計算科学技術活用型特定研究開発推進事業

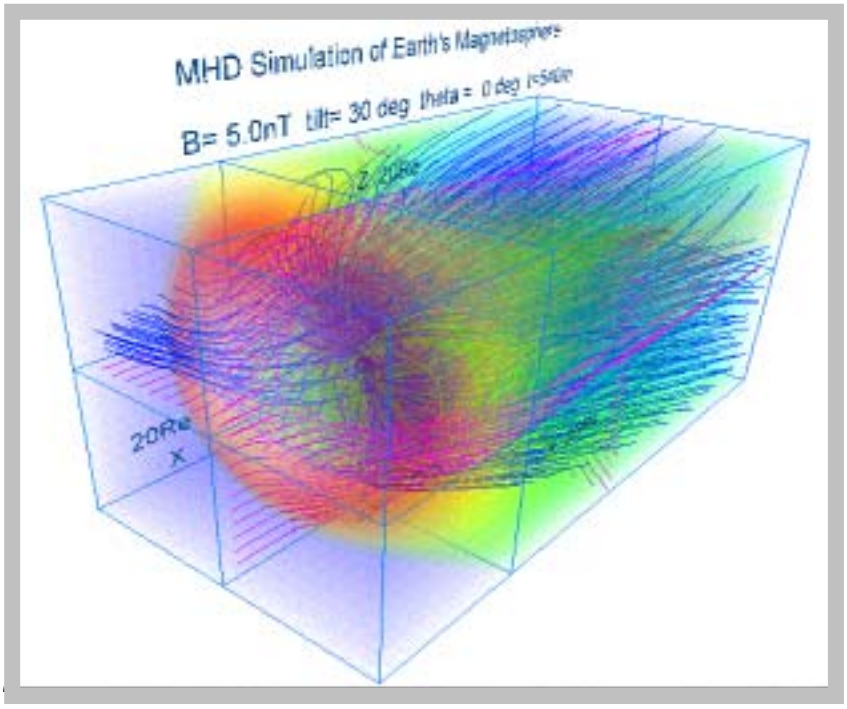
「宇宙シミュレーション・ネットラボラトリーシステムの開発」

科学研究費基盤研究(A)(1)

「共通並列計算電磁流体・粒子コードによる太陽風磁気圏電離圏ダイナミックスの研究」

VRML の利用方法

（Fortran とＣを用いて）



平成１６年　３月

名古屋大学太陽地球環境研究所

荻野竜樹

計算科学技術活用型特定研究開発推進事業

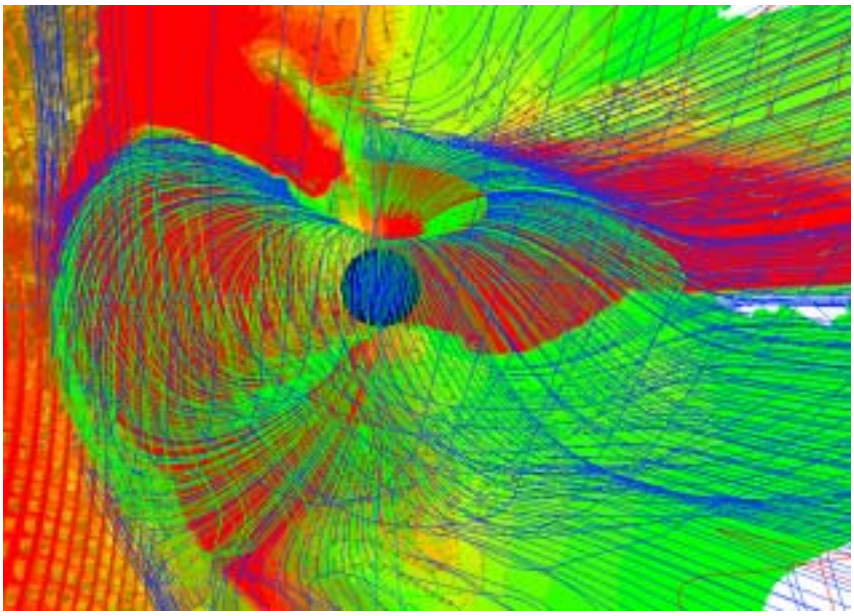
「宇宙シミュレーション・ネットラボラトリーシステムの開発」

科学研究費基盤研究(A)(1)

「共通並列計算電磁流体・粒子コードによる太陽風磁気圏電離圏ダイナミックスの研究」

VRML の利用方法

（Fortran とＣを用いて）



平成１６年　３月

名古屋大学太陽地球環境研究所

荻野竜樹

目次

目的

地球磁気圏などの３次元シミュレーション結果をよりよく理解するためには、３次元可視化は不可欠である。ここに、インターネット３次元可視化のための国際標準言語 VRML（Virtual Reality Modeling Language）の登場のよって、３次元画像処理専用機と３次元画像処理専用ソフトウエアを持たなくても、VRML ファイルと VRML のビューアさえあれば誰でも３次元画像を自分の好きなように見ることができるようになった。しかし、VRML ファイルを作成する便利なツールがないためと必要で複雑な VRML ファイルを作成するのが容易でないために、VRML の利用と普及は遅々として進まないのが世の中の現状である。

私達は磁気圏の３次元シミュレーション結果を VRML によって３次元可視化する方法に早くから着目し、画像処理に長年培ってきた経験と知識を活かして Fortran や C 言語を用いて VRML ファイルを作成することに取り組んできた。その延長として、計算科学プロジェクトで、VRML による可視化ツールの開発と標準化を更に進めて、プロジェクト参加者に Fortran と C 言語による VRML コンテンツ作成用のインターフェースサブルーチンパッケージを呈示・提供して、シミュレーションの３次元可視化に利用してもらうことを目的として、この「VRML の利用方法」のマニュアルを作成することにした。

内容

太陽風と地球磁気圏相互作用の３次元グローバル電磁流体力学的（MHD）シミュレーションで何が得られているかを知るためには３次元可視化は必要不可欠である。例えば、３次元の磁力線構造などを理解するために、座標軸を回転させて動画を作ることもよく使う方法である。市販の AVS などの３次元画像解析ツールなどは、大変便利で有意義なものであるが、３次元画像表示はあまりにも多様性に富んでいるので、本当に描きたい図を描こうとする場合、どうしても物足りない部分が出てくる。こうした場合、画像処理の基本プログラムを組むことになる。私達はこれを３次元画像解析専用機 TITAN や Indigo-2、及び３次元画像処理専用ソフトウエア Dore、AVS、Open-GL を用いることによって実施してきた。３次元空間で磁力線を描き、専用機のＺバッファなどの３次元画像処理機能を用いて、対象物を即座に回転したり、拡大縮小することにより、見易い視点を選んで３次元構造の理解に役立ててきた。

しかし、VRML（Virtual Reality Modeling Language）の登場のよって、３次元画像処理専用機と３次元画像処理専用ソフトウエアを持たなくても、誰でも VRML のビューアさえあれば３次元画像を自分の好きなように見ることができる状況が実現した。自分のコンピュータの能力に依存して３次元画像処理（回転、拡大縮小など）の速度は決まるが、最近のネットスケープやインターネットエクスプローラなどのブラウザを使えば、VRML2.0 対応の Cosmo Player 等のビューアが標準で付いている。パーソナルコンピュータも最近高速になってきたので、高速の cpu と グラフィックアクセラレータを積み、更に十分なメモリ（128 MB 以上）を載せれば、SGI 製の Indigo-2 などに劣らない性能を発揮できる。また、精度の高い３次元画像を快適に見たいのであれば Webspace や SGI の Cosmo Worlds の利用が更に有効である。

VRML ファイルの作成をどう実現するかであるが、私達は、VRML ファイル作成のための Fortran Interface Subroutine Package を準備し、フォートランプログラムを用いて、３次元シミュレーションデータから直接に VRML ファイル（*.wrl）を作っている。これは３次元と２次元の違いはあるが、PostScript 画像ファイルを作成する方法と同様の方法である。VRML のビューアには通常視点を移動する walk モードと対象物を移動・回転・ 拡大縮小する examine モードがあり、磁気圏の３次元構造をより詳しく調べることができる。現在の重要な問題点は、地球磁気圏の３次元磁力線描画の VRML ファイルが ascii ファイルを用いているために数ＭＢと非常に大きくなること である。これらの問題も圧縮 VRML ファイルを標準に用いるとか、VRML バイナリーファイルを用いることによって、かなり改善されることが期待される。太陽風やＩＭＦの変化に伴う地球磁気圏の時間変化の３次元動画を VRML で表示するのは今後の最も興味ある課題である。

この「VRML の利用方法」では、先ず、第１、２章に Fortran プログラムを用いて、VRML2.0 形式の３次元画像ファイルを作成する基本的な例題とシミュレーション結果の解析にそのまま利用できる例題を示し、更に、第３章に太陽風と地球磁気圏の MHD シミュレーション結果に適用した場合の具体例を示す。第４章では、地球磁気圏の３次元グローバルＭＨＤシミュレーションコード及び PostScript と VRML による画像解析をどのように実行するかを示した一連の具体例が載せてある。最後の第５、６章では、種々の地球磁気圏の３次元ＭＨＤモデルに対する VRML を用いた可視化方法の具体例が示してある。また、本マニュアルに示してある main Fortran program と subroutine package は全て CD-ROM にソースプログラムとして収められている。計算科学プロジェクト参加者の何らかのお役に立てば幸いである。

１．基本的な例題（１）

ここでは Fortran プログラムを用いて、VRML2.0 型式の画像ファイルを作成する基本的な例題を示す。具体的には、各メインプログラムとサブルーチンパッケージ：pk1subrtn.f をコンパイル・リンク・実行して VRML2.0 型式の画像ファイルを作成する。

使用方法：

1. f77 -c -O pk1subrtn.f

   サブルーチンパッケージ（pk1subrtn.f）をコンパイルしてオブジェクトモジュール（pk1subrtn.o）を作成する。

2. f77 -O box1.f pk1subrtn.o

   メインプログラム（box1.f etc.）とオブジェクトモジュール（pk1subrtn.o）をリンクして実行ファイル（a.out）を作成する。

3. a.out

   実行ファイル（a.out）を実行して VRML 2.0 ファイル（fort.10）を作成する。

4. mv fort.10 box1.wrl

   作成した VRML のファイル名を “ ***.wrl “ に変更する。

１－１．立方体

main program　box1.f
subroutine　box1 (r, g, b, sz)

説明：色の値と一辺の長さを与えて、立方体を描くプログラム

入力変数：

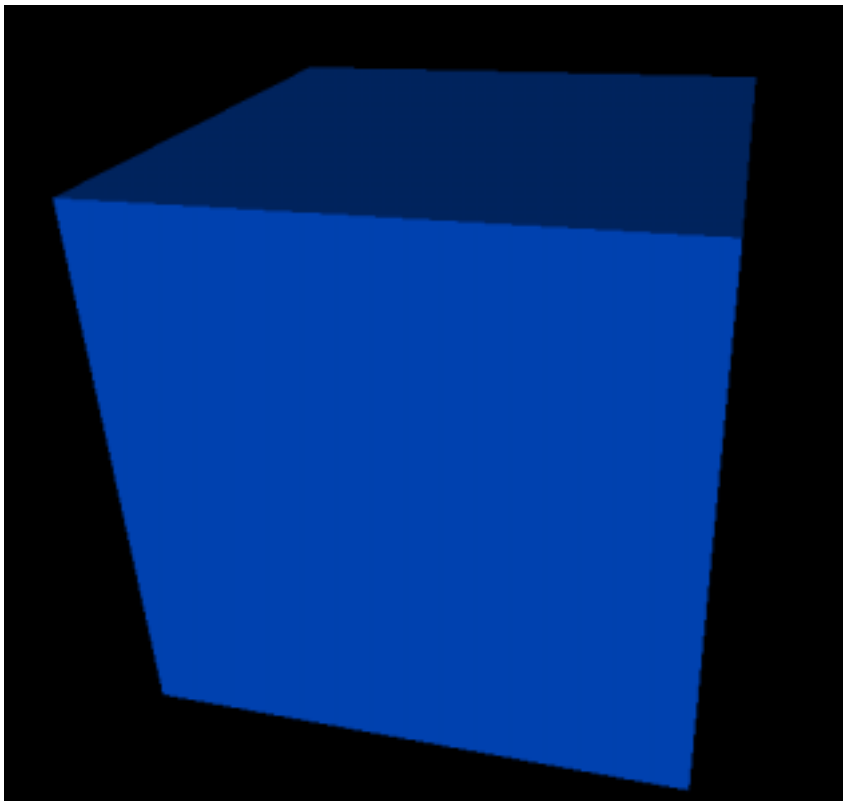
r：赤色の値 [0,1]

g：緑色の値 [0,1]

b：青色の値 [0,1]

sz：立方体の一辺の長さ

描画例：box1.wrl



１－２．直方体

main program　box2.f
subroutine　box2 (r, g, b, sx, sy, sz)

