面の数
AREA1	I4	(6,NAREA)	出力	切断面の位置 (IS1,JS1,KS1,IE1,JE1,KE1)
AREA2	I4	(6,NAREA)	出力	切断面の位置 (IS2,JS2,KS2,IE2,JE2,KE2)

(4) 利用方法

(a) 解析ソルバ側で切断面の位置に関する変数を COMMON 化する。

(b) RVS_USER_CUT_BFC 内でも上記 COMMON 変数を宣言する。

(c) RVS_USER_CUT_BFC 内に、入力変数 (IB) と COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- O 型および C 型の BFC 格子においては、BFC 座標系で異なる 2 つの面に対応する物理空間内の面が 1 つ存在する。これをトポロジー上の切断面という。この切断面に対応する BFC 座標系内の 2 つの面の各々を、これを囲む 2 つの格子点の格子インデックスを用いて指定する。
- IB

  IB には、可視化処理の対象となっている格子ブロックの番号が本システムから渡される。以下では、渡された番号をもつ格子ブロック内の切断面についてのデータを指定する。本システムではマルチブロック格子をサポートしていないため、IB には必ず 1 が渡される。
- NAREA

  格子ブロック内に存在する切断面の数を指定する。
- AREA1

  一方の面を、解析領域全体におけるグローバルな格子インデックスを用いて、IS1 $\leq$ IE1、JS1 $\leq$ JE1、KS1 $\leq$ KE1 を満たすように指定する。

- AREA2
  他方の面を、解析領域全体におけるグローバルな格子インデックスを用いて、(IS1, JS1, KS1) と (IS2, JS2, KS2) が物理空間で一致するように指定する。同様に、(IE1, JE1, KE1) と (IE2, JE2, KE2) も物理空間で一致するように指定する。
- 本ユーザ関数は、O 型および C 型の BFC 格子内でトレーサまたは流線を表示する場合にコーディングする必要がある。ただし切断面が存在しない場合は、コーディングの必要がない。なお、本システムでは切断面が存在する格子ブロックでのトレーサまたは流線表示はサポートしない。

2.2.3.5 RVS_USER_TIME

(1) 機能

時間発展計算における時間刻み幅を、本システムに供給する。

(2) 関数インターフェース

```
RVS_USER_TIME(DT)
```

(3) 引数

引数名	型	サイズ	入出力	内容
DT	R8	1	出力	時間刻み幅 Δt

(4) 利用方法

(a) 解析ソルバ側で時間刻み幅に関する変数を COMMON 化する

(b) RVS_USER_TIME 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_TIME 内に、COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

本ユーザ関数は、本システムのメインウィンドウのテキスト表示領域に時間刻み幅を表示させたい場合や、トレーサや流線の表示を行う場合にコーディングする。

2.2.3.6 RVS_USER_INIT

(1) 機能

本システム起動時に行いたい処理内容を記述する。

(2) 関数インターフェース

```
RVS_USER_INIT()
```

(3) 引数

なし。

(4) 利用方法

本ユーザ関数は、本システム初期化時 (解析プログラムから組み込み用サブルーチン RVS_INIT が呼び出されたとき) に内部的に呼び出される。

(5) 補足説明

なし。

2.2.3.7 RVS_USER_COMM

(1) 機能

並列化におけるコミュニケータ情報を供給する。

(2) 関数インターフェース

```
RVS_USER_COMM(IB,NCOMM)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
NCOMM	I4	1	出力	PE群のコミュニケータ

(4) 利用方法

(a) 解析ソルバ側で並列化におけるコミュニケータ情報に関する変数をCOMMON化する。

(b) RVS_USER_COMM内でも上記COMMON変数を宣言する

(c) RVS_USER_COMM内に、入力変数(IB)とCOMMON変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB

  IBには、可視化処理の対象となっている格子ブロックの番号が本システムから渡される。以下では、渡された番号をもつ格子ブロックを可視化処理するPE群のコミュニケータ情報を指定する。本システムではマルチブロック格子をサポートしていないため、IBには必ず1が渡される。

- NCOMM

  格子ブロックを可視化処理するPE群のコミュニケータを指定する。

- 本関数は、解析結果を並列処理により可視化する場合にコーディングする必要がある。

2.2.3.8 RVS_USER_DECOMPOSE_BFC

(1) 機能

並列化における BFC 格子の領域分割情報を供給する。

(2) 関数インターフェース

```
RVS_USER_DECOMPOSE_BFC(IB,NPI,NPJ,NPK,NIS,NJS,NKS)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
NPI	I4	1	出力	I 軸方向の PE 数
NPJ	I4	1	出力	J 軸方向の PE 数
NPK	I4	1	出力	K 軸方向の PE 数
NIS	I4	1	出力	I 軸方向の最初の格子点の格子インデックス (グローバル)
NJS	I4	1	出力	J 軸方向の最初の格子点の格子インデックス (グローバル)
NKS	I4	1	出力	K 軸方向の最初の格子点の格子インデックス (グローバル)

(4) 利用方法

(a) 解析ソルバ側で並列化における BFC 格子の領域分割情報に関する変数を COMMON 化する

(b) RVS_USER_DECOMPOSE_BFC 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_DECOMPOSE_BFC 内に、入力変数 (IB) と COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB

  IB には、可視化処理の対象となっている格子ブロックの番号が本システムから渡される。以下では、渡された番号をもつ格子ブロックについての領域分割情報を指定する。本システムではマルチブロック格子をサポートしていないため、IB には必ず 1 が渡される。

- NPI、NPJ、NPK

  I、J、K 各方向の PE 数を指定する。

- NIS、NJS、NKS

  本関数を呼び出した PE が担当する部分領域に関して、I、J、K 各方向の最初の格子点を解析領域全体におけるグローバルな格子インデックスを指定する。

- 本関数は、BFC 格子を用いた解析結果を並列処理により可視化する場合にコーディングする必要がある。

2.2.3.9 RVS_USER_DECOMPOSE_SIZE_FEM

(1) 機能

並列化における非構造格子の領域分割情報格納に必要な配列のサイズを供給する。

(2) 関数インターフェース

```
RVS_USER_DECOMPOSE_SIZE_FEM(IB,MAXBEL,MAXPE)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
MAXBEL	I4	1	出力	整合寸法 (IBEL、NPE 用)
MAXPE	I4	1	出力	整合寸法 (IPE 用)

(4) 利用方法

(a) 解析ソルバ側で並列化における非構造格子の領域分割情報格納に必要な配列サイズに関する変数を COMMON 化する。

(b) RVS_USER_DECOMPOSE_SIZE_FEM 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_DECOMPOSE_SIZE_FEM 内に、入力変数 (IB) と COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB

  IB には、可視化処理の対象となっている格子ブロックの番号が本システムから渡される。以下では、渡された番号をもつ格子ブロックについての領域分割情報格納に必要な配列サイズを指定する。本システムではマルチブロック格子をサポートしていないため、IB には必ず 1 が渡される。

- MAXBEL

  RVS_USER_DECOMPOSE_FEM で指定する境界要素番号 (ローカル) および共有 PE 数を格納するのに必要な配列のサイズを指定する。MAXBEL は NBEL 以上である必要がある。NBEL は RVS_USER_DECOMPOSE_FEM で指定する。

- MAXPE

  RVS_USER_DECOMPOSE_FEM で指定する共有 PE 番号を格納するのに必要な配列のサイズを指定する。本関数を呼び出した PE が担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する各要素に対する隣接部分領域数 (PE 数) の総和以上である必要がある。

- 本関数は、非構造格子を用いた解析結果を並列処理により可視化する場合にコーディングする必要がある。

2.2.3.10 RVS_USER_DECOMPOSE_FEM

(1) 機能

並列化における非構造格子の領域分割情報を供給する。

(2) 関数インターフェース

```
RVS_USER_DECOMPOSE_FEM(IB,NBEL,IBEL,NPE,IPE)
```

(3) 引数

引数名	型	サイズ	入出力	内容
IB	I4	1	入力	ブロック番号
NBEL	I4	1	出力	境界要素数
IBEL	I4	MAXBEL	出力	境界要素番号 (ローカル)
NPE	I4	MAXBEL	出力	共有 PE 数
IPE	I4	MAXPE	出力	共有 PE 番号

(4) 利用方法

(a) 解析ソルバ側で並列化における非構造格子の領域分割情報に関する変数を COMMON 化する

(b) RVS_USER_DECOMPOSE_FEM 内でも上記 COMMON 変数を宣言する

(c) RVS_USER_DECOMPOSE_FEM 内に、入力変数 (IB) と COMMON 変数の内容に応じて適切な値を引数に設定する処理を記述する。

(5) 補足説明

- IB

  IB には、可視化処理の対象となっている格子ブロックの番号が本システムから渡される。以下では、渡された番号をもつ格子ブロックについての領域分割情報を指定する。本システムではマルチブロック格子をサポートしていないため、IB には必ず 1 が渡される。

- NBEL

  本関数を呼び出した PE が担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する要素の数を指定する。

- IBEL

  本関数を呼び出した PE が担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する要素の要素番号 (ローカル) を指定する。

- NPE

  本関数を呼び出した PE が担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する各要素について、隣接部分領域数 (PE 数) を指定する。

- IPE

  本関数を呼び出したPEが担当する部分領域に関して、部分領域境界に位置し、かつ他の部分領域に接する各要素について、隣接部分領域を処理するPEの番号を指定する。

- 本関数は、非構造格子を用いた解析結果を並列処理により可視化する場合にコーディングする必要がある。

2.2.4 ユーティリティ関数 (サーバ機能)

ユーティリティ関数は、本システムが必要とするパラメータの一部を、解析ソルバのパラメータ内容に応じて与えるために用意されている。本システム利用の際には、解析ソルバの中に必要に応じてユーティリティ関数呼び出し部分を設け、パラメータの値を関数の引数として指定する。

2.2.4.1 RVS_UTIL_OBJECT_POLYGON

(1) 機能

非構造四面体格子または非構造六面体格子内のオブジェクト (障害物) を表示するためのオブジェクトデータを、本システムに供給する。

(2) 呼び出し形式

```
CALL RVS_UTIL_OBJECT_POLYGON(NTRIA,MAT_TRIA,CORD_TRIA,VEC_TRIA,IERR)
```

(3) 引数

引数名	型	サイズ	入出力	内容
NTRIA	I4	1	入力	三角形ポリゴンの数
MAT_TRIA	I4	NTRIA	入力	三角形ポリゴンの材質番号
CORD_TRIA	R8	(3,3,NTRIA)	入力	三角形ポリゴンの頂点の物理座標値
VEC_TRIA	R8	(3,3,NTRIA)	入力	三角形ポリゴンの頂点における単位法線ベクトル
IERR	I4	1	出力	エラーコード

(4) 利用方法

(a) 解析ソルバ側でオブジェクトに関する変数を用意する。

(b) 各時刻ステップでオブジェクトデータが変化しない場合は、組み込み用サブルーチン RVS_INIT を呼び出した後、反復 (時間発展) 計算に入る前に本ユーティリティ関数を呼び出す。各時刻ステップでオブジェクトデータが変化する場合は、本ユーティリティ関数を RVS_MAIN の直前で呼び出す。

(5) 補足説明

- NTRIA

  三角形ポリゴンの数として、1 以上の値を指定する。

- MAT_TRIA

  オブジェクトの色を用いた表示を行う場合、同じ材質番号をもつ三角形ポリゴンは同じ色で表示される。つまり、材質番号 n をもつ三角形ポリゴンは n 番目のオブジェクト色で表示される。MAT_TRIA(i) には、i 番目の三角形ポリゴンの材質番号を、1 から 256 の範囲で指定する。

- CORD_TRIA

  各三角形ポリゴンを構成する 3 つの頂点の物理座標値を指定する。

  CORD_TRIA(1,1,*) : 1 番目の頂点の X 座標
  CORD_TRIA(2,1,*) : 1 番目の頂点の Y 座標

CORD_TRIA(3,1,*) : 1番目の頂点のZ座標
CORD_TRIA(1,2,*) : 2番目の頂点のX座標
CORD_TRIA(2,2,*) : 2番目の頂点のY座標
CORD_TRIA(3,2,*) : 2番目の頂点のZ座標
CORD_TRIA(1,3,*) : 3番目の頂点のX座標
CORD_TRIA(2,3,*) : 3番目の頂点のY座標
CORD_TRIA(3,3,*) : 3番目の頂点のZ座標

- VEC_TRIA

  各三角形ポリゴンを構成する3つの頂点における単位法線ベクトルを指定する。ここで指定された法線ベクトルが、オブジェクト表示の際の輝度計算に利用される。

  VEC_TRIA(1,1,*) : 1番目の頂点における単位法線ベクトルのX成分
  VEC_TRIA(2,1,*) : 1番目の頂点における単位法線ベクトルのY成分
  VEC_TRIA(3,1,*) : 1番目の頂点における単位法線ベクトルのZ成分
  VEC_TRIA(1,2,*) : 2番目の頂点における単位法線ベクトルのX成分
  VEC_TRIA(2,2,*) : 2番目の頂点における単位法線ベクトルのY成分
  VEC_TRIA(3,2,*) : 2番目の頂点における単位法線ベクトルのZ成分
  VEC_TRIA(1,3,*) : 3番目の頂点における単位法線ベクトルのX成分
  VEC_TRIA(2,3,*) : 3番目の頂点における単位法線ベクトルのY成分
  VEC_TRIA(3,3,*) : 3番目の頂点における単位法線ベクトルのZ成分

  各法線ベクトルが単位ベクトルに正規化されていない場合、正確な輝度計算が行われない。

- IERR

  本関数は、特にエラーコードを出力しない。

- 複数のオブジェクトが重なっていても問題はない。
- オブジェクトを表示しない場合、またはポリゴンで定義されたオブジェクトが存在しない場合は、本ユーティリティ関数を呼び出す必要はない。

2.2.5 ユーティリティ関数(ポストプロセッシング機能)

ユーティリティ関数は、本システムが必要とするパラメータの一部を、解析ソルバから与えるために用意されている。本システム利用の際には、解析ソルバの中に必要に応じてユーティリティ関数呼び出し部分を設け、パラメータの値を関数の引き数として指定する。ユーザ関数はユーザがコーディングする必要があるが、ユーティリティ関数はコーディングを行う必要は無い。またユーザ関数が内部的に呼び出されるのに対し、ユーティリティー関数はユーザが呼び出しを行う。

2.2.5.1 RVS_UTIL_TOPOGRAPHY

(1) 機能

地形図データ ETOPO5 から、地形図データを読み込む。

(2) 関数インターフェース