説明：色の値と三辺の長さを与えて、直方体を描くプログラム

入力変数：

r：赤色の値 [0,1]

g：緑色の値 [0,1]

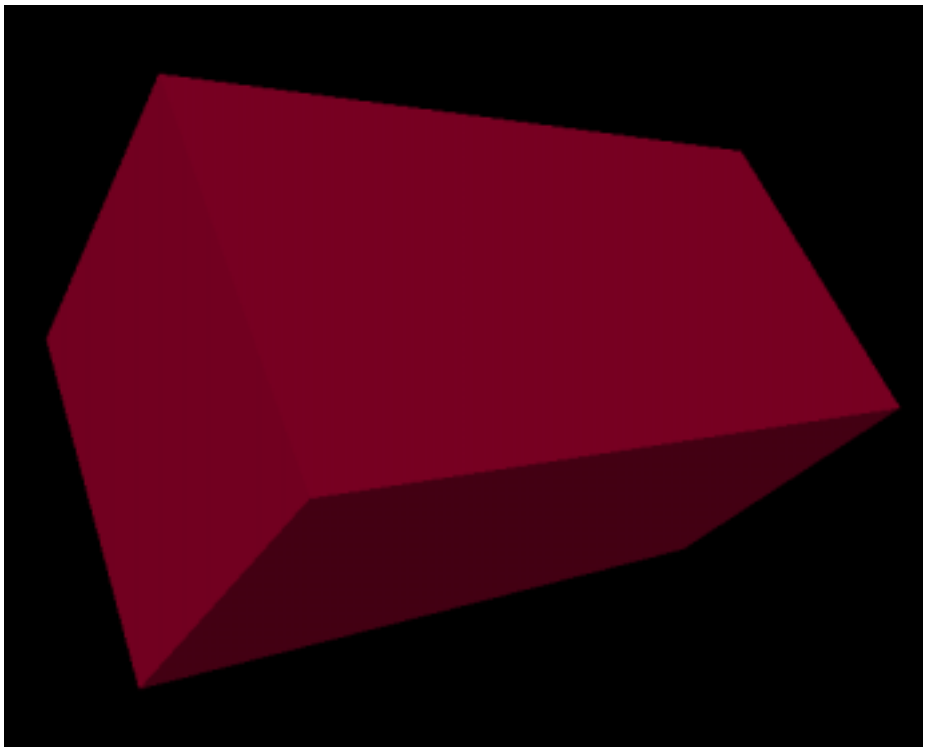
b：青色の値 [0,1]

sx：x 方向の一辺の長さ

sy：y 方向の一辺の長さ

sz：z 方向の一辺の長さ

描画例：box2.wrl



１－３．三角錐

main program　　cone1.f
subroutine　　cone1k1 (rd, h, r, g, b, x, y, z)

説明：色の値と底面の半径、高さを与え、三角錐を描くプログラム

入力変数：

rd：底面の半径

h：高さ

r：赤色の値 [0,1]

g：緑色の値 [0,1]

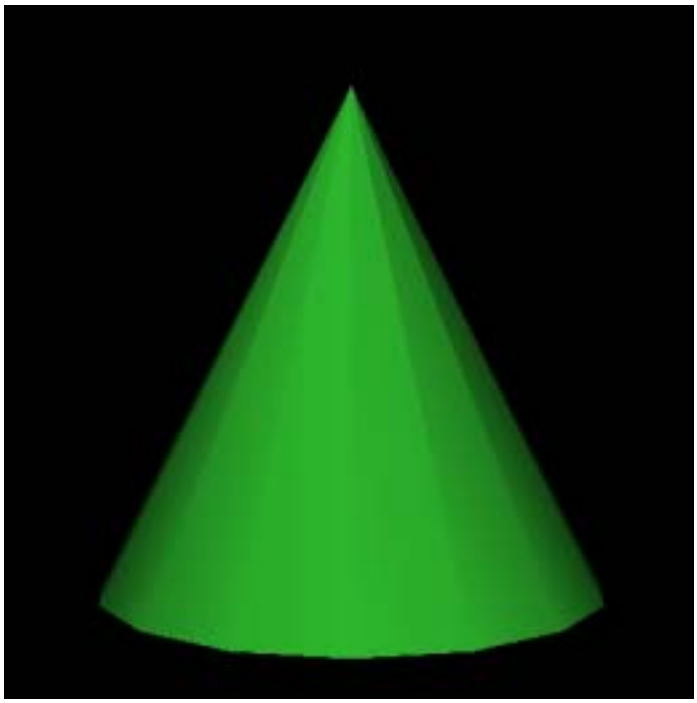
b：青色の値 [0,1]

x：x 方向の平行移動量

y：y 方向の平行移動量

z：z 方向の平行移動量

描画例：cone1.wrl



１－４．円柱

main program　　cylinder1.f
subroutine　　cylinder1 (rd, h, r, g, b)

説明：色の値と底面の半径、高さを与え、円柱を描くプログラム

入力変数：

rd：底面の半径

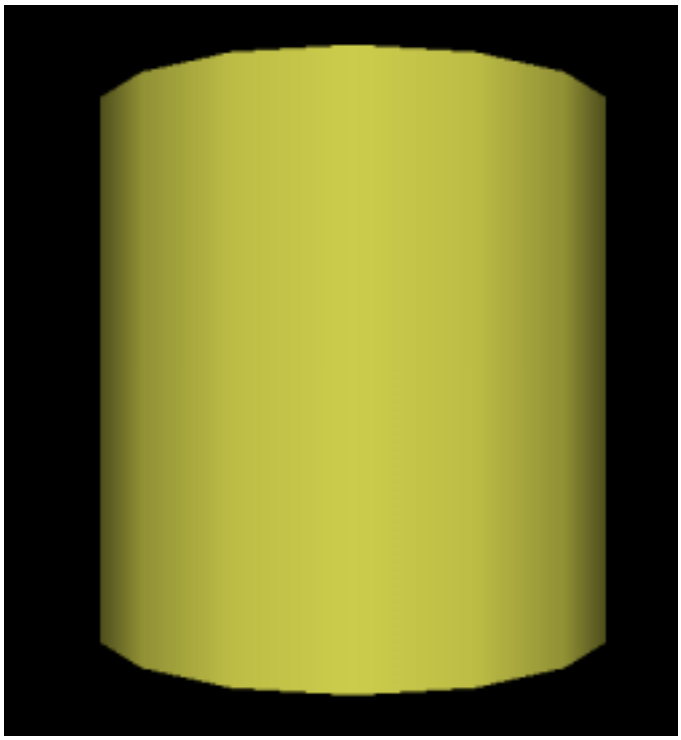
h：高さ

r：赤色の値 [0,1]

g：緑色の値 [0,1]

b：青色の値 [0,1]

描画例：cylinder1.wrl



１－５．球

main program　　sphere1.f
subroutine　　shere2 (r, g, b, rd)

説明：色の値と、半径を与え、球を描くプログラム

入力変数：

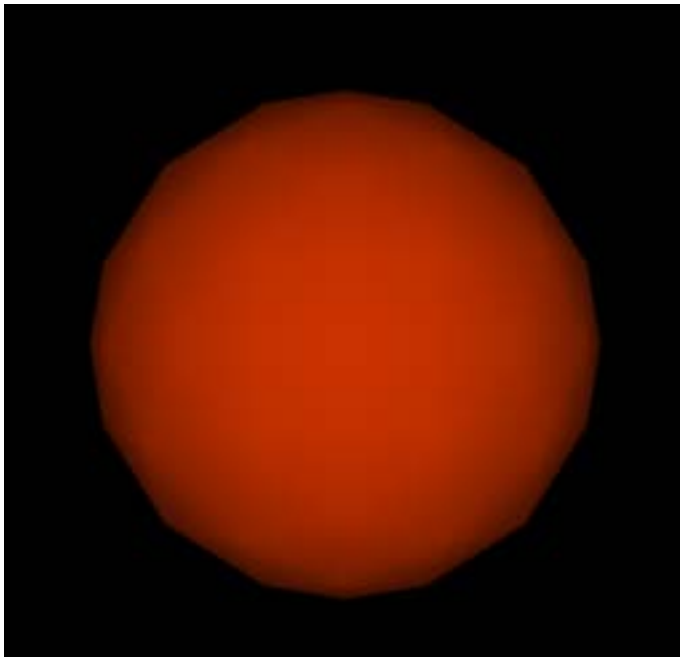
rd：半径

r：赤色の値 [0,1]

g：緑色の値 [0,1]

b：青色の値 [0,1]

描画例：sphere1.wrl



１－６．背景色

main program　　backgrd4.f
subroutine　　backd (n1, n2, rd),

説明：背景色を付けるプログラム

入力変数：

n1：地面の色を変化させる段階数

n2：空の色を変化させる段階数

rd：１段階あたりの角度(ラジアン)

描画例：backgrd4.wrl

１－７．文字列

main program　text1.f
subroutine　text1 (x, y, z, n, r, g, b, sz, ch1)

説明：文字列、フォントの色、フォントサイズを与え、文字列を描くプログラム

入力変数：

x：x 方向の平行移動量

y：y 方向の平行移動量

z：z 方向の平行移動量

n：文字列の数(10 個まで)

r：赤色の値 [0,1]

g：緑色の値 [0,1]

b：青色の値 [0,1]

sz：フォントサイズ

ch1：入力文字列 [40 文字以内]

描画例：text1.wrl

１－８．折れ線：単色

main program　　lineset1.f
subroutine　　line1 (ln, po, r, g, b)

説明：線分の両端の座標を与え、色を指定し、座標を線分で順に結んで一筆書きで描くプログラム

入力変数：

ln：データ数

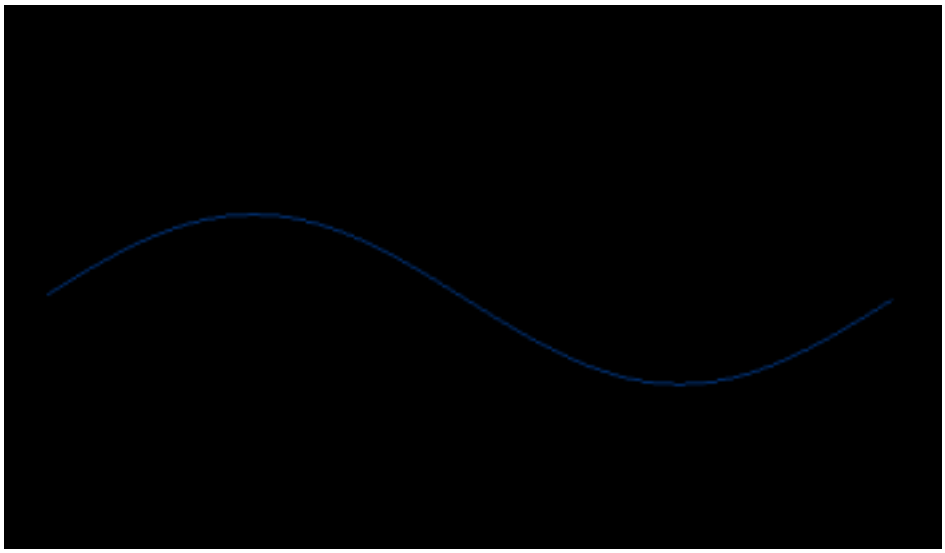
po(3*ln)：点の(x, y, z) 座標を順番に与える

r：赤色の値 [0,1]

g：緑色の値 [0,1]

b：青色の値 [0,1]

描画例：lineset1.wrl



１－９．折れ線：グラデーション

main program　　lineset2.f
subroutine　　line2 (ln, po, sco, eco)

説明：線分の両端の座標を与え、色を指定(グラデーション)し、座標を折れ線で結んで描くプログラム

入力変数：

ln：データ数

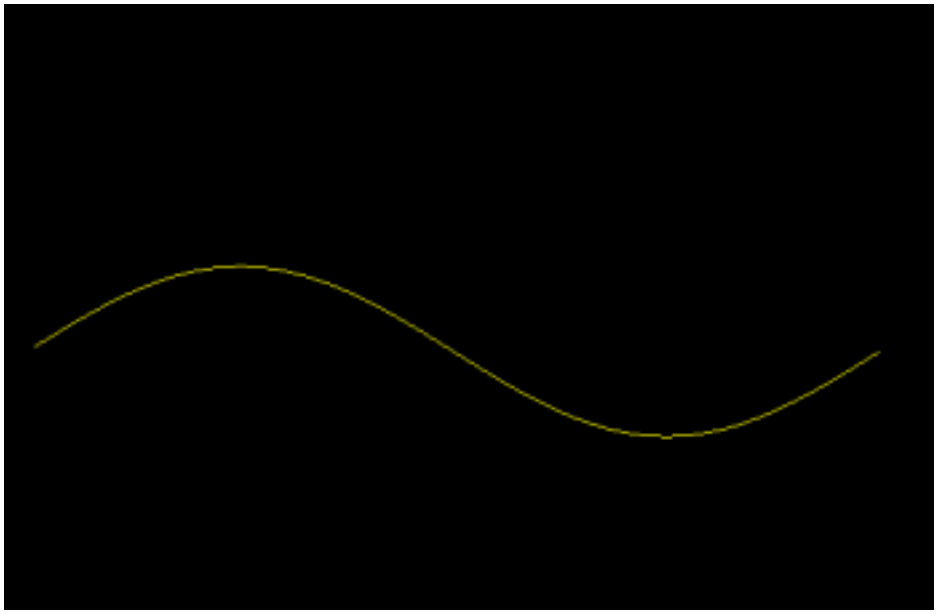
po(3*ln)：点の(x, y, z) 座標を順番に与える

sco：グラデーションの始まりの色(カラーコード)

eco：グラデーションの終わりの色(カラーコード)