```
RVS_UTIL_TOPOGRAPHY(LONG1,LAT1,LONG2,LAT2,LONG_INT,LAT_INT)
```

(3) 引数

引数名	型	サイズ	入出力	内容
LONG1	R8	1	入力	左上の点の経度 (東経 +、西経 −)
LAT1	R8	1	入力	左上の点の緯度 (北緯 +、南緯 −)
LONG2	R8	1	入力	右下の点の経度 (東経 +、西経 −)
LAT2	R8	1	入力	右下の点の緯度 (北緯 +、南緯 −)
LONG_INT	I4	1	入力	経度方向の刻み (分単位、5 分刻み)
LAT_INT	I4	1	入力	緯度方向の刻み (分単位、5 分刻み)

左上の点、右下の点については補足説明と図 2-4 を参考にされたい。

(4) 利用方法

(a) 解析ソルバ側で、rvs_postlib_() を呼び出す前に 1 回呼び出す。

(5) 補足説明

- 緯度・経度の範囲指定
  緯度・経度の範囲は、ソルバの格子に合わせて指定する。ETOPO5 形式のデータの並びに合わせ、図 2-4 の矩形のような範囲で指定する。このとき、関数には左上と右下の点の緯度・経度を記述する。
- LONG1, LONG2
  経度方向のみ、西経 0 度を 360 度とした指定方法も可能である。西経 160 度の場合、−160.0 でも 200.0 のどちらでもよい。
- LONG_INT, LAT_INT
  緯度・経度方向の刻みは分単位で、5 分刻みで指定する。1 度の場合は 60 を指定する。
- 地形図を表示しない場合、本ユーティリティ関数を呼び出す必要は無い。

(6) 制限事項

- ソルバ格子が X 成分と Y 成分が緯度・経度で定義されている BFC 直交格子にのみ対応している。
- 非構造格子ソルバには対応していない。

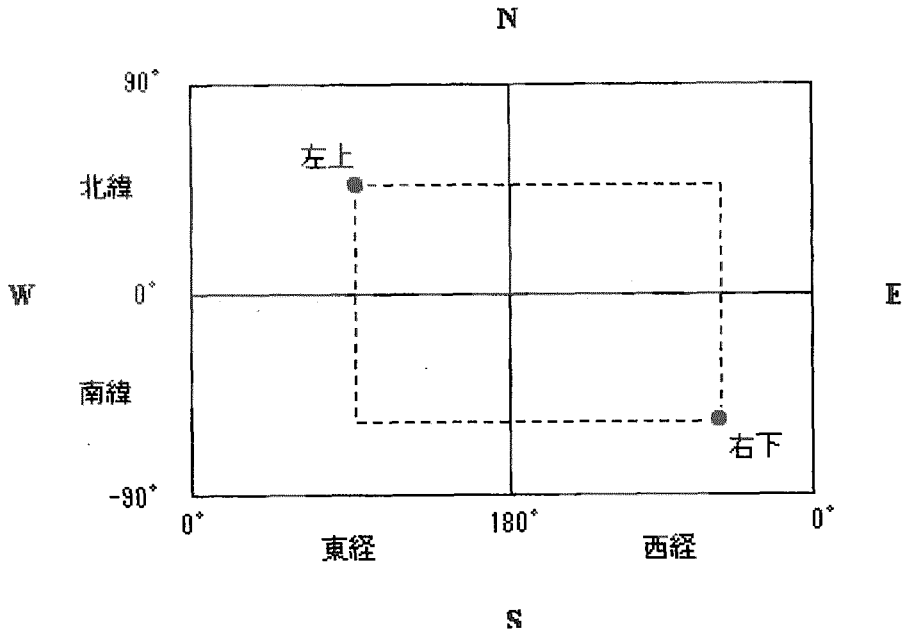


図 2-4: 地形図の指定範囲

- 地形図データ (etopo5.dat) は、プログラム実行を行うカレントディレクトリに置く。
- 地形図用の格子データはソルバの格子データとは別に確保される。そのため、ソルバ格子とは異なった範囲、刻みを指定することも可能である。ただし、緯度・経度の刻みを小さくし、表示範囲を広くすると計算時間およびメモリは増大することに留意されたい。
- 基準子午線（東経西経 0 度）をまたがる範囲および、北極南極をまたがる範囲は指定できない。補足説明の図上でひとつの矩形として表現できる範囲が対象である。

2.2.6 エラーコード

ここでは、組み込み用サブルーチンが出力するエラーコードの内容について説明する。

2.2.6.1 エラーコード仕様

エラーコードは8桁の整数で表現する。以下で、'x' は任意の数字を表す。

(1) エラー種別

コード番号	内容	説明
1xxxxxxx	INFORMATIVE	実行を継続し、処理内容にも問題がない。
2xxxxxxx	WARNING	実行を継続するが、処理内容に問題がある。
3xxxxxxx	FATAL	実行を中止する。

(2) エラーコードとエラー発生機能との関係

番号	エラー発生機能
x0100xxx	RVS_INIT
x0200xxx	RVS_MAIN
x0210xxx	RVS_BFC
x0220xxx	RVS_TET1
x0230xxx	RVS_HEX1
x0300xxx	RVS_TERM
x0911xxx	RVS_POSTLIB(NetCDF)
x0921xxx	RVS_POSTLIB(GrADS)
x0931xxx	RVS_POSTLIB(AVS Field Data)
x1000xxx	グラフィック部
x1100xxx	ボリュームレンダリング
x1200xxx	トレーサ
x1300xxx	等高線
x1400xxx	オブジェクト
x1500xxx	ベクトル
x1600xxx	格子図
x1700xxx	流線
x1800xxx	等値面図
x00xxxxx	その他

(3) エラーコードと内容との関係

番号	説明
1xxxx001	オブジェクト表示機能が選択されているが、オブジェクト数が 0 である。
1xxxx002	表示機能が選択されているが、表示対象物が指定されていない。
1xxxx999	バッチ処理モードで実行中に、ユニバーサルシナリオファイルの再生が終了した。
2xxxx001	LABEL の内容が空白である。
2xxxx002	オブジェクトの数が 256 個より大きい。
2xxxx003	整数作業領域が不足している。
2xxxx004	実数作業領域が不足している。
2xxxx005	NI、NJ、NK のいずれかが 1 以下である。3 次元格子を指定する必要がある。
2xxxx006	NV ＜ 1。 ベクトルデータが必要である。
2xxxx007	指定された座標面が解析領域外である。
2xxxx008	縮退した格子が存在する。
2xxxx009	格子点座標が、各軸方向に昇順または降順ではない。
2xxxx010	オブジェクトデータ変更フラグが 0 または 1 でない。
2xxxx011	指定された格子点が解析領域外である。
2xxxx012	必要な作業領域が内部的に確保できなかった。
3xxxx001	指定されたスカラデータ数またはベクトルデータ数に誤りがある。
3xxxx002	指定された格子点数、節点数、要素数が、0 以下である。
3xxxx003	指定された格子点数、節点数、要素数が、対応する整合寸法を超えている。
3xxxx004	NB が 1 でない。
3xxxx005	IB が 1 でない。
3xxxx006	格子格納領域の動的確保ができない。
3xxxx008	IMODE が 0 または 1 でない。
3xxxx009	データファイルオープンエラー
3xxxx010	データファイルリードエラー
3xxxx011	制御ファイルオープンエラー
3xxxx012	制御ファイルリードエラー
3xxxx013	未サポートのデータ
3xxxx014	制御データの文法エラー
3xxxx015	制御データの記述内容誤り
3xxxx997	クライアントとの通信が確立されていない。

2.2.6.2 エラーコード一覧

(1) INFORMATIVE

番号	エラー発生機能	説明
10200999	RVS_MAIN	バッチ処理モードで実行中に、 ユニバーサルシナリオファイルの再生が終了した。
11400001	オブジェクト	オブジェクト表示機能が選択されているが、 オブジェクト数が 0 である。
11300002	等高線	等高線表示機能が選択されているが、 表示すべき座標面または断面が指定されていない。
11500002	ベクトル	ベクトル表示機能が選択されているが、 表示すべき座標面または断面が指定されていない。
11800002	等値面	等値面表示機能が選択されているが、 表示すべき等値面が指定されていない。

(2) WARNING

番号	エラー発生機能	説明
20100001	RVS_INIT	LABEL の内容が空白である。
21300002	等高線	オブジェクトの数が 256 個より大きい。
21400002	オブジェクト	オブジェクトの数が 256 個より大きい。
21500002	ベクトル	オブジェクトの数が 256 個より大きい。
21000003	グラフィック部	整数作業領域が不足している。
21200003	トレーサ	整数作業領域が不足している。
21300003	等高線	整数作業領域が不足している。
21400003	オブジェクト	整数作業領域が不足している。
21500003	ベクトル	整数作業領域が不足している。
21700003	流線	整数作業領域が不足している
21800003	等値面	整数作業領域が不足している
21000004	グラフィック部	実数作業領域が不足している。
21200004	トレーサ	実数作業領域が不足している。
21300004	等高線	実数作業領域が不足している。
21400004	オブジェクト	実数作業領域が不足している。
21500004	ベクトル	実数作業領域が不足している。
21600004	格子図	実数作業領域が不足している。
21700004	流線	実数作業領域が不足している。
21200006	トレーサ	NV < 1。ベクトルデータが必要である。
21700006	流線	NV < 1。ベクトルデータが必要である。
21300007	等高線	指定された座標面が解析領域外である。
21500007	ベクトル	指定された座標面が解析領域外である。
21300010	等高線	オブジェクトデータ変更フラグが 0 または 1 でない。

番号	エラー発生機能	説明
21400010	オブジェクト	オブジェクトデータ変更フラグが 0 または 1 でない。
21500010	ベクトル	オブジェクトデータ変更フラグが 0 または 1 でない。
21200011	トレーサ	指定された格子点が解析領域外である。
21600011	格子図	指定された格子点が解析領域外である。
21700011	流線	指定された格子点が解析領域外である。
21200012	トレーサ	必要な作業領域が内部的に確保できなかった。
21700012	流線	必要な作業領域が内部的に確保できなかった。

(3) FATAL

番号	エラー発生機能	説明
30100001	RVS_INIT	NS $<$ 0 または NV $<$ 0 または NS+NV $=$ 0。
30210001	RVS_BFC	NS $<$ 0 または NV $<$ 0 または NS+NV $=$ 0。
30220001	RVS_TET1	NS $<$ 0 または NV $<$ 0 または NS+NV $=$ 0。
30230001	RVS_HEX1	NS $<$ 0 または NV $<$ 0 または NS+NV $=$ 0。
30210002	RVS_BFC	NI $\leqq$ 0 または NJ $\leqq$ 0 または NK $\leqq$ 0。
30220002	RVS_TET1	NNOD $\leqq$ 0 または NELM $\leqq$ 0。
30230002	RVS_HEX1	NNOD $\leqq$ 0 または NELM $\leqq$ 0。
30210003	RVS_BFC	MAXI $<$ NI または MAXJ $<$ NJ または MAXK $<$ NK。
30220003	RVS_TET1	MNOD $<$ NNOD または MELM $<$ NELM。
30230003	RVS_HEX1	MNOD $<$ NNOD または MELM $<$ NELM。
30100004	RVS_INIT	NB が 1 でない。
30210005	RVS_BFC	IB が 1 でない。
30220005	RVS_TET1	IB が 1 でない。
30230005	RVS_HEX1	IB が 1 でない。
30230006	RVS_HEX1	格子格納領域の動的確保ができない。
30100008	RVS_INIT	IMODE が 0 または 1 でない。
30200997	RVS_MAIN	クライアントとの通信が確立されていない。

3 利用方法

3.1 ハードウェアの条件

- サーバ機能

  対応機種 (OS)： SX-4 シリーズ (SUPER-UX)
  メモリ容量： 解析規模に依存

- ポストプロセッシング機能

  対応機種 (OS)： SX-4 シリーズ (SUPER-UX)
  メモリ容量： データサイズに依存

3.2 ディレクトリおよびファイル一覧

インストールディレクトリ以下のディレクトリ構成を以下に示す。

- サーバ機能

bin/	ロードモジュール
bin/bin.super/nasrvsui-server	組み込みロードモジュールのサンプル
sample/	サンプルデータ
sample/karman.case	サンプルソルバパラメータファイル
sample/karman.grid	サンプルソルバ格子データファイル
sample/karman.para	パラメータ設定ファイルのサンプル
sample/karman.obj	サンプルソルバオブジェクト (障害物) データファイル
resources/	各種リソースファイル
resources/system.para	パラメータ既定値設定ファイル
src/	ソースおよびオブジェクトファイル
lib/	ライブラリ
lib/libRVS_D.a	アーカイブファイル (倍精度)

- ポストプロセッシング機能

bin/	ロードモジュール
bin/rvs_post	ポストプロセッシング機能の実行モジュール
test/	サンプルデータ
test/avs/	AVS Filed Data のサンプル
test/grads/	GrADS データのサンプル
test/netcdf/	NetCDF データのサンプル
resources/	各種リソースファイル
resources/system.para	パラメータ既定値設定ファイル
src/	ソースおよびオブジェクトファイル
lib/	ライブラリ
lib/libRVS_D.a	アーカイブファイル (倍精度)

3.3 組み込みロードモジュールの生成

本システムのライブラリを組み込んだ解析プログラムまたはポストプロセッシングプログラムを、アーカイブファイルおよびユーザ関数とリンクし、組み込みロードモジュールを生成する。なお、OS のバージョンによりリンクすべきシステムライブラリが異なる場合があるので、注意されたい。

- mpi を利用しない場合
  f77sx -o *lm_file program.*f *user.*f *$SYSTEM*/lib/libRVS_D.a -l -lXt -l -lX11 -l -lSM -l -lICE -multi

- mpi を利用する場合
  mpif90 -o *lm_file program.*f *user.*f *$SYSTEM*/lib/libRVS_D.a -lXt -lX11 -lSM -lICE -Pmulti

ここで、*lm_file*、*program.*f、*user.*f、*$SYSTEM* は、それぞれ組み込みロードモジュール名、解析プログラムファイル名またはポストプロセッシングプログラムファイル名、ユーザ関数ファイル名、インストールディレクトリ名である。

3.4 ポストプロセッシング機能実行モジュールの実行

ポストプロセッシング機能は、実行モジュールとしても提供されている。これによりデータファイルがあれば、ライブラリを利用してポストプロセッシングプログラムを作成することなしに、本機能が利用できる。実行モジュールは以下の形式で起動する。

Module-name *imode Datafile-name Parameterfile-name*

Module-name	実行するポストプロセッシング機能のロードモジュール名 *$SYSTEM*/bin/rvs_post ここで*$SYSTEM* はインストールディレクトリ名である。

imode 実時間可視化システムの実行モード
imode に 0 を指定すると、バッチ処理が行われる。*imode* に 1 を指定するとサーバ・クライアント型の処理が行われる。

Datafile-name 可視化を行うデータファイル名
NetCDF の NetCDF 入力ファイル名、GrADS のデータ記述ファイル名、AVS Field Data ファイル名のいずれか。

Parameterfile-name データファイルを可視化するのに必要なパラメータ設定ファイル名
パラメータ設定ファイルについては 3.5 節参照。

データファイルとパラメータ設定ファイルの指定には以下の制限がある。

(1) 指定できる文字列は 255 文字以下である。

(2) 各ファイルの拡張子は、NetCDF 入力ファイルが ".nc"、GrADS データ記述ファイルが ".ctl"、AVS Field Data ファイルが ".fld"、パラメータ設定ファイルが ".para" でなければならない。

(3) *imode* が 1(分散処理モード) の場合、

- データファイルとパラメータ設定ファイルのファイル名はルート (/) からのフルパスで指定。
- システムのパラメータ既定値設定ファイル "system.para"(*$SYSTEM*/resources の下にファイルがある。ここで*$SYSTEM* は実時間可視化システムのインストールディレクトリ) が、パラメータ設定ファイルと同じディレクトリになければならない。

(4) *imode* が 0(バッチ処理モード) の場合、

- データファイルとパラメータ設定ファイルとシステムのパラメータ既定値設定ファイル "system.para" は同じディレクトリになくてはならない。
- そのディレクトリ名を環境変数 RVSLIB に設定する。
- データファイルとパラメータ設定ファイルの指定は、ファイル名だけ (ディレクトリを含めず) を指定する。

バッチ処理モード時の各ファイルのファイル名の定義が、サーバ機能とポストプロセッシング機能で異なるので注意が必要。サーバ機能におけるバッチ処理モードの利用方法は 3.6 節参照。

3.5 パラメータ設定ファイルの準備

パラメータ設定ファイルは、本システムの可視化パラメータの値を指定するためのものである。解析ケースによって各種パラメータに最適な値が設定できるよう、仕様を付録 B で説明している。本システム初期化時 (解析プログラムから組み込み用サブルーチン RVS_INIT が呼び出された時) には、指定されたパラメータ設定ファイルが読み込まれる。パラメータ既定値設定ファイルとして、*$SYSTEM*/resources/system.para が用意されている。

3.6 バッチ処理による実行

ユニバーサルシナリオファイル (USF) の連続再生による可視化画像のファイル出力を、バッチ処理により実行することができる。この場合、クライアント側に画像は表示されず、サーバ側でのみ処理が行われる。

USF を用いたバッチ処理を行うためには、パラメータ既定値設定ファイル、パラメータ設定ファイル、および USF を準備し、以下の名前で任意のディレクトリ下に置く。

- system.para　　パラメータ既定値設定ファイル
- user.para　　パラメータ設定ファイル
- scenario.usf　　ユニバーサルシナリオファイル (USF)

なお、同ディレクトリに para.index という名前のファイルがあると http 版用モードで動作してしまうため、バッチ処理を行う場合には必ずリネームもしくは削除すること。http 版モードで動作した場合には、出力画像の形式が JPEG に固定されるなどの現象などが起こる。

ポストプロセッシング機能をバッチ処理モードで利用する場合は上記とほぼ同じ設定が必要であるが、ただ一点異なることは、パラメータ設定ファイルのファイル名が任意のファイル名でよいということである。

サーバ機能の場合、バッチ処理において複数の USF を用意することにより、一回の解析計算で複数の動画ファイルを作成することも可能である。これは一つの解析計算を複数のカメラで撮影するような効果をもたらすものであり、マルチカメラ機能と呼ぶ。この機能を利用するためには、scenario.usf という名前の USF に代えて、以下のような名前をもつ最大 8 個までの USF を用意する。

scenario.N.usf (N=1,2,...,8)

用意されたユニバーサルシナリオファイルの数がカメラの台数を決め、N はカメラ番号となる。カメラ番号は 1 から順番に付ける必要がある。ただし scenario.usf という名前のファイルも用意されている場合は、マルチカメラ機能を利用できず scenario.usf に従った処理だけが行われる。次に、これらのファイルを置いたディレクトリのパス名を環境変数 RVSLIB に設定する。さらに、本システムのインストールディレクトリを環境変数 NASRVS_WD に設定する。

可視化画像をファイル出力するためには、パラメータ設定ファイルにサーバ側に画像を保存するための指定が必要である。このための設定の例を以下に示す。

Store_images true Server PPM 'rvs_image' 1 1000

ただし、'rvs_image' の部分にはファイル名となる任意の文字列を指定できる。このような指定が無い場合、画像データは出力されない。

本システムでは、次の動画および静止画形式の画像ファイルを出力することができる。

- RMV
  このファイル形式は、本システム独自の動画保存形式である。各時刻ステップの画像データ間の差分をとったものを、Huffman 可変長符合とランレングスを用いて圧縮している。また、

再生時刻情報を含んでおり、複数の動画ファイルと同期をとりながら再生するマルチビューに対応できる。

- PPM (Portable Pixmap Utilities)
  このファイル形式は、UNIX や PC 上で広く利用されている静止画像の保存形式である。各ピクセルの RGB 値をビット列でそのまま書き出したものであり、データ圧縮はなされていない。ファイル形式が単純であるため、画像データを扱うプログラムの作成も容易である。

- TGA (Targa Image File)
  このファイル形式は、UNIX や PC 上で広く利用されている静止画像の保存形式である。この形式は、画像データを取り扱う大抵の市販ソフトウェアでサポートされているため、作成した画像データの編集を行う場合に便利である。

- JPEG
  このファイル形式は、WWW(World Wide Web) 上で広く利用されている静止画像の保存形式で、Web ブラウザで表示することができる。

マルチカメラ機能を利用しない場合に生成される画像データファイルの名前は、以下のようになる。

RMV ：rvs_image.rmv
PPM ：rvs_image. 通し番号.ppm
TGA ：rvs_image+通し番号 (4 桁固定).tga
JPEG：rvs_image. 通し番号.jpg

マルチカメラ機能を利用した場合に生成される画像データファイルの名前は、以下のようになる。

PPM ：rvs_image. 通し番号. カメラ番号.ppm
RMV：rvs_image. カメラ番号.rmv
TGA ：rvs_image+通し番号 (4 桁固定). カメラ番号.tga

以上のような準備を行った後、実時間可視化システムを組み込んだロードモジュールをバッチ処理により実行する。環境変数 RVSLIB に設定したディレクトリには、実行ログが server-replay.log という名前でファイル出力される。

3.7 分散処理による実行

分散処理モードにおける環境設定についてはサーバ機能用クライアントとポストプロセッシング機能用クライアントで異なり、各クライアントのマニュアルに詳しい説明があるのでそれを参照すること。

謝　辞

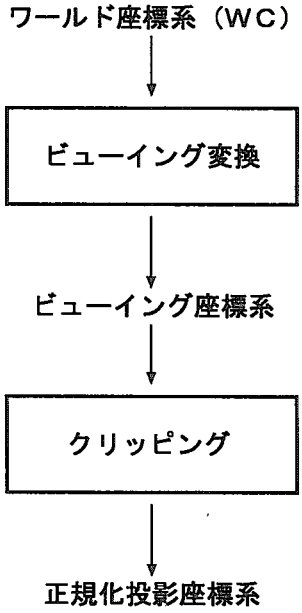
本報告書を取りまとめるにあたり、貴重な機会を与えてくださいました計算科学技術推進センター長矢川元基氏、並列処理支援技術開発グループリーダー平山俊雄氏に深謝致します。

付 録A　座標変換

本システムでは、解析結果の可視化表示を行うまでにいくつかの座標変換を行なう。以下ではこの座標変換についての概要を説明する。

A.1　座標変換の流れ

以下に座標変換の流れを示す。また以降ではそれぞれの座標系、変換処理の概要を説明する。



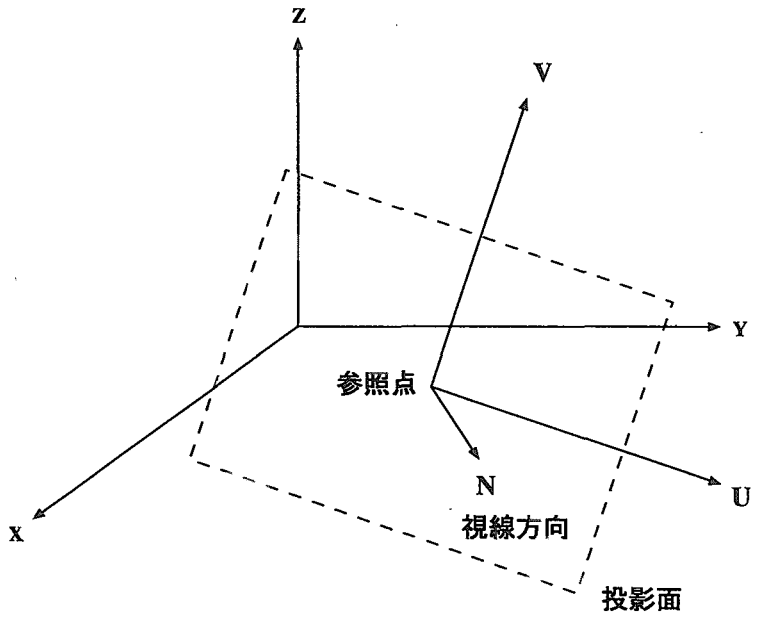
A.2　ワールド座標系