カラーコード(1: white 2: yellow 3: mazenda 4: red 5: sian 6: green 7: blue)

描画例：lineset2.wrl



１－１０．点線：単色

main program　　pointset1.f
subroutine　　point1 (ln, po, r, g, b)

説明：点の座標を与え、点線を描くプログラム

入力変数：

ln：データ数

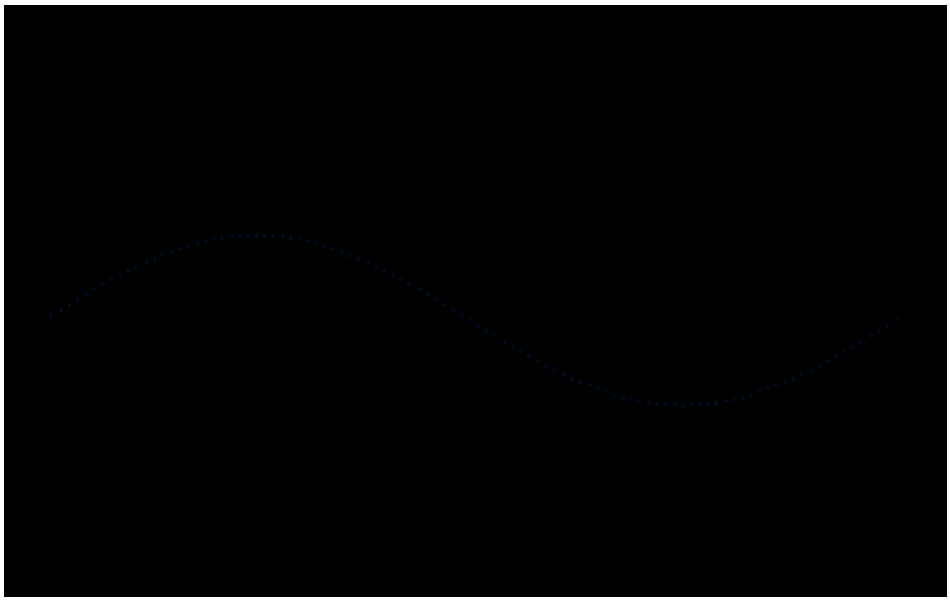
po(3*ln)：点の (x, y, z) 座標を順番に与える

r：赤色の値 [0,1]

g：緑色の値 [0,1]

b：青色の値 [0,1]

描画例：pointset1.wrl



１－１１．点線：グラデーション

main program　　pointset2.f
subroutine　　point2 (ln, po, sco, eco)

説明：点の座標を与え、色を指定(グラデーション)し、点線を描くプログラム

入力変数：

ln：データ数

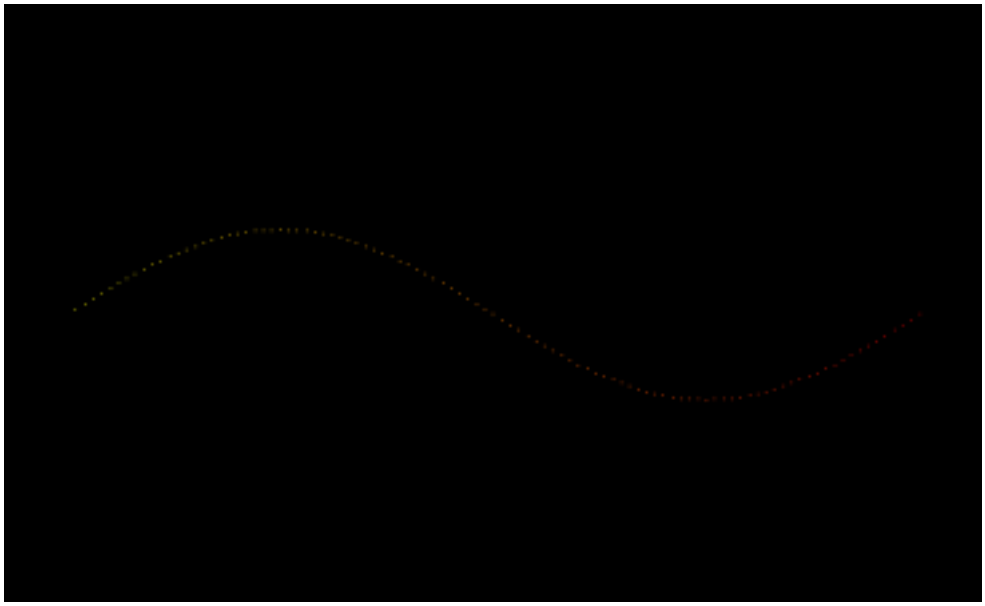
po(3*ln)：点の(x, y, z) 座標を順番に与える

sco：グラデーションの始まりの色(カラーコード)

eco：グラデーションの終わりの色(カラーコード)

カラーコード(1: white 2: yellow 3: mazenda 4: red 5: sian 6: green 7: blue)

描画例：pointset2.wrl



１－１２．太い矢印

main program　arrow2.f
subroutine　arrow2 (nx, ny, u, p, rd1, rd2, h, cx, cy, cz, br, bg, bb, cr, cg, cb)

説明：円筒(グラデーション色)と三角錐を組み合わせて、太さのある矢印を描くプログラム

入力変数：

nx：x 方向のデータ数

ny：y 方向のデータ数

u：点の色(r, g, b)を順番に与える

p：点の(x, y, z) 座標を順番に与える

rd1：円筒の底面の半径

rd2：三角錐の底面の半径

h：三角錐の高さ

cx：三角錐の x 中心の座標

cy：三角錐の y 中心の座標

cz：三角錐の z 中心の座標

br：円筒底面の赤色の値 [0,1]

bg：円筒底面の緑色の値 [0,1]

bb：円筒底面の青色の値 [0,1]

cr：三角錘の赤色の値 [0,1]

cg：三角錘の緑色の値 [0,1]

cb：三角錘の青色の値 [0,1]

描画例：arrow2.wrl

１－１３．太さのある曲線

main program　　linemesh1.f
subroutine　　linemesh (p, n1, rd, r, g, b)

説明：円筒をつないで太さのある線を描くプログラム

入力変数：

p：点の(x, y, z) 座標を順番に与える

n1：データ数

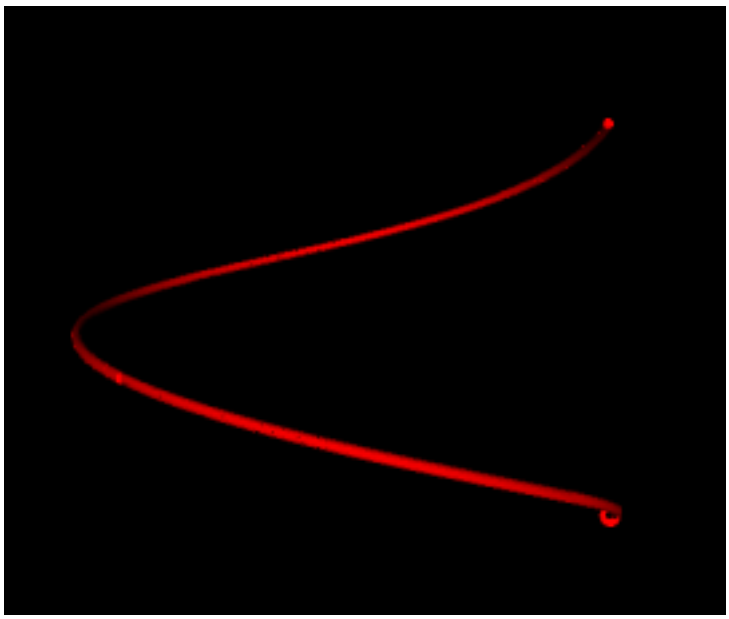
rd：円筒の半径

r：円筒の赤色の値 [0,1]

g：円筒の緑色の値 [0,1]

b：円筒の青色の値 [0,1]

描画例：linemesh1.wrl



１－１４．面（1）：単色三角メッシュ

main program　　mesh1.f
subroutine　　face1 (ln, po, r, g, b)

説明：点の座標を与え、三角メッシュで面を描くプログラム

入力変数：

ln：データ数

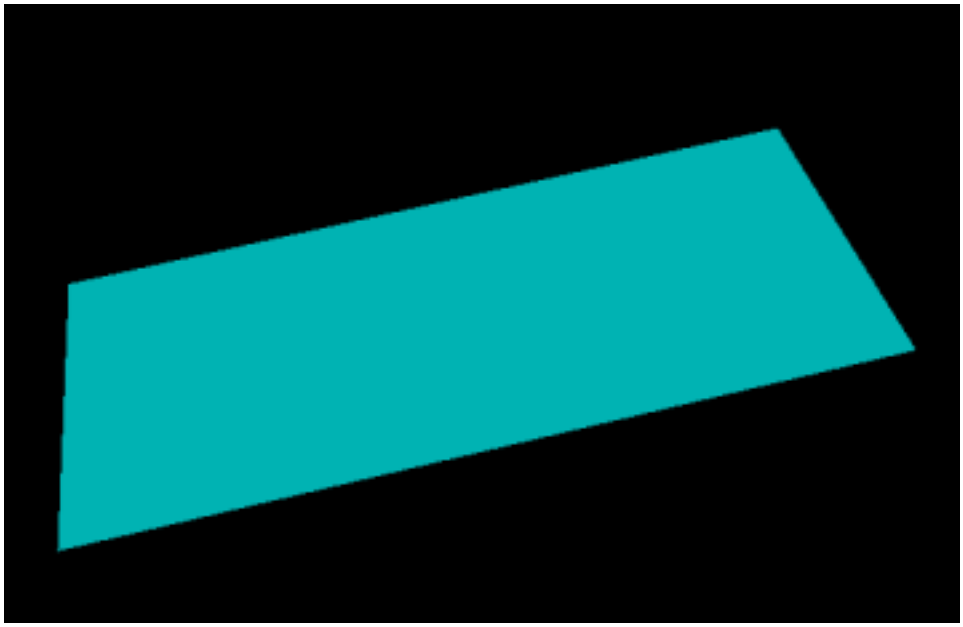
po：点の(x, y, z) 座標を順番に与える

r：面の赤色の値 [0,1]

g：面の緑色の値 [0,1]

b：面の青色の値 [0,1]

描画例：mesh1.wrl



１－１５．面（２）：三角ベルト

main program　　triangb1.f
subroutine　　triangbk1 (nx, ny, u, p)

説明：点データの座標と色を与え、三角ベルトで面を描くプログラム

入力変数：

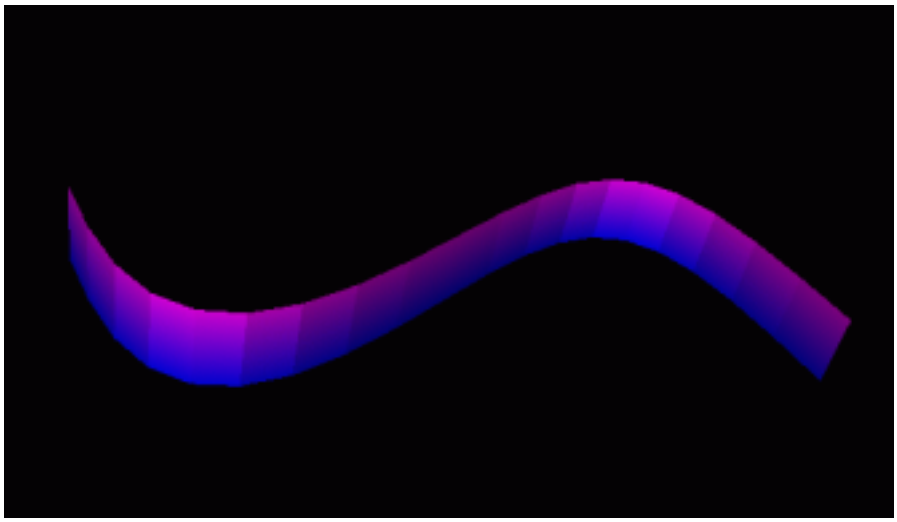
nx：x 方向のデータ数

ny：y 方向のデータ数

u：点の色(r, g, b)を順番に与える

p：点の(x, y, z) 座標を順番に与える

描画例：triangb1.wrl



１－１６．面（３）：多色三角メッシュ

main program　　triangm1.f
subroutine　　triangm (nx, ny, u, p)

説明：点データの座標と色を与え、三角メッシュで面を描くプログラム

入力変数：

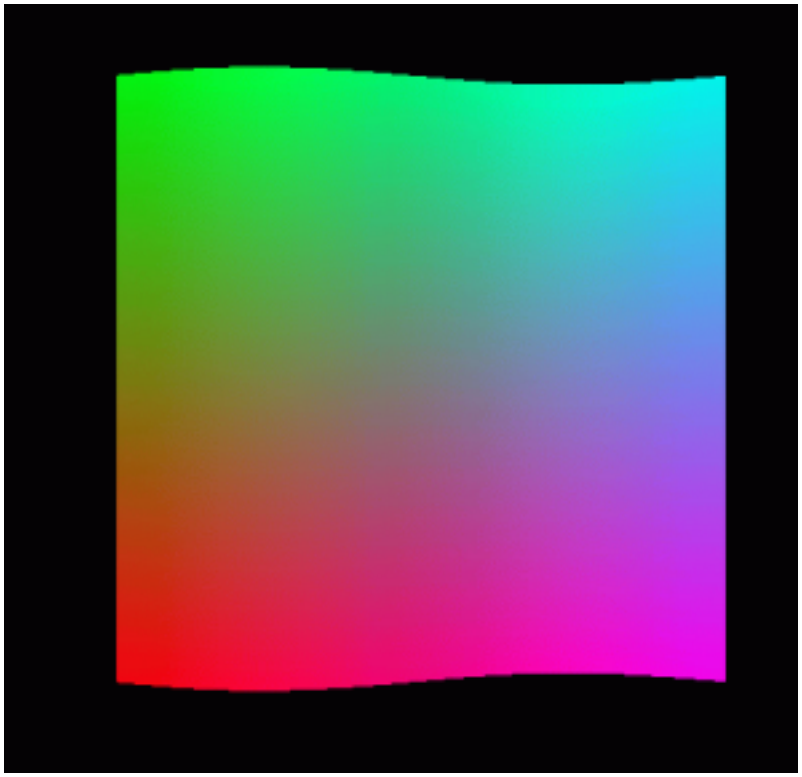
nx：x 方向のデータ数

ny：y 方向のデータ数

u：点の色(r, g, b)を順番に与える

p：点の(x, y, z) 座標を順番に与える

描画例：triangm1.wrl



１－１７．立体：三角メッシュで構成

main program　defusem1.f
subroutine　defusem (nx, ny, u, p, x1, y1, z1, x2, y2, z2, r, n)

説明：点のデータから三角メッシュで作成した面を、少しずつ移動させながら描画して、立体にするプログラム

入力変数：

nx：x 方向のデータ数

ny：y 方向のデータ数

u：点の色(r, g, b)を順番に与える

p：点の(x, y, z) 座標を順番に与える

x1：x 方向の平行移動量

y1：y 方向の平行移動量

z1：z 方向の平行移動量

x2：回転軸の x 座標

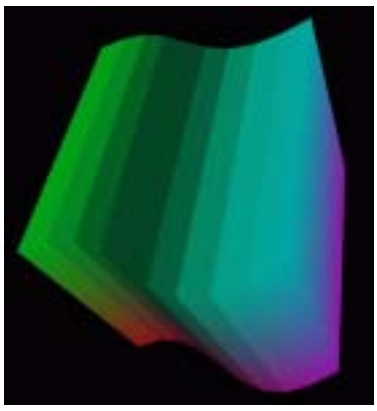
y2：回転軸の y 座標

z2：回転軸の z 座標

r：移動回転角(ラジアン)

n：処理回数

描画例：defusem1.wrl



１－１８．テクスチャ画像の貼りつけ

main program　　image1.f
subroutine　　imagetx1 (ln, po, ch1, n)

説明：三角メッシュで描いた面に、指定したテクスチャ画像を貼りつけるプログラム

入力変数：

ln：データ数

po：点の(x, y, z) 座標を順番に与える

ch1：テクスチャ画像の URL

n：URL の文字数

描画例：image1.wrl

１－１９．複雑な曲面：三角メッシュ

main program　　ebmesh3d.f
subroutine　　triangm1 (nx, ny, u, p)

説明：面を構成する点のデータを読み込み、三角メッシュで描画させるプログラム

入力変数：

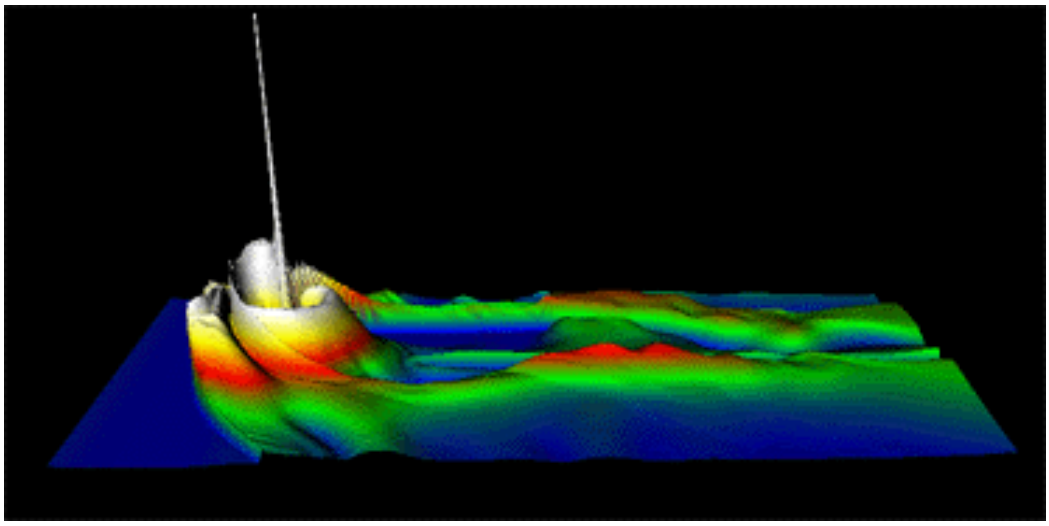
nx：x 方向のデータ数

ny：y 方向のデータ数

u：点の色(r, g, b)を順番に与える

p：点の(x, y, z) 座標を順番に与える

描画例：ebmesh3d.wrl



２．基本的な例題（２）

ここではFortranプログラムを用いて、VRML 2.0形式の画像ファイルを作成する基本的でかつシミュレーションデータの解析応用にそのまま利用できる例題を示す。具体的には、各メインプログラムとサブルーチンパッケージ：zvrsuba.fをコンパイル・リンク・実行してVRML 2.0形式の画像ファイルを作成する。

使用方法：

1. f77 -c -O zvrsuba.f

   サブルーチンパッケージ（zvrsuba.f）をコンパイルしてオブジェクトモジュール（zvrsuba.o）を作成する。

2. f77 -O msymbol.f zvrsuba.o

   メインプログラム（msymbol.f etc.）とオブジェクトモジュール（zvrsuba.o）をリンクして実行ファイル（a.out）を作成する。

3. a.out

   実行ファイル（a.out）を実行してVRML 2.0 ファイル（fort.10）を作成する。

4. mv fort.10 msymbol.wrl

   作成したVRMLのファイル名を“ ***.wrl “ に変更する。

２－１．文字

main program　msymbol.f

subroutine　symblv (x, y, z, h, chr, n)　…白

symblvc (x, y, z, h, r, g, b, chr, n)　…カラー

説明：入力する文字と原点、幅と高さ（、色）を与えて文字を描くプログラム

入力変数：

x：原点の x 座標

y：原点の y 座標

z：原点の z 座標

h：文字の高さ

r：赤色の値 [0,1]

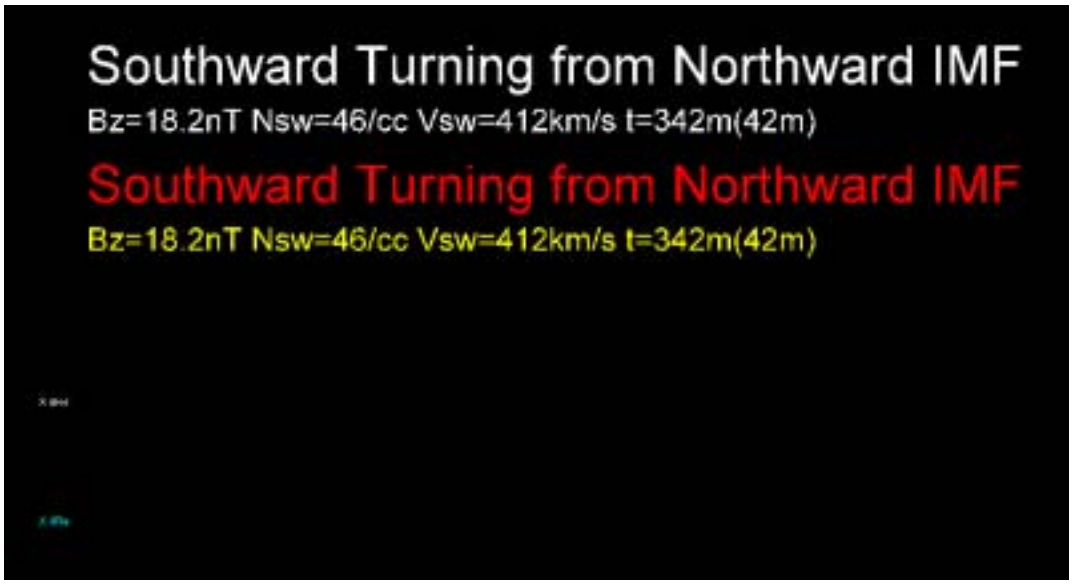
g：緑色の値 [0,1]

b：青色の値 [0,1]

chr：入力文字 [80 文字以内]

n：文字の数

描画例：msymbol.wrl



２－２．点

main program　mpoint.f
subroutine　point3d (nx, ny, nz, x0, y0, z0, xl, yl, zl, ico, u)

説明：nx*ny*nz 個の３次元の点の位置とカラーコードを与えて点を描くプログラム

入力変数：

nx：x 方向のデータ数

ny：y 方向のデータ数

nz：z 方向のデータ数

x0：原点の x 座標

y0：原点の y 座標

z0：原点の z 座標

x1：x 方向の一辺の長さ

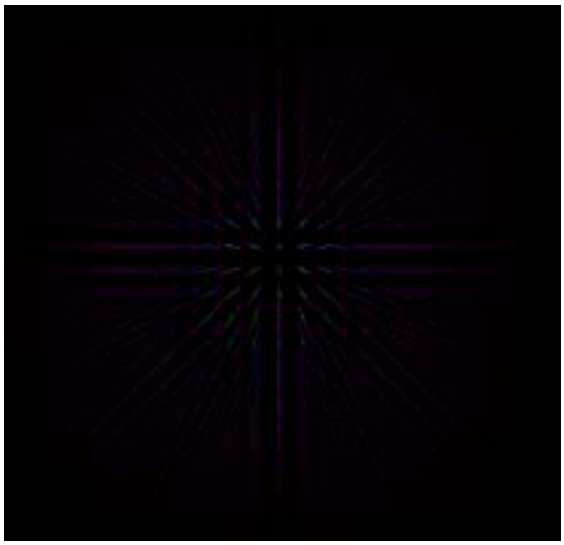
y1：y 方向の一辺の長さ

z1：z 方向の一辺の長さ

ico：カラーコード [1,7]

u (nx*ny*nz)：入力データ、多点の座標 (x, y, z) を順番に与える

描画例：mpoint.wrl



２－３．線

main program　mline.f
subroutine　line3d (lasl, r, g, b, po)

説明：一筆書きで描ける点列の座標を与え、色を指定し、点列を折れ線で結んで描く

プログラム

入力変数：

lasl：点列の数

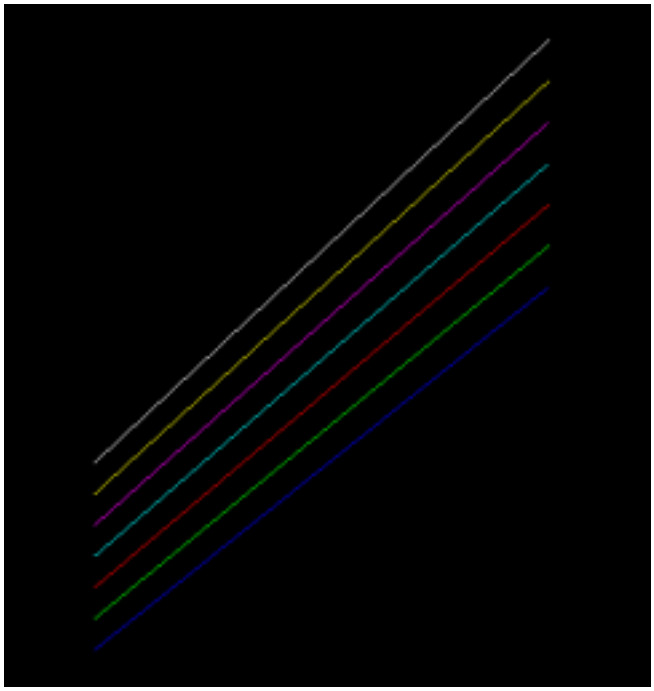
r：赤色の値 [0,1]

g：緑色の値 [0,1]

b：青色の値 [0,1]

po (3*lasl)：点列の (x, y, z) 座標を順番に与える

描画例：mline.wrl



２－４．三角メッシュ

main program　mtriangm.f
subroutine　triangm (nx, ny, u, p)