ワールド座標系 (WC) は物理空間内に設定された直交座標系で、XYZ 座標で表す。物体の形状や格子点の位置、視点位置等のビュー情報、光源の位置等の光源に関する情報は、この座標系で表される。

A.3 ビューイング変換とビューイング座標系

ワールド座標系は、ビューイング変換によりビューイング座標系に変換される。ビューイング座標系は、視点方向からの可視化表示を行なうために使用する座標系で、UVN 座標系で表す。表示面に向かって上垂直方向が V 軸正方向、右水平方向が U 軸正方向、奥方向（視線方向）が N 軸正方向となる。ワールド座標系とビューイング座標系を区別することにより、視点の移動といったビュー情報の変更を物理空間内の物体等の位置とは独立に考えることが可能となる。
ビューイング変換を行うには以下のパラメータを設定する必要がある。

- 表示参照点位置 ... ビューイング座標系の原点を物理空間内に設定する。また後述のウィンドウの中心となる。
- 視点位置 ... 視点位置を物理空間内に定義することにより、視点から参照点に向かう視線方向（N 軸方向）および視点から参照点までの距離（視点距離）を設定する。
- 投影面上向きベクトル ... V 軸方向を定義するためのベクトルである。必ずしも投影面に平行である必要はなく、投影面に射影されたベクトルの向きが V 軸方向となる。表示図に対し回転操作を行なったときには内部的に変更される。



XYZ軸 ; ワールド座標系
UVN軸 ; ビューイング座標系

A.4 クリッピング

ビューイング座標系において可視化の対象となる領域は、クリッピングボリュームと呼ばれる3次元領域により定義される。この領域内の可視化結果は、指定された投影法に従い正規化投影座標系に変換される。

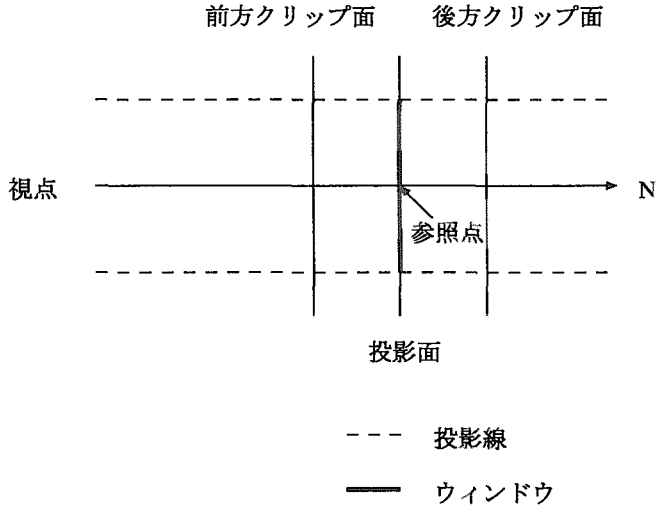
クリッピングボリュームを定義するためには以下のパラメータが必要である。

- 表示範囲設定 ...表示範囲として幅、高さ、奥行きを設定する。

ビューイング座標系におけるUV面をここではビュープレーン（投影面）と呼ぶ。表示範囲の幅、高さは、投影面上に四角形領域（ウィンドウ）を設定することで決定される。また奥行きは、ウィンドウ領域を投影面の前後に射影した前方クリップ面および後方クリップ面と呼ばれる2つの平面を設定することにより決まる。この前/後方クリップ面およびウィンドウの4つの頂点を通る投影線によって囲まれる領域はクリッピングボリュームと呼ばれ、可視化の対象範囲となる。クリッピングボリュームは、投影方法に応じて変化する。

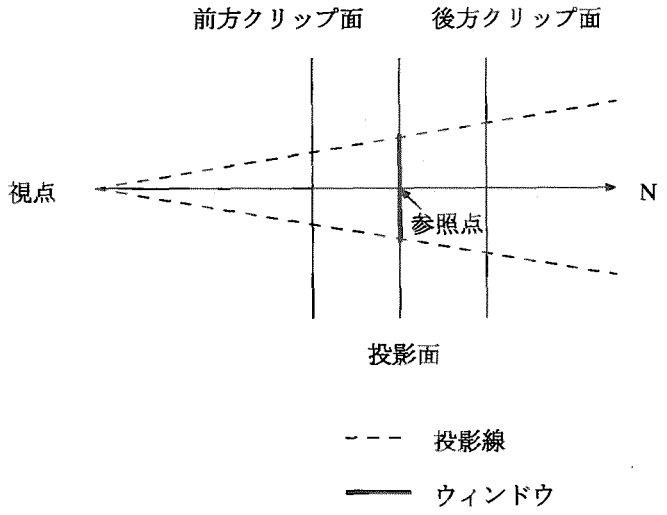
(1) 平行投影

3次元物体が遠くにあるときも近くにあるときも同じ大きさで表示する。投影線は、視点と参照点を結ぶ視線方向（N軸方向）に平行な直線となる。



(2) 透視投影

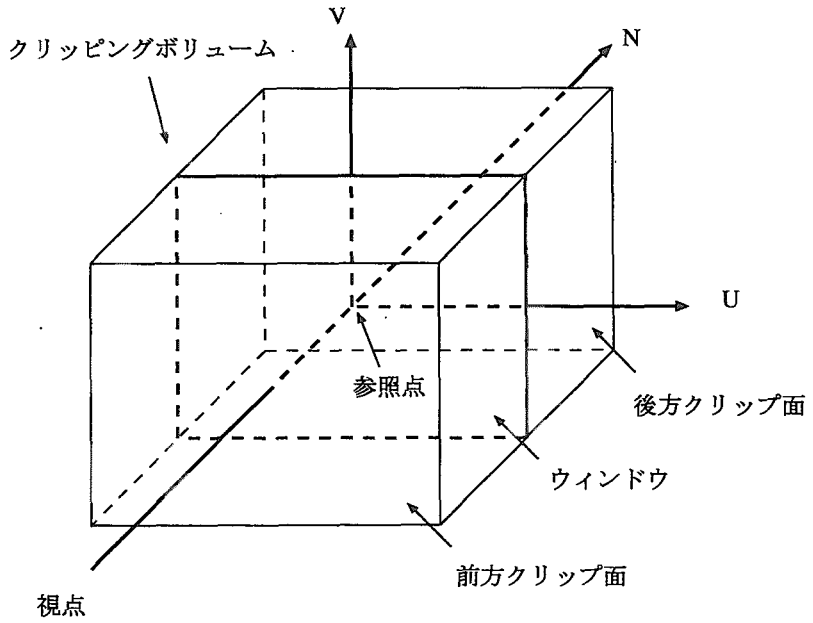
3次元物体が近くにあるときは、遠くにあるときよりも大きく表示する。投影線は、視点を通る直線となる。



例えば、

ウィンドウの幅 $= w$、ウインドウの高さ $= h$、
前方クリップ面の投影面からの距離 $= d_1$、
後方クリップ面の投影面からの距離 $= d_2$、

と指定した場合、平行投影の場合のクリッピングボリュームはビューイング座標系において、2点 $(-w/2, -h/2, -d_1)$ および $(w/2, h/2, d_2)$ を結ぶ線分を対角線とする直方体領域となる。



A.5 正規化投影座標系

クリッピングボリュームの写像が辺の長さ１の立方体となる座標系を、正規化投影座標系と呼ぶ。この座標系において、クリッピングボリューム内の可視化結果をウィンドウ上に平行投影する。投影結果は、メインウィンドウ内のビューポート (描画領域) に画像として表示される。本システムでは、ビューポートは正方形である。この処理の結果、ウィンドウが正方形でない場合、画像の縦横比が変化することになる。

付 録B　パラメータ設定ファイル

B.1　概要

パラメータ設定ファイルは、本システムの可視化パラメータの値を指定するためのものである。本システムにこのファイルを読み込ませると、各種可視化パラメータの値がファイルの内容に応じて更新され、ユーザインタフェース (GUI) のパラメータ入力画面に更新後の値が表示される。以後の可視化処理はこの値をもとに実行される。

本システムを起動すると、パラメータの既定値を設定するために用意されたファイルが必ず読み込まれる。これをパラメータ既定値設定ファイルと呼ぶ。このファイルをもとに、システムが初期化される。パラメータ既定値設定ファイルは以下の場所に用意されている。

$SYSTEM/resources/system.para

ここで、*$SYSTEM* は本システムのインストールディレクトリ名である。

ユーザは本システムのサーバ機能起動時に、GUI からパラメータ設定ファイルを指定する。このパラメータ設定ファイルは、パラメータ既定値設定ファイルの後に読み込まれる。

B.2　パラメータ設定ファイルの書式

ここではパラメータ設定ファイルの書式について説明する。なお、パラメータ既定値設定ファイルの書式とパラメータ設定ファイルの書式は同じである。

パラメータ設定ファイルは ASCII テキスト形式で、次のような構文になっている。最初の行 (#UI_LOG-1) は、必ず存在していなくてはならない。

#UI_LOG-1
procedure_name1 parameter_11_value parameter_12_value ...
procedure_name2 parameter_21_value parameter_22_value ...
parameter_2n_value parameter_2(n+1)_value ...
...

プロシージャ名 *procedure_name1*、*procedure_name2*、... は、関連する可視化パラメータのグループに対応する名前である。プロジージャ名以下には、対応するグループ内のすべてのパラメータの値 *parameter_**_value* を記述しなければならない。各パラメータの値の記述方法は、後述するようにパラメータのデータ型に依存する。隣接するパラメータの値はスペースで区切られ、各行は ⟨return⟩ で終わる。上に示したように、一つのプロシージャ名に対するパラメータ値の並びが複数行にまたがることもある。

プロシージャ名とこれに対応するパラメータ並びの一覧を次節に示す。

特別なプロシージャ名として、解析ソルバと可視化処理 1 回分の実行を示す "idle_proc" がある。"idle_proc" はパラメータ並びを持たない。"idle_proc" がファイルに記述されていると、パラメータ設定ファイルに従ったパラメータ値更新の際、解析ソルバまたは可視化処理が最低 1 回実行される。つまりこの場合、ファイルに記述されたすべてのパラメータの値を 1 度に反映できない。従って、"idle_proc" を含むファイルをシナリオセルファイルと呼び、パラメータ設定ファイルとは区別している (付録 C 参照)。

パラメータ設定ファイルの記述例を以下に示す。

Contour 1 ' ' true Coordinate_plane Contour_bands IJ 2.0 1.0 0.0 0.0 0.0 0.0 0.0

プロシージャ名 "Contour" に対するパラメータ並びは以下のようになっている。

Plane number = 1 (整数)
Update popup form = ' ' (パラメータ値はないが、' ' を記述しておく)
Plane selected = true (フラグ true または false)
Location type = Coordinate_plane (文字列)
Contour type = Contour_bands (文字列)
Coordinate_plane: direction = IJ (文字列)
Coordinate_plane: location = 2.0 (実数)
Arbitrary_plane: normal_x = 1.0 (実数)
Arbitrary_plane: normal_y = 0.0 (実数)
Arbitrary_plane: normal_z = 0.0 (実数)
Arbitrary_plane: location_x = 0.0 (実数)
Arbitrary_plane: location_y = 0.0 (実数)
Arbitrary_plane: location_z = 0.0 (実数)

パラメータ設定ファイルの具体的な記述のしかたについては、パラメータ既定値設定ファイル system.para も参考にされたい。

B.3 プロシージャ名およびパラメータ並び一覧

パラメータ設定ファイルに記述できるプロシージャ名と、これに対応するパラメータ並びの一覧を以下に示す。

プロシージャ名	パラメータ並び
Systemsize	Width(整数) Height(整数)
Systemmode	System mode(文字列) Time step interval(整数)
Network_compression	Network compression(文字列)
Colormode	Color mode(文字列) Dithering(フラグ)

プロシージャ名	パラメータ並び
Colortable	Table number(整数) Update popup form('') Max color(整数) \
	R(' 整数 * 128)
	R(整数 * 128') \
	G(' 整数 * 128)
	G(整数 * 128') \
	B(' 整数 * 128)
	B(整数 * 128') \
	I(' 整数 * 128)
	I(整数 * 128') Color bar('')
Background	Red(整数) Green(整数) Blue(整数)
Viewport	Projection(文字列) Eye x(実数) Eye y(実数) Eye z(実数) Look at x(実数) Look at y(実数) Look at z(実数) Up x(実数) Up y(実数) Up z(実数) Window size x(実数) Window size y(実数) Viewport size x(整数) Viewport size y(整数) Expansion rate(整数) Clipping Front(実数) Clipping Back(実数)
Light	Table number(整数) Update popup form('') Type(文字列) Intensity(整数) Red(整数) Green(整数) Blue(整数) Direction x(実数) Direction y(実数) Direction z(実数) Location x(実数) Location y(実数) Location z(実数)
Action_window_setup	3D－Cursor line width(文字列) Show wireframe cube at vrp(フラグ)
Rendering_setup	Rendering tool selected(フラグ) Scalar data(文字列) Min mapped range(実数) Max mapped range(実数) Color table number(整数) Min color index(整数) Max color index(整数) Min simulation value('no value') Max simulation value('no value')
Tracer_setup	Tracer tool selected(フラグ) Scalar data(文字列) Vector data(文字列) Min mapped range(実数) Max mapped range(実数) Color table number(整数) Min color index(整数) Max color index(整数) Min scalar simulation value('no value') Max scalar simulation value('no value') Min vector simulation value('no value') Max vector simulation value('no value')

プロシージャ名	パラメータ並び
Contour_setup	Contour tool selected(フラグ) Scalar data(文字列) Min mapped range(実数) Max mapped range(実数) Color table number(整数) Min color index(整数) Max color index(整数) Min simulation value('no value') Max simulation value('no value')
Object_setup	Object tool selected(フラグ) Scalar data(文字列) Min mapped range(実数) Max mapped range(実数) Color table number(整数) Min color index(整数) Max color index(整数) Min simulation value('no value') Max simulation value('no value')
Vector_setup	Vector tool selected(フラグ) Scalar data(文字列) Vector data(文字列) Min mapped range(実数) Max mapped range(実数) Color table number(整数) Min color index(整数) Max color index(整数) Min scalar simulation value('no value') Max scalar simulation value('no value') Min vector simulation value('no value') Max vector simulation value('no value')
Stream_line_setup	Stream_line tool selected(フラグ) Scalar data(文字列) Vector data(文字列) Min mapped range(実数) Max mapped range(実数) Color table number(整数) Min color index(整数) Max color index(整数) Min scalar simulation value('no value') Max scalar simulation value('no value') Min vector simulation value('no value') Max vector simulation value('no value')
Grid_line_setup	Grid_line tool selected(フラグ)
Isosurface_setup	Isosurface tool selected(フラグ) Scalar data(文字列) Min simulation value('no value') Max simulation value('no value')
Topography_setup	Topography tool selected(フラグ)
Rendering	Grid type(文字列) Interpolation(文字列) Element width(整数) Element height(整数) Transparency(文字列) Sampling point: specification(文字列) Sampling point: interval(文字列)

プロシージャ名	パラメータ並び
Tracer	Display type(文字列) Clear tracer('') Particle color Red(整数) Particle color Green(整数) Particle color Blue(整数) Max number of tracers(整数) Time step interval(整数) min_i(実数) max_i(実数) interval_i(実数) min_j(実数) max_j(実数) interval_j(実数) min_k(実数) max_k(実数) interval_k(実数)
Contour	Plane number(整数) Update popup form('') Plane selected(フラグ) Location type(文字列) Contour type(文字列) Coordinate_plane:direction(文字列) Coordinate_plane:location(実数) Arbitrary_plane:normal_x(実数) Arbitrary_plane:normal_y(実数) Arbitrary_plane:normal_z(実数) Arbitrary_plane:location_x(実数) Arbitrary_plane:location_y(実数) Arbitrary_plane:location_z(実数)
Object	Display type(文字列) Object number(整数) Update popup form('') Red(整数) Green(整数) Blue(整数)
Vector	Plane number(整数) Vector color Red(整数) Vector color Green(整数) Vector color Blue(整数) Vector scale(実数) Display interval(整数) Update popup form('') Plane selected(フラグ) Display type(文字列) Location type(文字列) Coordinate_plane:direction(文字列) Coordinate_plane:location(実数) Arbitrary_plane:normal_x(実数) Arbitrary_plane:normal_y(実数) Arbitrary_plane:normal_z(実数) Arbitrary_plane:location_x(実数) Arbitrary_plane:location_y(実数) Arbitrary_plane:location_z(実数)
Stream_line	Display type(文字列) Color red(整数) Color green(整数) Color blue(整数) Max number of stream lines(整数) Time step interval(整数) min_i(実数) max_i(実数) interval_i(実数) min_j(実数) max_j(実数) interval_j(実数) min_k(実数) max_k(実数) interval_k(実数) Number of time steps(整数)
Grid_line	Grid cube number(整数) Update popup form('') Color red(整数) Color green(整数) Color blue(整数) Display type(文字列) min_i(整数) max_i(整数) interval_i(整数) min_j(整数) max_j(整数) interval_j(整数) min_k(整数) max_k(整数) interval_k(整数)

プロシージャ名	パラメータ並び
Isosurface	Isosurface number(整数) Shading(文字列) Update popup form('') Color red(整数) Color green(整数) Color blue(整数) Display type(文字列) Isosurface value(実数) Transparency(整数)
Topography	Red(整数) Green(整数) Blue(整数)
Store_images	Store data(フラグ) Store at(文字列) File type(文字列) Base name of files(' 文字列') Step interval(整数) Max number of files(整数)
Ppm_replay	PPM pathname(文字列) Start PPM replay('')
Rmv_replay	RMV pathname(文字列) Start RMV replay('')
Tga_replay	TGA pathname(文字列) Start TGA replay('')

注意事項

(1) 横罫線で区切られたプロシージャ名およびパラメータ値の並びは、1 行に記述する。パラメータ並びが複数行にまたがる (1 つのプロシージャ名に対し、複数の横罫線による区切りが存在する) 場合、最初の行の先頭にのみプロシージャ名を記述する。

(2) プロシージャ名とパラメータ値の間、および隣接するパラメータ値の間はスペースで区切られ、1 行は ⟨return⟩ で終わる。

(3) 括弧 () 内は各パラメータのデータ型または記述するべき文字列を表す。また例えば、(' 整数*128) とある場合はアポストロフィ(') に続けて整数を 128 個記述し、(整数*128') とある場合は整数を 128 個記述した後にアポストロフィを記述する。

(4) (フラグ) には true または false を記述する。

(5) ('') および ('no value') の場合は、そのままこの文字列を記述する。また、表中のバックスラッシュ(\) は、そのままファイルに記述する。

付 録C　シナリオファイル

C.1　概要

シナリオファイルは、本システムの動作をあらかじめ指定するためのものである。シナリオファイルには、視点の動き (カメラワーク) など、本システムによる可視化のシナリオ (台本) を記述しておく。これを実時間可視化実行時に再生することにより、シナリオに則った可視化が自動的に行われる。例えば、*Viewport* 画面 (クライアントモジュールのオンラインマニュアル参照) 内の各種パラメータを変更するようなシナリオを記述しておけば、実時間可視化実行中に、表示図の回転、移動、サイズ変更が自動的に行われる。

シナリオファイルには次のものがある。

- ユニバーサルシナリオファイル (USF) : 可視化シナリオをキーフレーム方式により記述する。再生方法については 3.6 節参照。
- シナリオセルファイル : 本システムの動作を、可視化ステップごとに記述する。ただし、シナリオセルファイルを直接再生する機能は現在サポートしていない。ユニバーサルシナリオファイルをバッチ処理により再生した場合は、実行ログがシナリオセルファイルとして出力される。

以下では、各シナリオファイルについて説明する。

C.2　ユニバーサルシナリオファイル

ユニバーサルシナリオファイル (USF) の記述規則は以下の通りである。

- キーフレーム方式により記述する。キーフレームの記述は、$beginframe と$endframe の間で行う。
- 各キーフレームは解析時刻 (無次元) または解析ステップのいずれかに対応付けることができる。ただし、解析時刻に対応付けたキーフレームと解析ステップに対応付けたキーフレームの混在はできない。指定は、$beginframe に続けて以下のように行う。

```
time = 解析時刻        #解析時刻を対応させる場合
step = 解析ステップ    #解析ステップを対応させる場合
```

- 引き続くキーフレームの間で視点 (カメラ) を動かしたり、一定の間隔で可視化を行う場合は、$beginmotion と$endmotion で囲まれるブロックを記述する。このブロック内では、パラメータの補間方法、可視化の時刻間隔または時間ステップ数、作成する画像数、一つの画像に対するフレーム数が指定できる。一つの画像に対するフレーム数は動画ファイルを出力する場合に有効である。引き続くキーフレーム間で値の異なるパラメータは、指定された補間方法に従って逐次補間が行われる。補間方法としては線形補間が可能であり、さらに視点の位置について移動平均を施すことができる。

- システムモードを解析ソルバ中断モードにする場合は、同じ時刻または時刻ステップに対応する複数のキーフレームを記述する。この場合、自動的に解析ソルバ中断モードになる。このモードにおいても、パラメータの補間が行われる。

- 各パラメータは、メインキーワード、またはメインキーワードとサブキーワードで識別される。メイン/サブキーワードは、パラメータの意味を大まかに示す。パラメータ値の指定は、以下の形式で行う。

  メインキーワード = パラメータ値;

  メインキーワードとサブキーワードの両方をもつパラメータの値の指定は、$メインキーワード; と$end; の間で、以下の形式で行う。

  サブキーワード = パラメータ値;

  キーフレーム内には複数のパラメータに対する値指定が可能であるが、そのキーフレームで変更すべきパラメータについてのみ記述すればよい。各指定の最後は ";" で区切る。

- パラメータが配列形式の場合は、以下の形式のいずれでも指定が可能である。

  メイン (サブ) キーワード = パラメータ値_1　パラメータ値_2 ...;
  メイン (サブ) キーワード = パラメータ値_1, パラメータ値_2, ...;
  メイン (サブ) キーワード [要素番号] = パラメータ値;

- 同一キーフレーム内に、同じパラメータに対する指定が複数存在する場合、最後の指定が有効となる。

- 全パラメータの初期設定は、本システム起動時にパラメータ既定値設定ファイルおよびGUI上で指定されたパラメータ設定ファイル (付録 B 参照) を読み込むことにより行われる。ただし、最初のキーフレームにおいて全パラメータ値を指定することによっても可能である。

- 一つの行において、"#" 以降はコメントとみなす。

以下に、$beginframe と$endframe の間に記述できるキーワード名の一覧を示す。

メインキーワード	サブキーワード	サイズ	内容	補間
viewing	projection	1	投影方法 'parallel'、'perspective' のいずれか	×
	eye	3	視点の位置	○
	look_at	3	参照点の位置	○
	up	3	投影面上向きベクトル	○
	window_size	2	ウィンドウの大きさ	○
***_setup	selected	1	表示/非表示 'true'、'false' のいずれか	×

注意事項

(1) ***は rendering、tracer、stream_line、contour、object、vector、grid、isosurface のいずれかを表す。

(2) 補間欄は、当該パラメータがキーフレーム間で補間可能かどうかを表す。

○：補間が可能。
×：補間は不可能。

以下に、$beginmotion と$endmotion の間に記述できるキーワード名の一覧を示す。

メインキーワード	サイズ	内容
interpolation	1	引き続くキーフレーム間でのパラメータ補間方法 'linear'、'moving_average' のいずれか
interval	1	可視化の時刻間隔または時間ステップ数 (リアルタイムモードの場合のみ有効)
images	1	作成する画像数 (解析ソルバ中断モードの場合のみ有効)
slackness	1	一つの画像に対するフレーム数 (動画ファイル出力を行う場合のみ有効)

注意事項

(1) パラメータ補間方法として線形補間 ('linear') が指定された場合、隣接キーフレーム間でパラメータ値の線形補間が行われる。パラメータ補間方法として移動平均 ('moving_average') が指定された場合、上記に加えて視点の位置が移動平均される。移動平均の場合の実際の視点の位置は、線形補間の場合における前後 5 表示ステップでの視点の位置を平均した地点である。移動平均を行うと、視点の動きが滑らかになる。

(2) キーワード images は、解析ソルバ中断モードにおいて、隣接キーフレーム間で作成する画像の数を決定する。

(3) キーワード slackness には、一つの画像を (そのコピーも含めて) 何枚出力するかを決定する。大きな値を指定すると、動きの遅い動画が生成されるが、動画ファイルの容量は大きくなる。

(4) キーワード interval に、引き続くキーフレーム間の時刻間隔または時間ステップ数より大きな値が指定された場合、システムモードは自動的に可視化表示中断モードになる。

以下に、USF の記述例を示す。