説明：原点の位置と一辺の長さ、データ数、カラーコードを与えて nx*ny 個の (x, y, z)

座標の２次元データ u(3*nx*ny)の画像を三角メッシュで描くプログラム

入力変数：

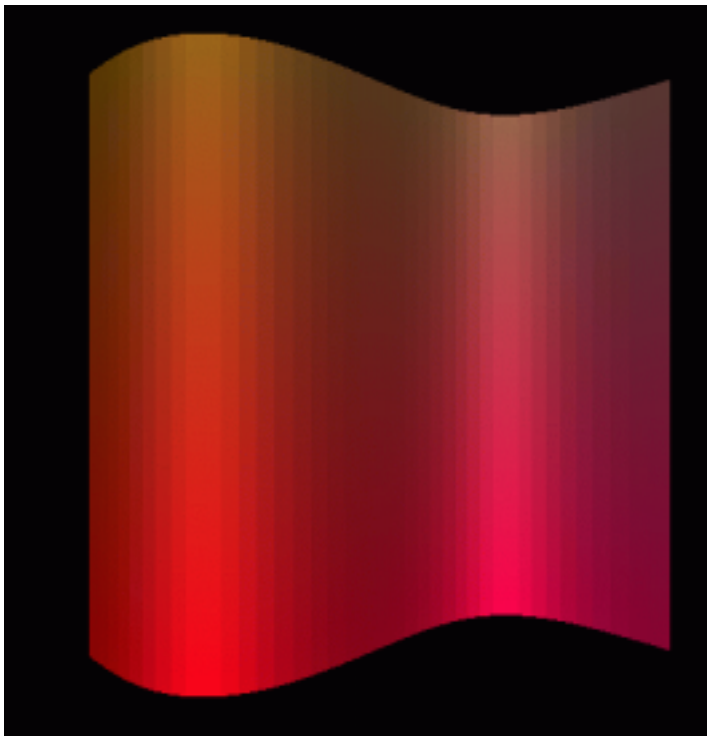
nx：x 方向のデータ数

ny：y 方向のデータ数

u (3*nx*ny)：入力データ

p (3*nx*ny)：計算に必要な補助データ

描画例：mtriangm.wrl



２－５．ピクセルイメージ

main program　mzpt03.f
subroutine　pixel2(nx, ny, xb, yb, x1, y1, ipx0, ico, icc, zcc, u)

説明：原点の位置と一辺の長さ、データ数、カラーコードを与えて２次元データ u (nx*ny) の画像をピクセルイメージで描くプログラム

入力変数：

nx：x 方向のデータ数

ny：y 方向のデータ数

xb：原点の x 座標

yb：原点の y 座標

xl：x 方向の一辺の長さ

yl：y 方向の一辺の長さ

ipx0：3＝rgb カラー、4＝rgb カラーと透明度が設定できる

ico：カラーコード [1,7]

icc：1=xy 平面, 2=xz 平面, 3=yz 平面に描画

zcc：選択した面の位置

u (nx*ny)：入力データ

描画例：mzpt03.wrl

２－６．枠とピクセルイメージと文字

main program　mpix015.f

説明：枠とピクセルイメージ、文字を組み合わせて描くプログラム

入力変数：x, y, z, h, ch1, n, icc, zcc, r, g, b, x1, y1, x2, y2, nx, ny, xb, yb, x1, y1, ipx0, ico, u

詳しくは各サブルーチン参照。

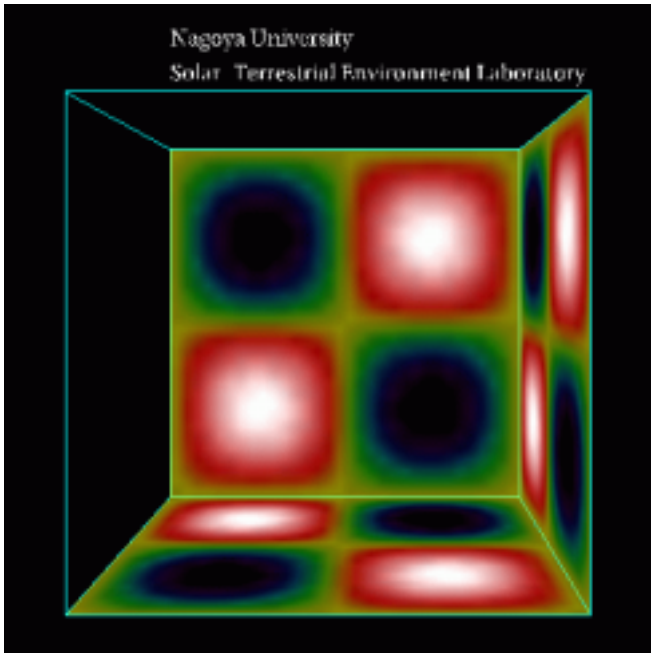
使用しているサブルーチン：

symblv (x, y, z, h, ch1, n)：

rect1 (icc, zcc, r, g, b, x1, y1, x2, y2)：

pixel2 (nx, ny, xb, yb, x1, y1, ipx0, ico, icc zcc, u)：

描画例：mpix015.wrl



２－７．等値面

main program　mcube301.f
subroutine　csuba (ic, aa, u) < (nx, ny, nz, x0, y0, z0, xl, yl, zl, nol, ico)

説明：３次元のデータ u (nx*ny*nz)を与えて等値面を描くプログラム

入力変数：

nx = ic (2)：x 方向のデータ数

ny = ic (3)：y 方向のデータ数

nz = ic (4)：z 方向のデータ数

x0 = aa (11)：原点の x 座標

y0 = aa (12)：原点の y 座標

z0 = aa (13)：原点の z 座標

xl = aa (8)：x 方向の一辺の長さ

yl = aa (9)：y 方向の一辺の長さ

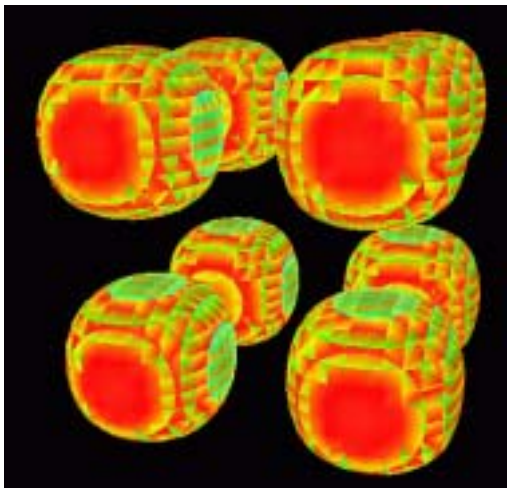
zl = aa (10)：z 方向の一辺の長さ

ico = ic (1)：カラーコード [1,7]

nol：直方体の辺の数；12

u (nx*ny*nz)：３次元の入力データ

描画例：mcube301.wrl



２－８．外部磁気圏の３次元格子

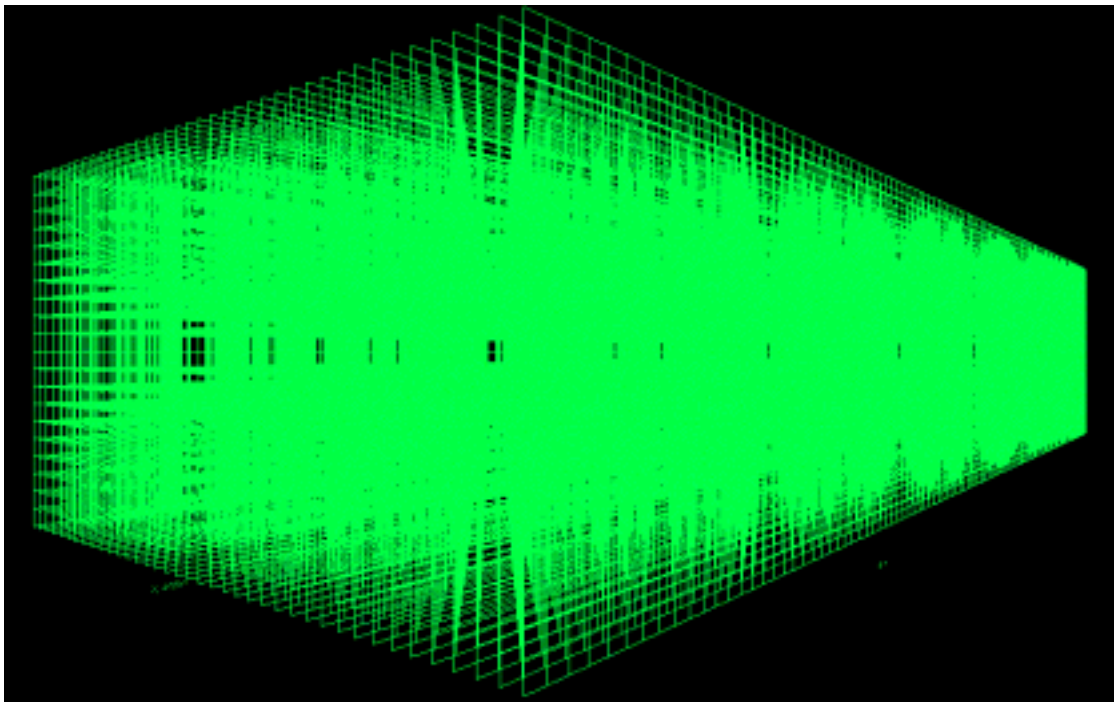
main program　outline3d.f
subroutine　lat3d1 (ic, aa) < (nx, ny, nz, xl, yl, zl, r, g, b)

説明：外部磁気圏を３次元格子で描くプログラム

入力変数：

nx = ic(1)：x 方向のデータ数
ny = ic(2)：y 方向のデータ数
nz = ic(3)：z 方向のデータ数
x1 = aa(1)：x 方向の一辺の長さ
y1 = aa(2)：y 方向の一辺の長さ
z1 = aa(3)：z 方向の一辺の長さ
r = aa(4)：赤色の値 [0,1]
g = aa(5)：緑色の値 [0,1]
b = aa(6)：青色の値 [0,1]

描画例：outline3d.wrl



２－９．内部磁気圏の３次元格子

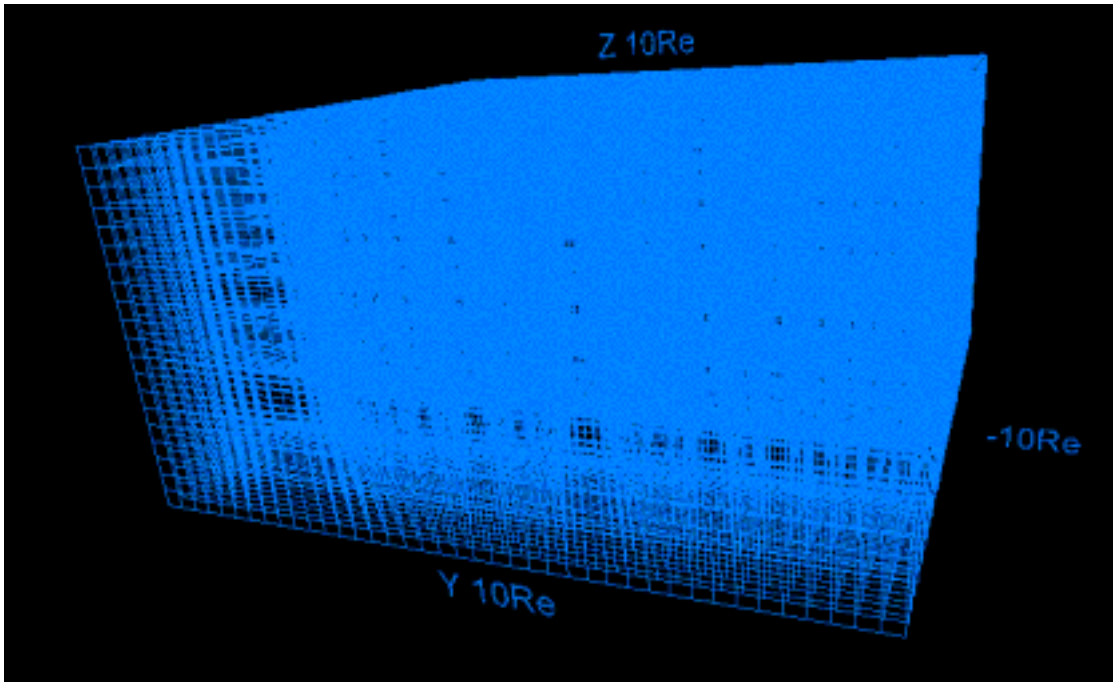
main program　inline3d.f
subroutine　lat3d2 (ic, aa) < (nx, ny, nz, xl, yl, zl, r, g, b)