```
#
# This is the 1st keyframe.
#
$beginframe time=0;
   $viewing;
      eye=1.0 0.0 0.0; look_at=0.0 0.0 0.0;
   $end;
   $object_setup;
      selected='true';
   $end;
$endframe;
#
# Camerawork between 1st and 2nd keyframes
#
$beginmotion;
   interpolation='linear'; interval=0.05; slackness=1;
$endmotion;
#
# This is the 2nd keyframe.
#
$beginframe time=10;
   $viewing;
      eye[1]=0.0; eye[2]=1.0;
   $end;
$endframe;
#
# Camerawork between 2nd and 3rd keyframes
#
$beginmotion;
   images=150;
$endmotion;
#
# This is the 3rd keyframe.
```

```
#
$beginframe time=10;
   $viewing;
      eye[2]=0.0; eye[3]=1.0;
   $end;
$endframe;
#
# This is the 4th keyframe.
#
$beginframe time=20;
$endframe;
```

このように記述された可視化シナリオの具体的内容は、以下の通りである。なお、特に断りの無い限り、シナリオ再生以前に設定されていたパラメータの値がそのまま有効である。

第 1 フレーム 解析時刻が 0 の時点で表示を開始。オブジェクトを表示。視点の位置は (1.0, 0.0, 0.0)、参照点の位置は (0.0, 0.0, 0.0)。

第 1 フレーム～第 2 フレーム 解析時刻が 0 から 10 までの間、リアルタイムモードで表示を実行。この間、視点を (1.0, 0.0, 0.0) から (0.0, 1.0, 0.0) に線形補間により徐々に移動。参照点の位置は不変。表示時刻間隔は 0.05。動画ファイル出力を行う場合は、1 つの画像を 1 フレームずつ出力。

第 2 フレーム この時点でシステムモードが解析ソルバ中断モードに変化。

第 2 フレーム～第 3 フレーム 解析ソルバ中断モードのままで、150 表示ステップの間 に視点を (0.0, 1.0, 0.0) から (0.0, 0.0, 1.0) に線形補間により徐々に移動。

第 3 フレーム この時点でシステムモードがリアルタイムモードに変化。

第 3 フレーム～第 4 フレーム パラメータの変更無しに、リアルタイムモードで解析時刻 20 まで表示を実行。表示時刻間隔は 0.05。動画ファイル出力を行う場合は、1 つの画像を 1 フレームずつ出力。

第 4 フレーム 引き続くキーフレームが存在しないため、可視化シナリオに則った処理を終了。

C.3 シナリオセルファイル

現在の実時間可視化システムにはシナリオセルファイルを読み込む機能はなく、ユニバーサルシナリオファイルをバッチ処理により再生した場合の実行ログである server-replay.log ファイルのフォーマットにのみ利用されている。

シナリオセルファイルには、本システムの動作を (アニメーションにおいてセル画を描くのと同様に) 可視化表示ステップごとに記述する。ファイルの記述形式はパラメータ設定ファイル (付録 B

参照) とほぼ同じであるが、シナリオセルファイルでは解析ソルバと可視化処理の実行についても指定できる。このために、解析ソルバと可視化処理 1 回分の実行を示すキーワード (プロシージャ名) "idle_proc" が用意されている。

以下では、シナリオセルファイルの記述方法を具体例を用いて説明する。

```
#UI_LOG-1
Systemmode Realtime_mode 1
idle_proc
idle_proc
idle_proc
Viewport Parallel 1.0 0.0 0.0 1.2 3.4 5.6 0.0 0.0 1.0 10.0 10.0 512 20.0 20.0
idle_proc
idle_proc
idle_proc
Viewport Parallel 0.0 1.0 0.0 1.2 3.4 5.6 0.0 0.0 1.0 10.0 10.0 512 20.0 20.0
idle_proc
idle_proc
idle_proc
```

このシナリオセルファイルに則った本システムの動作は次の通りである。まず、システムモードはリアルタイムモードに、リアルタイムモード時の Time step interval(クライアントモジュールのオンラインマニュアル参照) は 1 に設定される。その後、プロシージャ名 idle_proc が 3 回指定されており、解析ソルバが 3 ステップ実行される。Time step interval は 1 に設定されているため、可視化処理解析ソルバの各ステップの後に実行される。Time step interval が例えば 3 に設定されている場合、可視化処理解析ソルバが 3 回実行された後に 1 回だけ実行される。

シナリオセルファイルの例を、もう 1 つ示す。

```
#UI_LOG-1
Systemmode Halt_flowsolver 1
idle_proc
idle_proc
Viewport Parallel 0.0 1.0 0.0 1.2 3.4 5.6 0.0 0.0 1.0 10.0 10.0 512 20.0 20.0
idle_proc
```

このシナリオセルファイルに則った本システムの動作は次の通りである。まず、システムモードは解析ソルバ中断モードに設定される。その後、プロシージャ名'idle_proc' が 2 回指定されているが、システムモードが解析ソルバ中断モードとなっているため、解析ソルバは実行されない。また、解析ソルバ中断モードでは、可視化処理可視化パラメータが変更された場合にのみ実行される。'idle_proc' の 1 回目の指定と 2 回目の指定の間で可視化パラメータは変更されていないので、'idle_proc' の 2 回目の指定では可視化処理は実行されない。'idle_proc' の 3 回目の指定では、変更された *Viewport* 画面関連のパラメータを用いて可視化処理が実行される。

付 録D　NetCDFファイル

NetCDF とは、Network Common Data Form の略で、Unidata Program Center の Glenn Davis、Russ Rew、および Steve Emmerson によって開発された、配列指向のデータアクセスのためのインターフェース、およびそのインターフェースの実装を与える C、FORTRAN、C++ソフトウェアライブラリである。

本システムのポストプロセッシング機能で取り扱う NetCDF データの形式は、以下の様な CDL (Common Data Form Language) 書式を用いたものである。

(1) 書式概要

- フリーカラム。
- フィールドデリミッタはカンマ。
- // 以降はコメント文。
- 次元、変数および属性の名前は、任意の英数字とアンダースコア "_" およびハイフン "-" の列で構成され、先頭は文字またはアンダースコアでなければならない。
- 大文字小文字は区別される。

(2) netcdf

NetCDF 形式のデータを設定する。tital は、ncgen ユーティリティを使用して、NetCDF 形式のデータを作成するときに、デフォルトのファイル名 "tital.nc" に使用される。

(3) dimensions:

次元データを設定する。次元データは、正の整数値もしくは、unlimited で無制限次元を宣言する。無制限次元は、1 つの NetCDF データに 1 つしか宣言できない。
次元の宣言は、名前＝サイズ の形式となる。
今回の実時間可視化システム対応のフォーマットでは、座標データは、座標変数を、取得するスカラデータおよびベクトルデータは、4 次元（3 つの変数座標の次元と、1 つの無制限次元からなる）のデータを、取得するようになっているため、次元データの宣言は、最低でも、3 つの変数座標の次元と、1 つの無制限次元を宣言する。また無制限次元には、時間を宣言する。

(4) variables:

変数および属性を設定する。変数は、データ型、変数名、および変数が生成された際に指定された次元のリストで構成される。属性は、関連する変数に対しての名前、型、長さ、および値から構成される。型、および長さは、CDL 表記では、宣言されない。これらは、属性に

割り当てられた値から得られる。
変数の宣言は、次元を持つ変数の場合:
　　型　変数名 (次元名 1、次元名 2、…);
スカラー変数の場合:
　　型　変数名;
属性の宣言は、変数の属性の場合:
　　変数名:属性名　＝　値のリスト;
グローバル属性の場合:
　　:属性名　＝　値のリスト;
となる。
今回の実時間可視化システム対応のフォーマットでは、変数については、座標データは、座標変数として、また取得するスカラデータおよびベクトルデータは、4 次元のデータとして変数を宣言しなければならない。また、4 次元変数の次元リストは、無制限次元 (時間)、Z 座標、Y座標、X標の順に宣言し、データとして取得するものは、常に同じ次元リストになるように宣言しなければならない。変数の型は、int、float、もしくは、double で宣言し、変数名は、15 文字以内にしなかればならない。(補足 1、補足 2)。
属性については、何の制約もない。
補足 1: 変数名は、15 文字より多くてもかまわないが、RVS_NETCDF ルーチンでは、取得する変数名を最初の 15 文字で判断するので、変数名が 15 文字より多くて、最初の 15 文字まで同じ並びの文字列であると、RVS_NETCDF ルーチンで取得できないことがある。
補足 2: 変数名に "_NOT" は使用できない。この文字列は、RVS_NETCDF ルーチンで使用している。

(5) data:
変数データの割り付けを行う。
データについては、何の制約もない。

以下は、CDL 表記の簡単な例。

```
netcdf example_1 {        // NetCDF ファイルに対する CDL 表記の例
dimensions:               // 次元名およびサイズが宣言される
        lat = 5, lon = 10, level = 4, time = unlimited;
variables:                // 変数の型、名前、形、属性
        float   temp( time, level, lat, lon );
                          temp: long_name = "temperature";
                          temp: units     = "celsius";
        float   rh( time, level, lat, lon );
                          rh: long_name   = "relative humidity";
                          rh: valid_range = 0.0, 1.0 ;     //最小、最大値
        int     lat( lat ), lon( lon ), level( level );
                          lat: units      = "degrees_north";
                          lon :units      = "degrees_east";
```

```
                level: units     = "millibars";
        short   time( time );
                time: units      = "hours since 1996-1-1";
        // グローバル属性
                : source         = "Fictional Model Output";
data:           //オプションのデータ割り当て
        level   = 1000, 850, 700, 500;
        lat     = 20, 30, 40, 50, 60;
        lon     = -160, -140, -118, -96, -84, -52, -45, -35, -25, -15;
        time    = 12;
        rh      = 0.5, 0.2, 0.4, 0.2, 0.3, 0.2, 0.4, 0.5, 0.6, 0.7,
                  0.1, 0.3, 0.1, 0.1, 0.1, 0.1, 0.5, 0.7, 0.8, 0.8;
}
```

付 録E　GrADS形式ファイル

GrADS とは COLA(Center for Ocean-Land-Atmosphere Studies：コロンビア大学海洋陸面大気研究所）で開発されたフリーの気象データプロット用ソフトウエアである。GrADS 形式のデータは gridded data と station data があるが、本システムのポストプロセッシング機能で取り扱うのは gridded data である。gridded data はデータ記述ファイル (data descriptor file) と実データファイル (binary data set) の２種類のファイルから構成される。実データファイルは、物理量などの計算結果がバイナリー形式で格納されており、データ記述ファイルにはそれを読み込むための制御パラメータが格納されている。ポストプロセッシング機能では、制御パラメータのうち主なものに対応している。制御パラメータの情報については以下の表にまとめる。
表で対応欄に○印がついているものには対応、×印がついているものには未対応である。OPTIONS パラメータのサブパラメータは同時に複数の指定が可能である。また、パラメータ記述で大文字、小文字による区別はない。

フィールド1	フィールド2	フィールド3	フィールド4	フィールド5	対応
DSET	ファイル名				○
TITLE	タイトル文字列				○
UNDEF	実数値				○
OPTIONS	YREV				○
	ZREV				○
	SEQUENTIAL				○
	BYTESWAPPED				×
	TEMPLATE				×
	BIG_ENDIAN				×
	LITTLE_ENDIAN				×
	CRAY_32BIT_IEEE				×
XDEF	Number(整数)	LINEAR	Start(実数)	Increment (実数)	○
		LEVELS	値1	値2 ...	○

フィールド1	フィールド2	フィールド3	フィールド4	フィールド5	対応
YDEF	Number(整数)	LINEAR	Start(実数)	Increment (実数)	○
		GAUSR15	Start(実数)		×
		GAUSR20	Start(実数)		×
		GAUSR30	Start(実数)		×
		GAUSR40	Start(実数)		×
		LEVELS	値1	値2 ...	○
ZDEF	Number(整数)	LINEAR	Start(実数)	Increment (実数)	○
		LEVELS	値1	値2 ...	○
TDEF	Number(整数)	LINEAR	Start-time (文字列)	Increment (実数)	○
VARS	Number(整数)				○
Abrev(文字列)	Lves(整数)	Units(整数)	コメント文字列		○
	Lves(整数)	-1,10,yy	コメント文字列		×
	Lves(整数)	-1,20,yy	コメント文字列		×
	Lves(整数)	-1,30,yy	コメント文字列		×
	Lves(整数)	-1,40,yy	コメント文字列		×
ENDVARS					○
FILEHEADER	Length(整数)				×
THEADER	Length(整数)				×
XYHEADER	Length(整数)				×

付 録F　AVS Field Dataファイル

F.1　概要

AVS(Application Visualization System) は米国 Advanced Visual SystemsInc. が開発したビジュアライゼーションのツールである。AVS が扱うデータ形式の一つに Field Data と呼ばれる形式がある。Field Data はメッシュ状に並んだ格子点上にデータ値が定義されている構造型のデータである。また、AVS Field Data は以下に示す 3 種類のフォーマット ((1)～(3)) がある。

(1) アスキーで記述するヘッダ部分とバイナリで記述するデータ部分、座標部分から構成され、ヘッダ部分とデータ部分の間にセパレータ（^L^L）が挿入される。ヘッダ部分にはデータの次元数、各軸方向のサイズ、データの型などを記述する。1 ステップ分のデータファイルである。

ヘッダ部 (アスキー)
セパレータ (^L^L)
データ部分 (バイナリ)
座標部分 (バイナリ)

(2) ヘッダ部分、データ部分、座標部分を別ファイルとする。ヘッダ部分にデータ部分、座標部分のファイル名を記述。データ部分、座標部分のファイルはアスキー、バイナリどちらでも可。

(3) 上記 (2) のフォーマットをベースにマルチステップ（時系列）のデータを扱えるように拡張したもの。データ部分、座標部分のファイルはアスキーに限られる。

ポストプロセッシング機能では、このうち (3) のマルチステップデータに対応する。

F.2　AVS Field Data ヘッダ部の書式

本節では、AVS Field Data のマルチステップデータのヘッダ部の書式について説明する。一般に AVS Field Data ファイルといった場合には、上記 (1) のフォーマットの場合を除いて、ヘッダ部のファイルを指す。本マニュアルでもそれに従っている。

AVS Field Data ファイル名の拡張子は ".fld" である。

マルチステップデータのヘッダ部の記述方法は 2 種類ある。データ部分や座標部分のファイル指定を各ステップ毎に記述する方法と、1 ステップ分の記述をループで回す方法である。まず、ステップ毎の方法を説明し、後でループで回す方法を説明する。また、ファイルの記述に関して、ファイル名の指定以外は大文字、小文字を区別しない。

F.2.1 ステップ毎に記述する方法

以下にヘッダ部分の記述例を示す。

```
# AVS field file
#
ndim=2
dim1=4
dim2=2
nspace=3
veclen=1
data=float
field=irregular
time file=sample.dat filetype=ascii skip=1 close=1
variable 1 file=sample.dat filetype=ascii skip=2 offset=0 stride=4 close=1
coord 1 file=sample.dat filetype=ascii skip=2 offset=1 stride=4 close=1
coord 2 file=sample.dat filetype=ascii skip=2 offset=2 stride=4 close=1
coord 3 file=sample.dat filetype=ascii skip=2 offset=3 stride=4 close=1
EOT
time file=sample.dat filetype=ascii skip=10 close=1
variable 1 file=sample.dat filetype=ascii skip=11 offset=0 stride=4 close=1
coord 1 file=sample.dat filetype=ascii skip=11 offset=1 stride=4 close=1
coord 2 file=sample.dat filetype=ascii skip=11 offset=2 stride=4 close=1
coord 3 file=sample.dat filetype=ascii skip=11 offset=3 stride=4 close=1
EOT
(time から EOT までが 1 ステップ分のデータ記述)
```

ここで、データファイル sample.dat には以下のデータがあるとする。

```
data X Y Z
1
5.0  -2.0 -1.0 0.0
10.0 -1.0 -1.0 0.0
20.0  0.0 -1.0 0.0
25.0  1.0 -1.0 0.0
30.0 -2.0  0.0 0.0
40.0 -1.0  0.0 0.0
50.0  0.0  0.0 0.0
60.0  2.0  0.0 0.0
2
5.0  -2.0 -1.0 0.0
10.0 -1.0 -1.0 0.0
```

```
20.0  0.0 -1.0 0.0
25.0  1.0 -1.0 0.0
30.0 -2.0  0.0 0.0
40.0 -1.0  0.0 0.0
50.0  0.0  0.0 0.0
60.0  2.0  0.0 0.0
3
5.0  -2.0 -1.0 0.0
10.0 -1.0 -1.0 0.0
20.0  0.0 -1.0 0.0
50.0  2.0 -1.0 0.0
35.0 -2.0  0.0 0.0
45.0 -1.0  0.0 0.0
55.0  0.0  0.0 0.0
95.0  2.0  0.0 0.0
```

各キーワードの説明

(1) #␣AVS␣field␣data
必須行。␣は空白を表現。大文字、小文字、空白は正確に記述する。２行目以降#で始まる行はコメント行。

(2) ndim=⟨ 数値 ⟩
計算空間での次元数

(3) dim1=⟨ 数値 ⟩、dim2=⟨ 数値 ⟩、dim3=⟨ 数値 ⟩
各軸方向のデータのサイズ。
ndim が 1 の場合は dim1 のみを指定する。
ndim が 2 の場合は dim1 と dim2 を指定する。
ndim が 3 の場合は dim1 と dim2 と dim3 を指定する。

(4) nspace=⟨ 数値 ⟩
物理空間の次元数
X,Y 空間の場合、nspace=2
X,Y,Z 空間の場合、nspace=3

(5) veclen=⟨ 数値 ⟩
物理量データの数

(6) data=⟨ 文字列 ⟩
物理量データのデータの型。複数の物理量がある場合、すべて同じデータ型でなければならない。以下の中から選択。