説明：内部磁気圏を３次元格子で描くプログラム

入力変数：

nx = ic(1)：x 方向のデータ数
ny = ic(2)：y 方向のデータ数
nz = ic(3)：z 方向のデータ数
x1 = aa(1)：x 方向の一辺の長さ
y1 = aa(2)：y 方向の一辺の長さ
z1 = aa(3)：z 方向の一辺の長さ
r = aa(4)：赤色の値 [0,1]
g = aa(5)：緑色の値 [0,1]
b = aa(6)：青色の値 [0,1]

描画例：inline3d.wrl



２－１０．電離圏の３次元格子

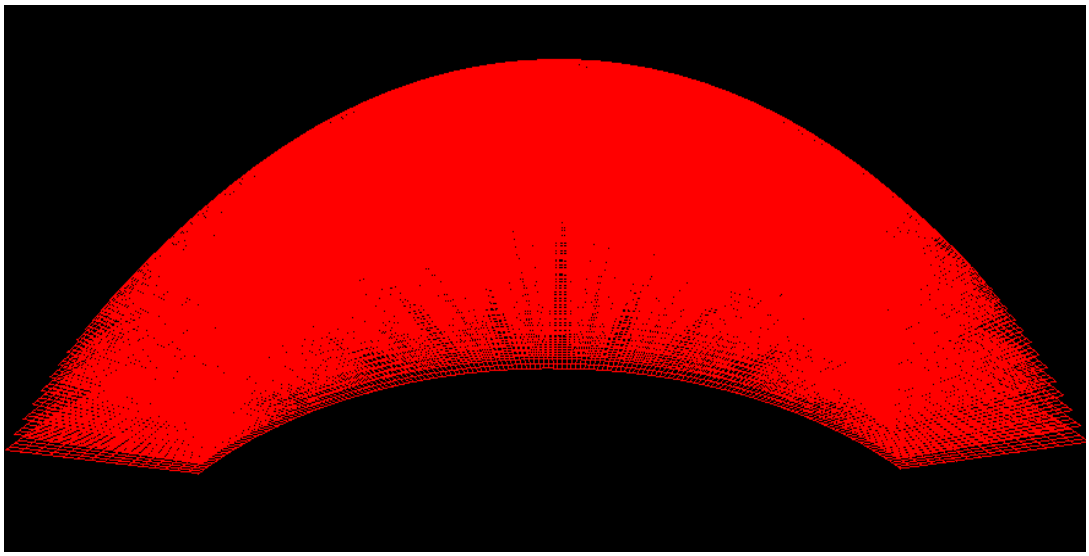
main program　denline3d.f
subroutine　lat3d3 (ic, aa) < (re1, re2, r, g, b)

説明：電離圏を３次元格子で描くプログラム

入力変数：

re1 = aa(1)：内円の半径（Re）
re2 = aa(2)：外円の半径（Re）
r = aa(3)：赤色の値 [0,1]
g = aa(4)：緑色の値 [0,1]
b = aa(5)：青色の値 [0,1]

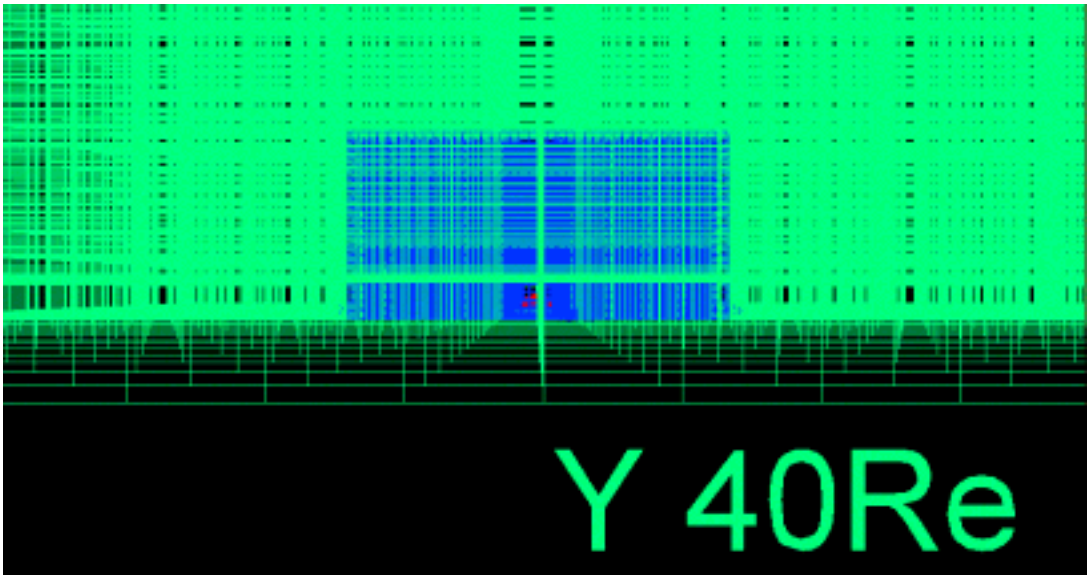
描画例：denline3d.wrl

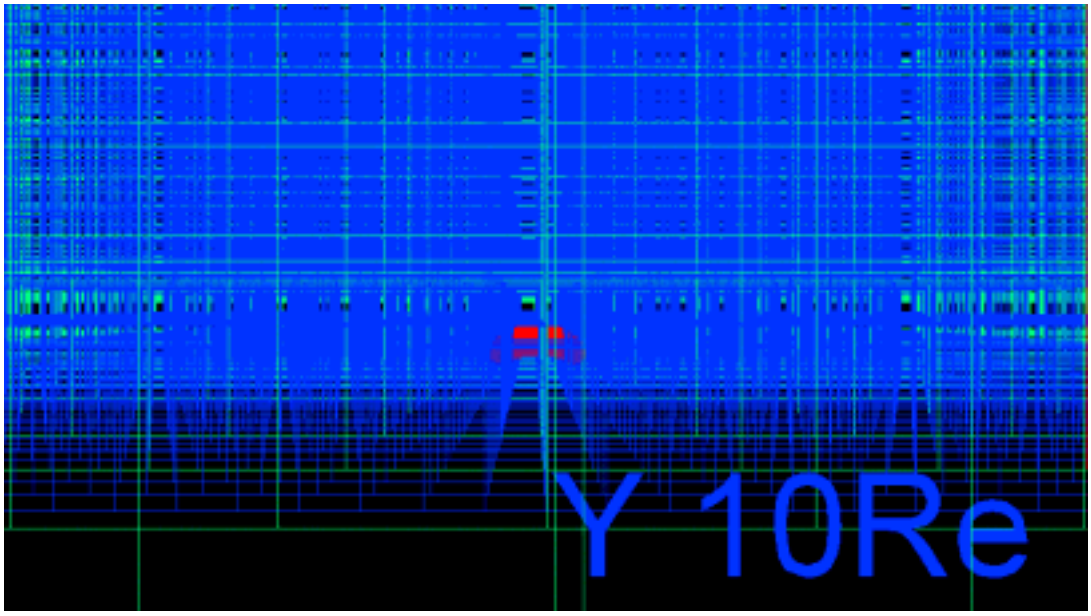


２－１１．外部磁気圏，内部磁気圏，電離圏を合成した図

cat outline3d.wrl inline3d.wrl denline3d.wrl > line3d.wrl

描画例：line3d.wrl





３．MHD シミュレーションへの応用

ここでは太陽風と地球磁気圏の３次元グローバルＭＨＤシミュレーションから得られた朝夕非対称の３次元磁気圏データをインプットに用いて、ＶＲＭＬファイルを作成する具体的な応用例を示す。

使用方法：

1. f77 -c -O zvrsuba.f

   サブルーチンパッケージ（zvrsuba.f）をコンパイルしてオブジェクトモジュール（zvrsuba.o）を作成する。

2. f77 -O zvrcroa.f zvrsuba.o

   メインプログラム（zvrcroa.f or zvrmaga.f）とオブジェクトモジュール（zvrsuba.o）をリンクして実行ファイル（a.out）を作成する。

3. a.out

   実行ファイル（a.out）を実行して VRML 2.0 ファイル（fort.10）を作成する。

4. mv fort.10 zvrcroa.wrl

   作成した VRML のファイル名を “ ***.wrl “ に変更する。

5. cat zvrcroa.wrl zvrmaga.wrl > zvr1.wrl

   ２つの VRML のファイルを合成する。

３－１．地球磁気圏のプラズマ温度等の断面図をピクセルイメージで描く

朝夕非対称の３次元ＭＨＤシミュレーションデータを入力として、メインプログラム：zvrcroa.f サブルーチンパッケージ：zvrsuba.f を実行して地球磁気圏のプラズマ温度等の断面図をピクセルイメージで描く VRML ファイルを作成する。

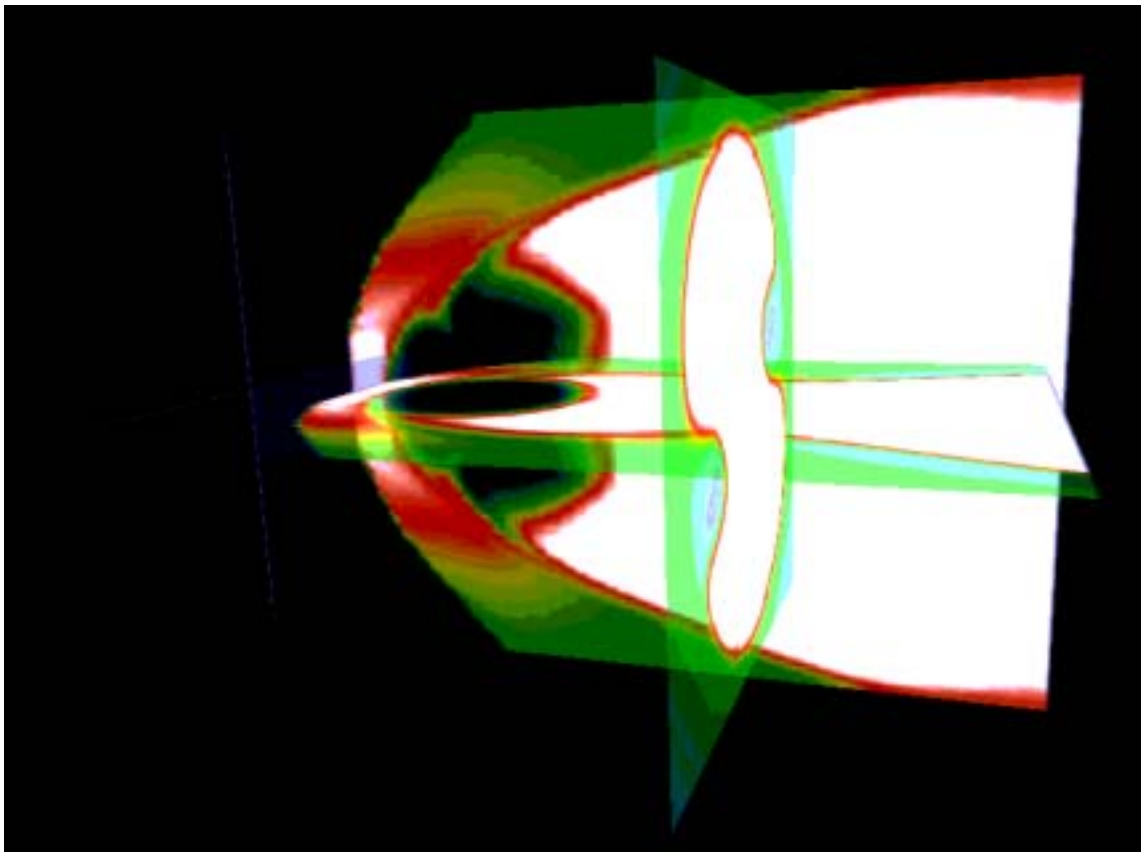
使用している主なサブルーチン：

pixel1(nx,ny,xb,yb,xl,yl,ipx0,ico,icc,zcc,vmin,vmax,u)

結果を受け取る 3D MHD シミュレーションコード：eartha10.f

描画例：地球磁気圏の３次元ＭＨＤシミュレーション結果の断面図を

半透明法を利用したピクセルイメージで描いている：zvrcroa.wrl



主なサブルーチンの詳細

＜最大値と最小値を利用してピクセルイメージを描く＞

subroutine pixel1(nx,ny,xb,yb,xl,yl,ipx0,ico,icc,zcc,vmin,vmax,u)

説明：u(i1)の最大値 vmax と最小値 vmin を外部から与えて u(i1)を正規化し、原点の

位置と一辺の長さ、データ数、カラーコードを与えて２次元データ u(nx*ny)の画像をピクセルイメージで描く

入力変数：

nx：x 方向のデータ数

ny：y 方向のデータ数

xb：原点の x 座標

yb：原点の y 座標

xl：x 方向の一辺の長さ

yl：y 方向の一辺の長さ

ipx0：3＝rgb カラー、4＝rgb カラーと透明度が設定できる

ico：カラーコード の種類 [1,7]

icc：1=xy 平面, 2=xz 平面, 3=yz 平面に描画

zcc：選択した面の位置

vmin：外部から与える u(i1)の最大値

vmax：外部から与える u(i1)の最小値

u (nx*ny)：入力データ u(i1), i1=i+nx*(j-1)

使用している主なサブルーチン：

pixebe(nx,ny,ipx0)：ピクセル画像の開始

pixeco(nx,ny,ico,u)：rgb カラーの設定

pixect(nx,ny,ico,u)：rgb カラーと透明度の設定

pixeen(xb,yb,xl,yl,icc,zcc)：ピクセル画像の終了

３－２．地球磁気圏の３次元構造を描く

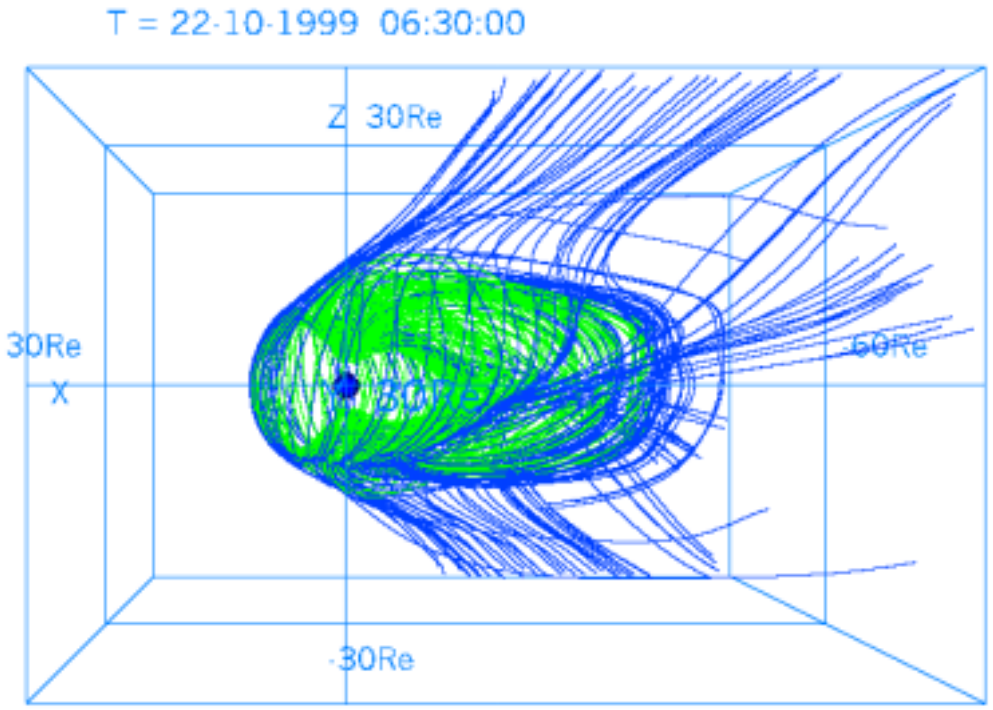
朝夕非対称の３次元MHDシミュレーションデータを入力として、メインプログラム：zvrmaga.f サブルーチンパッケージ：zvrsuba.f を実行して地球磁気圏の磁力線の３次元構造を描く VRML ファイルを作成する。

使用している主なサブルーチン：

ainte1a(ia,aa,f,p)
zsub33(ia,aa)
ainte1a(ia,aa,f,p)
ainte21(ia,aa,f,p)

結果を受け取る 3D MHD シミュレーションコード：eartha10.f

描画例：zvrmaga.wrl



主なサブルーチンの詳細

＜磁力線を予め描いて判定＞

subroutine ainte1a(ia,aa,f,p)

説明：３次元のMHDシミュレーションデータ f(i1)から３成分の磁場データを用いて

磁力線を追跡し、open, closed, detached 磁力線の判定を行う

入力変数：

nx=ia(1)：磁力線の出発点の x 方向のデータ数

ny=ia(2)：磁力線の出発点の y 方向のデータ数

mx=ia(3)：３次元データ f(i1)の x 方向のデータ数

my=ia(4)：３次元データ f(i1)の y 方向のデータ数

mz=ia(5)：３次元データ f(i1)の z 方向のデータ数

nxz=ia(6)：地球の中心を調節するパラメータ

mi=ia(7)：入力変数の最初の位置

mo=ia(8)：出力変数の最初の位置

th0=aa(1)*pi/180.0：磁力線の出発点の最低緯度

aru=aa(2)：内側境界の遷移を特徴づけるパラメータ

ar1=aa(3)：内側境界の遷移領域の外端半径

arb=aa(4)：内側境界の遷移領域の内端半径

xxl=aa(5)：x 方向のシミュレーションボックスの長さ

yyl=aa(6)：y 方向のシミュレーションボックスの長さ

zzl=aa(7)：z 方向のシミュレーションボックスの長さ

b0=aa(8)：ダイポール磁場の大きさ（通常 1.0）

gx0=aa(13)：グラフを描く原点の x 座標

gy0=aa(14)：グラフを描く原点の y 座標

gxl=aa(15)：グラフを描く x 方向の長さ

gyl=aa(16)：グラフを描く y 方向の長さ

gth=aa(17)*pi/180.0：グラフの座標の回転角度

gxmi=aa(18)：グラフを描く x 方向の最小値

gxma=aa(19)：グラフを描く x 方向の最大値

ep1=aa(20)：判定の最小閾値

使用している主なサブルーチン：

quant1(x,y,z,hx,hy,hz,1,mx,my,mz,nxp,aa,f,q)：

３次元格子点のＭＨＤデータ f(i1)から任意の場所(x,y,z)

のＭＨＤデータ q(8)を計算

＜３次元の磁力線を極域から出発して描く＞

subroutine ainte1(ia,aa,f,p)

説明：３次元のＭＨＤシミュレーションデータ f(i1)から３成分の磁場データを用いて、極域から出発して磁力線を追跡して描く

入力変数：

nx=ia(1)：磁力線の出発点の x 方向のデータ数

ny=ia(2)：磁力線の出発点の y 方向のデータ数

mx=ia(3)：３次元データ f(i1)の x 方向のデータ数

my=ia(4)：３次元データ f(i1)の y 方向のデータ数

mz=ia(5)：３次元データ f(i1)の z 方向のデータ数

nxz=ia(6)：地球の中心を調節するパラメータ

mi=ia(7)：入力変数の最初の位置

mo=ia(8)：出力変数の最初の位置

th0=aa(1)*pi/180.0：磁力線の出発点の最低緯度

aru=aa(2)：内側境界の遷移を特徴づけるパラメータ

ar1=aa(3)：内側境界の遷移領域の外端半径

arb=aa(4)：内側境界の遷移領域の内端半径

xxl=aa(5)：x 方向のシミュレーションボックスの長さ

yyl=aa(6)：y 方向のシミュレーションボックスの長さ

zzl=aa(7)：z 方向のシミュレーションボックスの長さ

b0=aa(8)：ダイポール磁場の大きさ（通常 1.0）

gx0=aa(13)：グラフを描く原点の x 座標

gy0=aa(14)：グラフを描く原点の y 座標

gxl=aa(15)：グラフを描く x 方向の長さ

gyl=aa(16)：グラフを描く y 方向の長さ

gth=aa(17)*pi/180.0：グラフの座標の回転角度

gxmi=aa(18)：グラフを描く x 方向の最小値

gxma=aa(19)：グラフを描く x 方向の最大値

ep1=aa(20)：判定の最小閾値

使用している主なサブルーチン：

quant1(x,y,z,hx,hy,hz,1,mx,my,mz,nxp,aa,f,q)：３次元格子点のＭＨＤデータ f(i1)

から任意の場所(x,y,z)のＭＨＤデータ q(8)を計算

linebe(r,g,b)：折れ線描写の開始

linep2(x1,y2,z1,1,lasl)：折れ線描写の実行

lineen(lasl)：折れ線描写の終了

＜３次元の磁力線を赤道面から出発して描く＞

subroutine ainte21(ia,aa,f,p)

３次元のＭＨＤシミュレーションデータ f(i1)から３成分の磁場データ

を用いて、赤道面から出発して磁力線を追跡して描く

入力変数：

nx=ia(1)：磁力線の出発点の x 方向のデータ数

ny=ia(2)：磁力線の出発点の y 方向のデータ数

mx=ia(3)：３次元データ f(i1)の x 方向のデータ数

my=ia(4)：３次元データ f(i1)の y 方向のデータ数

mz=ia(5)：３次元データ f(i1)の z 方向のデータ数

nxz=ia(6)：地球の中心を調節するパラメータ

mi=ia(7)：入力変数の最初の位置

mo=ia(8)：出力変数の最初の位置

th0=aa(1)*pi/180.0：磁力線の出発点の最低緯度

aru=aa(2)：内側境界の遷移を特徴づけるパラメータ

ar1=aa(3)：内側境界の遷移領域の外端半径

arb=aa(4)：内側境界の遷移領域の内端半径

xxl=aa(5)：x 方向のシミュレーションボックスの長さ

yyl=aa(6)：y 方向のシミュレーションボックスの長さ

zzl=aa(7)：z 方向のシミュレーションボックスの長さ

b0=aa(8)：ダイポール磁場の大きさ（通常 1.0）

gx0=aa(13)：グラフを描く原点の x 座標

gy0=aa(14)：グラフを描く原点の y 座標

gxl=aa(15)：グラフを描く x 方向の長さ

gyl=aa(16)：グラフを描く y 方向の長さ

gth=aa(17)*pi/180.0：グラフの座標の回転角度

gxmi=aa(18)：グラフを描く x 方向の最小値

gxma=aa(19)：グラフを描く x 方向の最大値

ep1=aa(20)：判定の最小閾値

xlim=aa(23)：x 座標の最小値の制限

xli2=aa(24)：y,z 座標の最小値の制限

使用している主なサブルーチン：

quant1(x,y,z,hx,hy,hz,1,mx,my,mz,nxp,aa,f,q)：３次元格子点のＭＨＤデータ f(i1)

から任意の場所(x,y,z)のＭＨＤデータ q(8)を計算

linebe(r,g,b)：折れ線描写の開始

linep2(x1,y2,z1,1,lasl)：折れ線描写の実行

lineen(lasl)：折れ線描写の終了

＜オープンとクローズド領域の境界の検定＞

subroutine zsub33(ia,aa)

説明：ainte1a の結果と用いて、地球磁気圏の３次元磁力線の中から、オープン領域とクローズド領域の間の狭い領域に存在する磁力線を選び出して、その選択された磁力線だけ描くように検定結果を ainte1 に渡す。

入力変数：

nxg=ia(1)：磁力線の出発点の x 方向のデータ数

nyg=ia(2)：磁力線の出発点の y 方向のデータ数

関連するサブルーチン：

ainte1a(ia,aa,f,p)
ainte1(ia,aa,f,p)