```
short
byte
integer
float
double
```

(7) field=⟨ 文字列 ⟩
座標情報。以下の中から選択

uniform	直交等間隔格子：座標データが不要
rectilinear	直交不等間隔格子：軸方向の座標データを定義
irregular	その他の構造格子

(8) label=⟨ 文字列 ⟩ ⟨ 文字列 ⟩...
物理量データにラベルをつける。オプションの項目。

(9) time 行、variable 行、coord 行は、ほぼ同じキーワードで構成される。

- time 行
  time file=⟨ 文字列 ⟩ filetype=⟨ 文字列 ⟩ skip=⟨ 数値 ⟩ close=⟨ 数値 ⟩
  または、
  time value=⟨ 文字列 ⟩
  time 行は、ステップ毎に画像に表示したいコメント（時刻やステップ番号）を指定する。コメントがファイルに格納されている場合は、time file=... を記述し、直接指定する場合は time value=を記述する。time 行はオプションであり、必須項目ではない。
- variable 行
  variable n file=⟨ 文字列 ⟩ filetype=⟨ 文字列 ⟩ skip=⟨ 数値 ⟩ offset=⟨ 数値 ⟩ stride=⟨ 数値 ⟩ close=⟨ 数値 ⟩
  variable 行は、データファイルに関する情報を指定する。
- coord 行
  coord n file=⟨ 文字列 ⟩ filetype=⟨ 文字列 ⟩ skip=⟨ 数値 ⟩ offset=⟨ 数値 ⟩ stride=⟨ 数値 ⟩ close=⟨ 数値 ⟩
  coord 行は、座標ファイルに関する情報を指定する。
  第 1 ステップ以降、座標値が変化しない場合は記述不要。uniform 型のデータの場合も記述不要。

(a) n
variable 行の場合は、各データ成分を区別するための番号で、1 から始まる整数で指定する。
coord 行の場合は、座標軸を区別する番号で、1 から始まる整数で指定する。1 が X、2 が Y、3 が Z。

(b) file=⟨ 文字列 ⟩
読み込むファイル名を指定する。絶対パス、相対パスのどちらでも可。ディレクトリの区切りは "/" または "\"。

(c) filetype=⟨ 文字列 ⟩
読み込むファイルがアスキーかバイナリかを指定する。マルチステップデータの場合は必ず ascii を指定

(d) skip=⟨ 数値 ⟩
最初にどれだけのデータを読み飛ばすかを指定する。
アスキーの場合は読み飛ばす行数を指定
バイナリの場合は読み飛ばすバイト数を指定
既定値は 0。

(e) offset=⟨ 数値 ⟩
読み飛ばすカラム数を指定。たとえば 4 カラム目からデータを読み込む場合は offset=3 と指定する。既定値は 0。
filetype=ascii の場合のみ有効。

(f) stride=⟨ 数値 ⟩
データとデータの間の読み飛ばし間隔を指定する。
アスキーの場合は、読み飛ばす項目数を指定。データ間のデリミッタは、1 つ以上の空白。
バイナリの場合は読み飛ばすバイト数を指定。
既定値は 1。

(g) close=⟨ 数値 ⟩
現在のステップのデータを読み込み後にファイルをクローズするかどうかの指定。
close=0 ならば、ファイルをクローズしない。
close=1 ならば、ファイルをクローズする。
このキーワードはマルチステップデータのみのキーワード。
既定値は 1

(10) EOT
ステップの終わりを指定する。

1 つの time,variable,coord で指定されたファイルのファイルポインタは個別に管理されている。たとえ同じファイルを指定していても close などの値は個別に設定できる。また skip などの値も個別管理されているものとして指定しなくてはならない。

F.2.2 ループで回す方法

以下にヘッダ部分の記述例を示します。

```
# AVS field file
#
nstep=3
ndim=2
```

```
dim1=4
dim2=2
nspace=3
veclen=1
data=float
field=irregular
time file=sample.dat filetype=ascii skip=1 close=0
variable 1 file=sample.dat filetype=ascii skip=2 offset=0 stride=4 close=0
coord 1 file=sample.dat filetype=ascii skip=2 offset=1 stride=4 close=0
coord 2 file=sample.dat filetype=ascii skip=2 offset=2 stride=4 close=0
coord 3 file=sample.dat filetype=ascii skip=2 offset=3 stride=4 close=0
EOT
DO
time file=sample.dat filetype=ascii skip=8 close=0
variable 1 file=sample.dat filetype=ascii skip=1 offset=0 stride=4 close=0
coord 1 file=sample.dat filetype=ascii skip=1 offset=1 stride=4 close=0
coord 2 file=sample.dat filetype=ascii skip=1 offset=2 stride=4 close=0
coord 3 file=sample.dat filetype=ascii skip=1 offset=3 stride=4 close=0
EOT
ENDDO
```

追加されたキーワード

(1) nstep=⟨ 数値 ⟩
ステップ数を指定する。上述のステップ毎に記述する方法でも nstep キーワードの指定はできる。その場合、実際のステップ数は nstep の値と EOT の数の小さいほうがとられる。

(2) DO, ENDDO
このキーワードの間の指定を繰り返し数が nstep になるまで繰り返す。skip などの設定値を各ステップで同じにするためには close=0 でなければならない。

F.3 ポストプロセッシング機能での制限事項、注意事項

(1) 計算空間の次元数 ndim において、ndim=1 の指定はサポートしていない。

(2) 物理量データの型 data の選択肢中、short と byte の指定はエラーとしている。これは、マルチステップデータはファイルタイプがアスキーに限定されるためである。

(3) 物理量データにラベルを付ける label はオプションの項目である。だが、PATRAS では、ラベル指定は必須であるので、label 指定がない場合にはポストプロセッシング機能のほうで、LABEL1,LABEL2,... というラベルを付与する。

(4) time 行の指定は無視する。

(5) ステップ毎に座標データを指定する場合、座標データはすべてのステップで同一のデータでなければならない。

(6) ファイル名の指定は、フルパス、相対パス、ファイル名だけのいずれの方法でも可能である。ただし、ファイル名だけの場合には、ポストプロセッシング機能を起動したカレントディレクトリにファイルがなければならない。また、同様に相対パスで指定する場合もカレントディレクトリからの相対パス指定ということになる。

(7) Field Data ファイルやデータファイルは 1 行 255 文字までで記述する。

付 録G　RMVコンバータ

G.1　概要

RMV コンバータは、地球シミュレータ用実時間可視化システム 独自のアニメーションファイル形式である RMV 形式のファイルを、Windows で標準的なアニメーションファイル形式である AVI 形式に変換するプログラムである。RMV コンバータは、Windows95,98/NT4.0 上で動作する。

G.2　利用方法

マイコンピュータもしくはエクスプローラを用い、rmvconv.exe を置いたフォルダを開き、rmvconv.exe をマウスでダブルクリックしてツールを起動する。

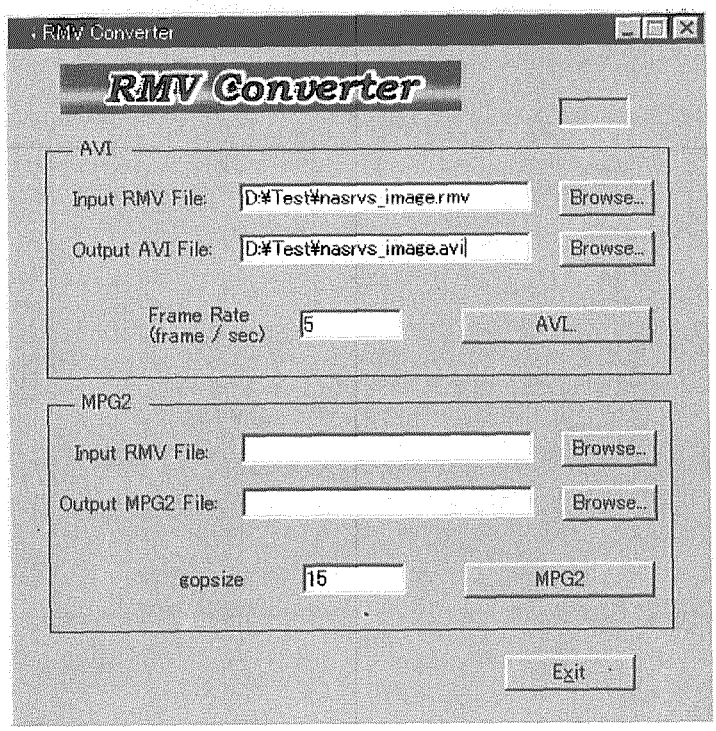


図 G-1: 画面イメージ

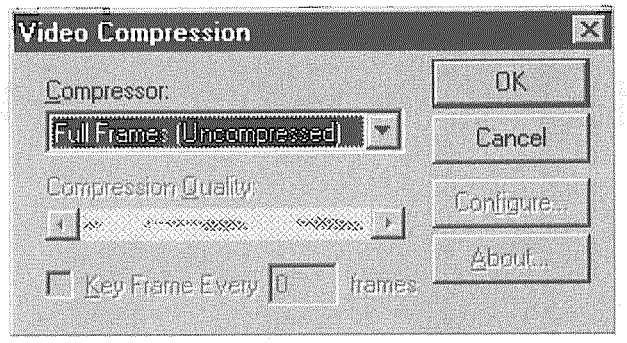


図 G-2: ダイアログウィンドウ

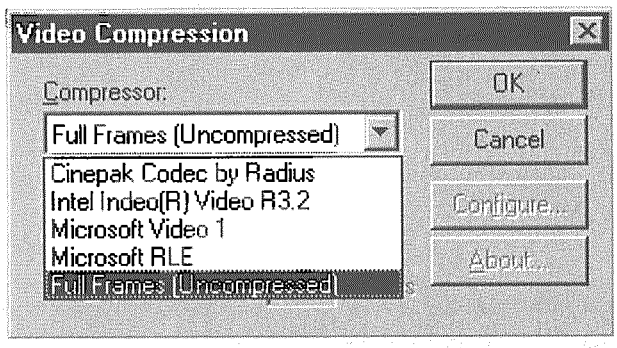


図 G-3: Microsoft Video 1 を選択

G.2.1 RMV 形式から AVI 形式に変換する場合

(1) ファイル名の指定

RMV コンバータの GUI の Input RMV File のテキストボックスに変換する RMV ファイルのファイル名を入力する。Browse ボタンにより、ファイルブラウザで選択することもできる。RMV ファイルはあらかじめ PC 上に転送しておく。次に、Output AVI File のテキストボックスに出力する AVI ファイルのファイル名を入力する。Browse ボタンにより、ファイルブラウザで選択することもできる。ここにファイル名だけ指定すると、RMV ファイルと同じディレクトリに AVI ファイルが作成される。

(2) パラメータの指定

RMV コンバータの GUI の Frame Rate で、1 秒あたりのフレーム数を変更できる。既定値は 5 フレーム/秒である。

(3) AVI 形式への変換

ファイル名を指定し、RMV コンバータの GUI の AVI ボタンをクリックするとビデオの圧縮ダイアログが表示される。ダイアログの圧縮プログラムから、Microsoft Video 1 を選択する。選択すると、圧縮の品質が設定可能になる。キーフレームの設定も行えるが、指定しな

くてもよい。ダイアログの OK ボタンをクリックすると、変換が開始される。変換中には、RMV コンバータの GUI の右上のボックスに、変換が終了したフレーム数が表示される。変換が終了すると、終了メッセージを表示したダイアログが表示される。

G.2.2 RMV 形式から MPEG2 形式に変換する場合

(1) ファイル名の指定

RMV コンバータの GUI の Input RMV File のテキストボックスに変換する RMV ファイルのファイル名を入力する。Browse ボタンにより、ファイルブラウザで選択することもできる。RMV ファイルはあらかじめ PC 上に転送しておく。次に、Output MPG2 File のテキストボックスに出力する MPEG2 ファイルのファイル名を入力する。Browse ボタンにより、ファイルブラウザで選択することもできる。ここにファイル名だけ指定すると、RMV ファイルと同じディレクトリに MPEG2 ファイルが作成される。

(2) パラメータの指定

RMV コンバータの GUI の Frame Rate で、GOP 内フレーム数を変更できる。既定値は 15 である。

(3) MPEG2 形式への変換

ファイル名を指定し、RMV コンバータの GUI の MPG2 ボタンをクリックすると、変換が開始される。変換中には、RMV コンバータの GUI の右上のボックスに、変換が終了したフレーム数が表示される。変換が終了すると、終了メッセージを表示したダイアログが表示される。

G.3 動作環境

- 対応 OS: Windows95/98, Windows NT4.0
- メモリ: 64MB 以上

G.4 注意事項

- AVI 形式に変換する際、ビデオの圧縮ダイアログの圧縮プログラムの選択肢には、その PC で再生可能な圧縮形式の一覧が表示されるが、RMV コンバータで対応している圧縮形式は Microsoft Video 1 のみである。
- AVI 形式は縦横のピクセルサイズが 4 の倍数なので、変換する RMV ファイルの画像サイズも 4 の倍数が望ましい。
- MPEG2 形式は縦横のピクセルサイズが 16 の倍数なので、変換する RMV ファイルの画像サイズも 16 の倍数が望ましい。

付 録H　マルチビューRMVプレイヤ

H.1　概要

マルチビューRMVプレイヤは、地球シミュレータ用実時間可視化システム独自のアニメーションファイル形式であるRMV形式のファイルの同期再生を行うプログラムである。

H.2　利用方法

マルチビューRMVプレイヤの実行には、Java2実行環境が必要である。Java2実行環境は、SUNもしくはOS提供元のホームページで入手可能である。

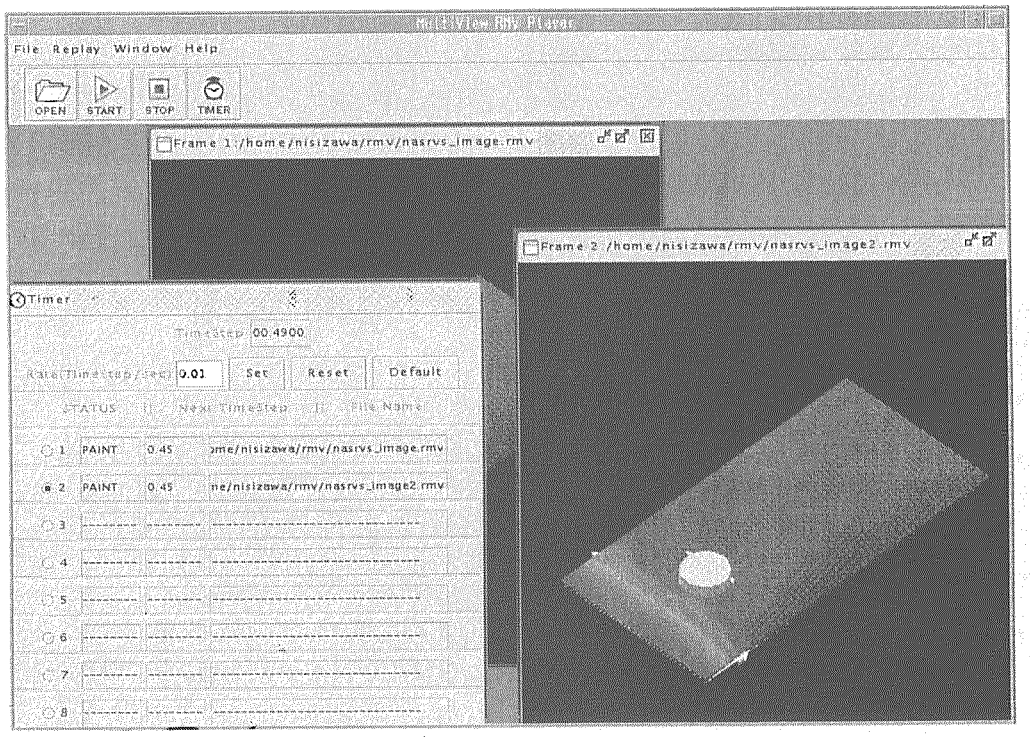


図 H-1: 画面イメージ

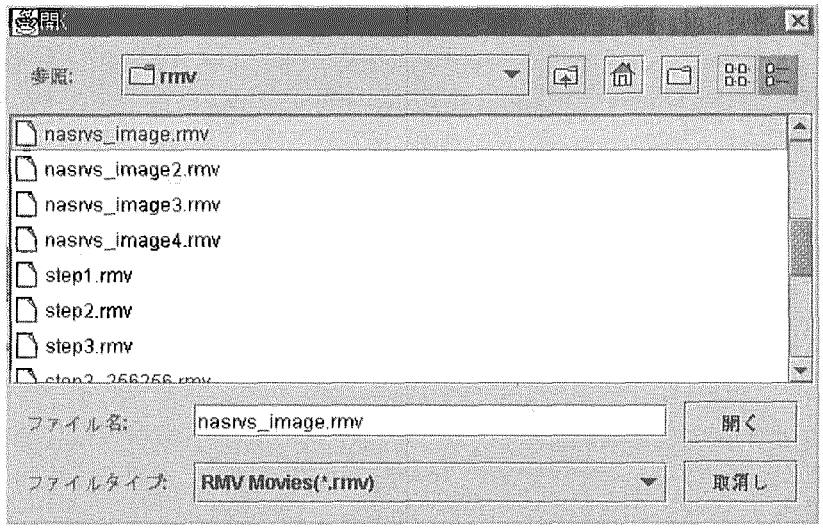


図 H-2: ファイルを選択

H.2.1 PC での起動方法

Windows マシンでは、マイコンピュータもしくはエクスプローラを用い、rmvplay.jar を置いたフォルダを開き、rmvplay.jar をマウスでダブルクリックしてツールを起動する。

H.2.2 ワークステーションでの起動方法

UNIX マシンでは、以下のようなコマンドを実行する。以下は java コマンドが/opt/java1.2/bin/にインストールされている場合の例である。

```
% limit datasize unlimited
% limit stacksize unlimited
% /opt/java1.2/bin/java -mx64m -jar rmvplay.jar
```

-mx64m の部分は、機種や Java2 実行環境のバージョンによっては-Xmx64m の場合もある。

H.2.3 操作方法

H.2.3.1 基本操作

(1) ファイルを開く

ツールバーのファイルボタンもしくはメニューバーの「File - Open」からファイル選択ダイアログウィンドウを開き、RMV ファイルを選択する。

ファイル選択ダイアログウィンドウから一度に選択可能なファイルは 1 個であるが、操作を繰り返すことにより、最大 8 個の RMV ファイルを開くことができる。

(2) 再生の開始

ツールバーのスタートボタンもしくはメニューバーの「Replay - Start」を選択することにより、RMV ファイルの再生が開始される。この際、ツールバーのスタートボタンはポーズボタンに変わる。

(3) 再生の一時停止

ツールバーのポーズボタンもしくはメニューバーの「Replay - Pause」を選択することにより、再生を一時停止できる。この際、ツールバーのポーズボタンはリスタートボタンに変わる。

(4) 一時停止の解除 (再開)

ツールバーのリスタートボタンもしくはメニューバーの「Replay - Pause」を再度選択することにより、一時停止を解除することができる。この際、ツールバーのリスタートボタンはポーズボタンに変わる。

(5) 再生の中止

ツールバーのストップボタンもしくはメニューバーの「Replay - Stop」を選択することにより、再生を中止できる。この際、ツールバーのポーズもしくはリスタートボタンはスタートボタンに戻る。中止すると中止時点からの再開はできない。中止した後スタートボタンを押すと、初期フレームから再生をやりなおす。

H.2.3.2 タイマーウィンドウ

メインウィンドウ内に、タイマーウィンドウが表示される。

マルチビュー RMV プレイヤはアプリケーション内で動くシステムタイマーと、RMV ファイルのフレームのタイムステップ情報が一致したとき、フレームの描画を行う。システムタイマー、1秒あたりのタイムステップ、各 RMV ファイルのステータスを表示するのがタイマーウィンドウの役割である。

以下は各部の説明である。

タイマーウィンドウ上部はタイマー操作を行う GUI が占める。

- TimeStep

  タイムステップを表示。0 から始まり、再生中は 1 秒あたり Rate ずつタイムステップが増える。

- Rate

  1 秒あたりに進ませるタイムステップ。既定値は 0.005 ステップ／秒である。

- Set ボタン: Rate の更新
- Reset ボタン: Rate を現在設定されている値に戻す
- Default ボタン: Rate を既定値に戻す

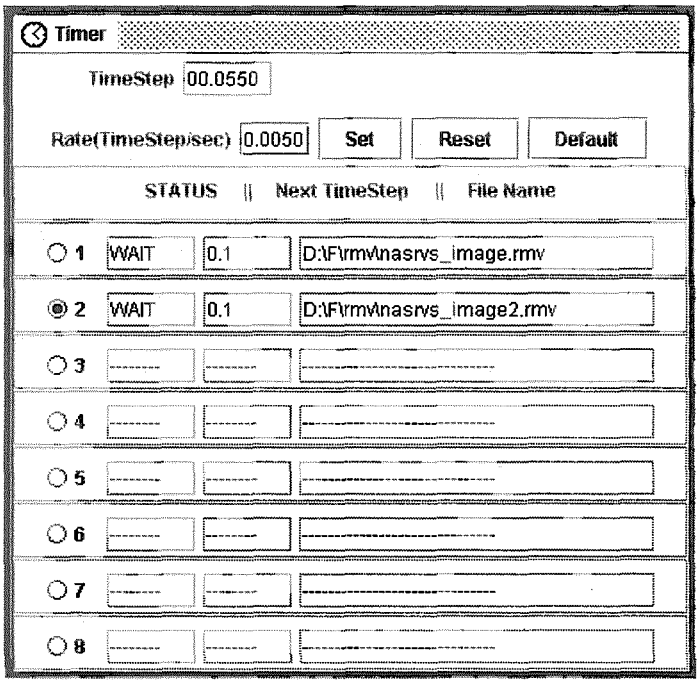


図 H-3: タイマーウィンドウ

タイマーウィンドウ下部は各 RMV ファイルの状態表示を行う。

- Status: ファイルの状態

  WAIT: 次のステップを待つ
  READING: ピクセル読み込み中
  PAINT: ピクセルを表示済み

- Next TimeStep: 次に表示するタイムステップ
- File Name: 表示されている RMV ファイルのファイル名
- 番号の横のラジオボタンを押すと、該当のウィンドウが手前に表示される

タイマーウィンドウが RMV ファイルの下に隠れてしまった際には、ツールバーのタイマーボタンもしくはメニューバーの「Window - Timer Window」を選択すると、手前に表示される。

H.2.3.3 終了方法

メニューバーの「File - Exit」を選択するか、ウィンドウマネージャを用いてマルチビュー RMV プレイヤを終了することができる。その際、終了確認ダイアログウィンドウが表示されるので、終了したい場合は「Yes」を、キャンセルの場合は「No」を選択する。

H.3 動作環境

- 対応マシン: Java2 実行環境のインストールされているマシン
- メモリ: 96MB 以上 (128MB 以上推奨)

H.4 注意事項

- 再生可能な RMV ファイルの縦横のピクセルサイズは、最大 512 × 512 ピクセルである。
- 同時に開くことのできる RMV ファイルは 8 個までである。

さらに詳細な説明については、付属のオンラインマニュアルを参考にされたい。

索引

This is a blank page.

国際単位系 (SI) と換算表

表1　SI基本単位および補助単位

量	名称	記号
長さ	メートル	m
質量	キログラム	kg
時間	秒	s
電流	アンペア	A
熱力学温度	ケルビン	K
物質量	モル	mol
光度	カンデラ	cd
平面角	ラジアン	rad
立体角	ステラジアン	sr

表2　SIと併用される単位

名称	記号
分，時，日	min，h，d
度，分，秒	°，′，″
リットル	l，L
トン	t
電子ボルト	eV
原子質量単位	u

1 eV=1.60218×10^{-19} J

1 u=1.66054×10^{-27} kg

表5　SI接頭語

倍数	接頭語	記号
10^{18}	エクサ	E
10^{15}	ペタ	P
10^{12}	テラ	T
10^{9}	ギガ	G
10^{6}	メガ	M
10^{3}	キロ	k
10^{2}	ヘクト	h
10^{1}	デカ	da
10^{-1}	デシ	d
10^{-2}	センチ	c
10^{-3}	ミリ	m
10^{-6}	マイクロ	μ
10^{-9}	ナノ	n
10^{-12}	ピコ	p
10^{-15}	フェムト	f
10^{-18}	アト	a

表3　固有の名称をもつSI組立単位

量	名称	記号	他のSI単位による表現
周波数	ヘルツ	Hz	s^{-1}
力	ニュートン	N	m·kg/s^2
圧力，応力	パスカル	Pa	N/m^2
エネルギー，仕事，熱量	ジュール	J	N·m
工率，放射束	ワット	W	J/s
電気量，電荷	クーロン	C	A·s
電位，電圧，起電力	ボルト	V	W/A
静電容量	ファラド	F	C/V
電気抵抗	オーム	Ω	V/A
コンダクタンス	ジーメンス	S	A/V
磁束	ウェーバ	Wb	V·s
磁束密度	テスラ	T	Wb/m^2
インダクタンス	ヘンリー	H	Wb/A
セルシウス温度	セルシウス度	℃	
光束	ルーメン	lm	cd·sr
照度	ルクス	lx	lm/m^2
放射能	ベクレル	Bq	s^{-1}
吸収線量	グレイ	Gy	J/kg
線量当量	シーベルト	Sv	J/kg

表4　SIと共に暫定的に維持される単位

名称	記号
オングストローム	Å
バーン	b
バール	bar
ガル	Gal
キュリー	Ci
レントゲン	R
ラド	rad
レム	rem

1 Å= 0.1 nm=10^{-10} m

1 b=100 fm^2=10^{-28} m^2

1 bar=0.1 MPa=10^5 Pa

1 Gal=1 cm/s^2=10^{-2} m/s^2

1 Ci=3.7×10^{10} Bq

1 R=2.58×10^{-4} C/kg

1 rad=1 cGy=10^{-2} Gy

1 rem=1 cSv=10^{-2} Sv

（注）

1. 表1－5は「国際単位系」第5版，国際度量衡局 1985年刊行による。ただし，1 eVおよび1 uの値はCODATAの1986年推奨値によった。
2. 表4には海里，ノット，アール，ヘクタールも含まれているが日常の単位なのでここでは省略した。
3. barは，JISでは流体の圧力を表わす場合に限り表2のカテゴリーに分類されている。
4. EC閣僚理事会指令ではbar，barnおよび「血圧の単位」mmHgを表2のカテゴリーに入れている。

換　算　表

力	N(=10^5dyn)	kgf	lbf
	1	0.101972	0.224809
	9.80665	1	2.20462
	4.44822	0.453592	1

粘度　1 Pa·s(N·s/m^2)=10 P(ポアズ)(g/(cm·s))

動粘度　1 m^2/s=10^4St(ストークス)(cm^2/s)

圧力	MPa(=10 bar)	kgf/cm^2	atm	mmHg(Torr)	lbf/in^2(psi)
	1	10.1972	9.86923	7.50062×10^3	145.038
	0.0980665	1	0.967841	735.559	14.2233
	0.101325	1.03323	1	760	14.6959
	1.33322×10^{-4}	1.35951×10^{-3}	1.31579×10^{-3}	1	1.93368×10^{-2}
	6.89476×10^{-3}	7.03070×10^{-2}	6.80460×10^{-2}	51.7149	1

エネルギー・仕事・熱量	J(=10^7erg)	kgf·m	kW·h	cal(計量法)	Btu	ft·lbf	eV
	1	0.101972	2.77778×10^{-7}	0.238889	9.47813×10^{-4}	0.737562	6.24150×10^{18}
	9.80665	1	2.72407×10^{-6}	2.34270	9.29487×10^{-3}	7.23301	6.12082×10^{19}
	3.6×10^6	3.67098×10^5	1	8.59999×10^5	3412.13	2.65522×10^6	2.24694×10^{25}
	4.18605	0.426858	1.16279×10^{-6}	1	3.96759×10^{-3}	3.08747	2.61272×10^{19}
	1055.06	107.586	2.93072×10^{-4}	252.042	1	778.172	6.58515×10^{21}
	1.35582	0.138255	3.76616×10^{-7}	0.323890	1.28506×10^{-3}	1	8.46233×10^{18}
	1.60218×10^{-19}	1.63377×10^{-20}	4.45050×10^{-26}	3.82743×10^{-20}	1.51857×10^{-22}	1.18171×10^{-19}	1

1 cal = 4.18605 J (計量法)

= 4.184 J　(熱化学)

= 4.1855 J　(15 °C)

= 4.1868 J (国際蒸気表)

仕事率　1 PS (仏馬力)

= 75 kgf·m/s

= 735.499 W

放射能	Bq	Ci
	1	2.70270×10^{-11}
	3.7×10^{10}	1

吸収線量	Gy	rad
	1	100
	0.01	1

照射線量	C/kg	R
	1	3876
	2.58×10^{-4}	1

線量当量	Sv	rem
	1	100
	0.01	1

(86年12月26日現在)

地球シミュレータ用可視化システム利用手引書