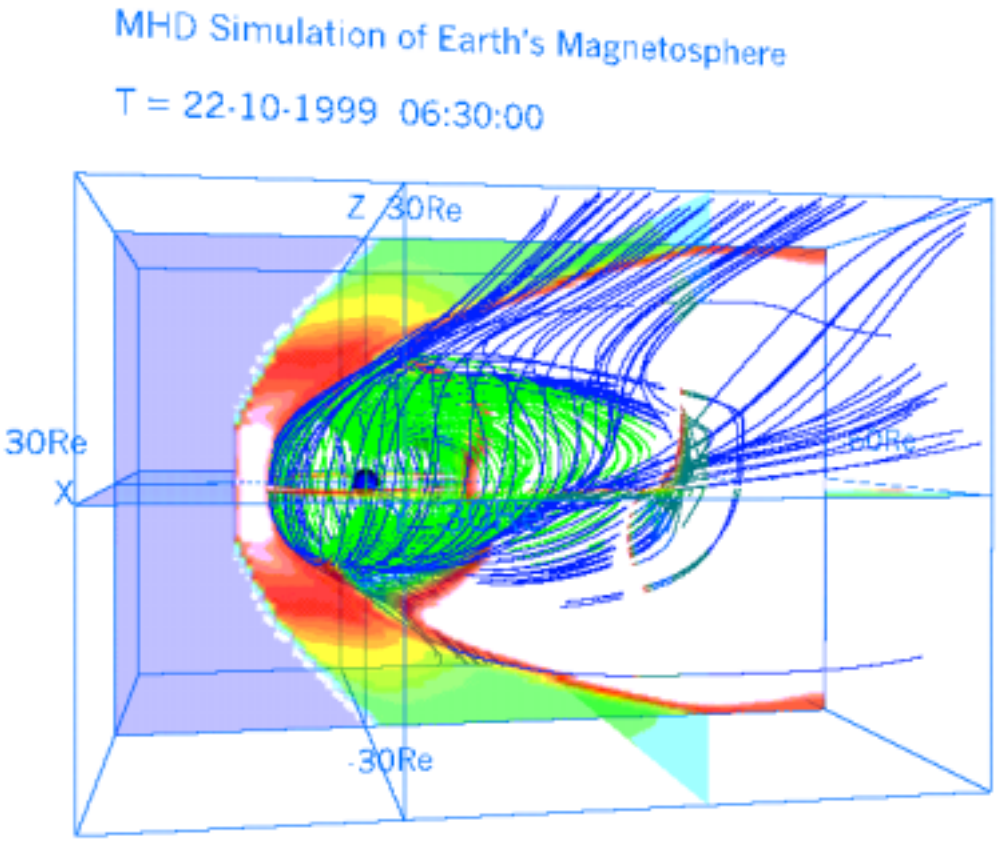
３－３．ピクセルイメージと磁力線の３次元画像の合成

地球磁気圏のプラズマ温度等の断面図をピクセルイメージで描いたもの（３－１参照 ）と地球磁気圏の磁力線の３次元構造を描いたもの（３－２参照 ）を”cat”で合成した例を示す。

zvrcroa.wrl + zvrmaga.wrl → zvr01.wrl

cat zvrcroa.wrl zvrmaga.wrl > zvr01.wrl

描画例：zvr01.wrl



３－４．地球磁気圏のプラズマ温度分布を、多重ピクセル面イメージを使って描く

朝夕非対称の３次元ＭＨＤシミュレーションデータを入力として、メインプログラム：zvrmpxa.f とサブルーチンパッケージ：zvrsuba.f をコンパイル・リンク・実行して、地球磁気圏のプラズマ温度分布をピクセル面を重ねて描く VRML ファイルを作成する。

入力変数

tra : ピクセル面の透明度 (0.0 ~ 0.5 : 0.0 で透明度 100%、0.5 で透明度 50%)

ipn : 描く面の数

icc : 描く面の種類 (1 : zx 面、2 : yz 面、3 : xy 面)

使用している主なサブルーチン (３－１参照)：

pixel1(nx,ny,xb,yb,xl,yl,ipx0,ico,icc,zcc,vmin,vmax,tra,u)

描画例：地球磁気圏の３次元ＭＨＤシミュレーション結果の温度分布を

半透明法を利用したピクセルイメージで描いている

fig.3-1 : fa410zx.wrl zx 面を重ねて描いた例 (icc=1, ipn=11 の場合)

fig.3-2 : fa410yz.wrl yz 面を重ねて描いた例 (icc=2, ipn=11 の場合)

fig.3-3 : fa410xy.wrl xy 面を重ねて描いた例 (icc=3, ipn=11 の場合)

fig.3-4 : fa410xyz.wrl 上記 3 つの方向で描いたファイルを cat で合成した例

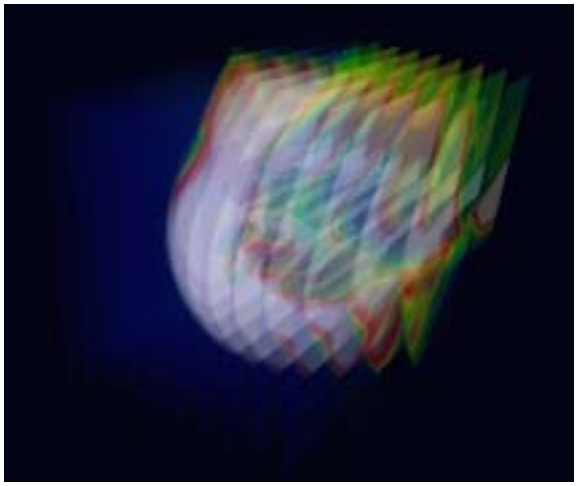


fig.3-1 : fa410zx.gif
(made from fa410zx.wrl)

zx 面を重ねて描いた図 (multiple zx planes)

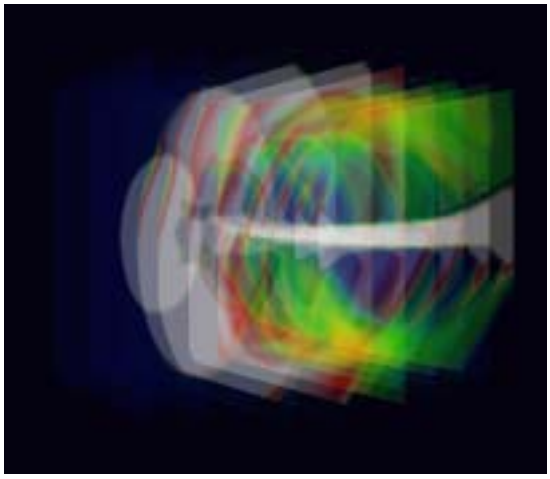


fig.3-2 : fa410yz.gif
(made from fa410yz.wrl)

yz 面を重ねて描いた図 (multiple yz planes)

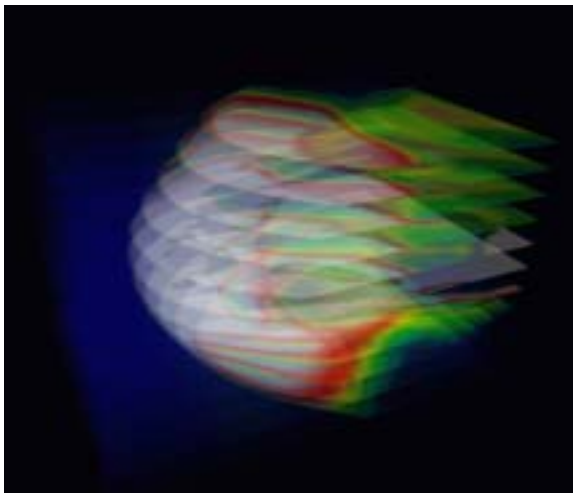


fig.3-3 : fa410xy.gif
(made from fa410xy.wrl)

xy 面を重ねて描いた図 (multiple xy planes)

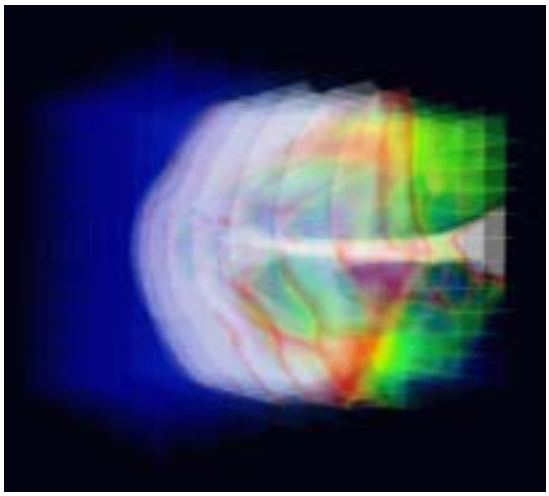


fig.3-4 : fa410xyz.gif
(made from fa410xyz.wrl)

3 方向の画像を重ねた合成図

4. 3-Dimensional MHD Simulation of Earth's Magnetosphere (English)

<Example to execute the MHD Code and Graphic programs>

We will demonstrate how to execute the 3-Dimensional magnetohydrodynamic (MHD) Simulation of Earth's Magnetosphere in 1/4 volume and how to use the graphics programs to make PostScript files and VRML files in this section. In the MHD model, MHD and Maxwell's equations are solved in the solar-magnetospheric coordinate system by using modified leap-frog method when the upstream solar wind and interplanetary magnetic field (IMF) boundary conditions are given. Moreover, north-south symmetry and dawn-dusk symmetry are assumed, therefore it is enough to solve 1/4 volume as the simulation box. The main simulation Fortran program, earthb10.f is fully vectorized and can be executed on many kinds of computers. By executing the main MHD simulation code, a simulated binary file is produced as output. When the output binary file is used as input, graphics programs can be executed to make PostScript files and VRML files for three dimensional visualization.

main program : earthb10.f
earthb10.f using modified leap-frog scheme
3D MHD simulation of 1/4 earth's magnetosphere
Cartesian coordinate finite resistivity 45 degree boundary

graphics program to make PostScript files

1. gm150b.f (main) + gsub150.f (subroutine)
   noon-midnight meridian and equatorial plots (black and white)
2. gm220b.f (main) + gsub220.f (subroutine)
   energy distribution of cross section
3. gm480b.f (main) + gsub480.f (subroutine)
   3-dimensional magnetic field lines

3-dimensional graphics program by VRML files
<Virtual Reality Modeling Language>

1. zvrmagb.f (main) + zvrsubbf (subroutine)
   3-dimensional magnetic field lines
2. zvrcrob.f (main) + zvrsubb.f (subroutine)
   cross sectional pattern by pixel image

<<Summary of parameters in MHD Simulation Code>>

(nx,ny,nz) = (180,60,60)	grid number without boundary
nxp=30	parameter to determine earth position
last=1024	number of time steps
iiq0=8	a unit of modified leap-frog scheme
iip0= 32	adjust upstream boundary condition
iis0= 1024	sampling step of data
thx=4.00	parameter to adjust time step

(xl,yl,zl) = (90.5,30.5,30.5)Re	length in each direction
hx=xl/float(nx+1)=0.5Re	grid interval in x direction
hy=yl/float(ny+1)=0.5Re	grid interval in y direction
hz=zl/float(nz+1)=0.5Re	grid interval in z direction
t=0.5*hx*thx	time interval
t(real)=t*ts	real time to one time step advance
=0.5*0.5*4.00*0.937	ts is normalization value in time
=0.937 sec	

x=0.5*hx*float(2*i-nx2-1+2*nxp)	x position versus grid number
y=0.5*hy*float(2*j-3)	y position versus grid number
z=0.5*hz*float(2*k-3)	z position versus grid number

where nx2=nx+2, ny2=ny+2 and nz2=nz+2

ro01=5.0E-4 (5/cc)	mass density of solar wind
pr01=3.56E-8	pressure of solar wind
vsw=0.044 (300km/s)	speed of solar wind
bis=CP(11)=1.5E-4 (5nT)	amplitude of IMF

eatt	resistivity
rmuu	viscosity
eud0	friction or collision term

1-dimensional array variable f(i1)=f(i,j,k,m)

n1=nx+2,n2=n1*(ny+2),n3=n2*(nz+2)
nb=8,nbb=11,n4=n3*nb,n5=n3*nbb

i1=i+n1*(j-1)+n2*(k-1)+n3*(m-1)

m=1 : rho, plasma density
m=2 : Vx
m=3 : Vy
m=4 : Vz
m=5 : P, plasma pressure
m=6 : Bx
m=7 : By
m=8 : Bz

<<execution of main program>>
1. f77 -O earthb10.f
2. a.out &

where file must be defined in open statement like

```
c           open(10,file='earthb10.data',
c       1               access='sequential',form='unformatted')
            open(11,file='earthb11.data',
        1               access='sequential',form='unformatted')
c
```

or

1. f77 -o earthb10 -O earthb10.f
2. earthb10 &

<<execution of PostScript graphics program>>

1. f77 -c -O gsub150.f
2. f77 -O gm150b.f gsub150.o
3. a.out > gm150b.ps &
4. gs gm150b.ps
5. lp gm150b.ps

1. f77 -c -O gsub220.f
2. f77 -O gm220b.f gsub220.o
3. a.out > gm220b.ps &

1. f77 -c -O gsub480b.f
2. f77 -O gm480b.f gsub480b.o
3. a.out & : output is written in fort.10

<<execution of VRML graphics program>>

1. f77 -c -O zvrsubb.f
2. f77 -O zvrmagb.f zvrsubb.o
3. a.out & : output is written in fort.10
4. mv fort.10 fort.102

1. f77 -c -O zvrsubb.f
5. f77 -O zvrcrob.f zvrsubb.o
6. a.out & : output is written in fort.10
7. mv fort.10 fort.101
8. cat fort.101 fort.102 > zvrml01.wrl

References:

T. Ogino, A three-dimensional MHD simulation of the interaction of the solar wind with the earth's magnetosphere: The generation of field-aligned currents, J. Geophys. Res., 91, 6791-6806 (1986).

T. Ogino, R.J. Walker and M. Ashour-Abdalla, A global magnetohydrodynamic simulation of the magnetosheath and magnetopause when the interplanetary magnetic field is northward, IEEE Transactions on Plasma Science, Vol.20, No.6, 817-828 (1992).

T. Ogino, Two-Dimensional MHD Code, (in Computer Space Plasma Physics), Ed. by H. Matsumoto and Y. Omura, Terra Scientific Publishing Company, 161-215, 411-467 (1993).

T. Ogino, R.J. Walker and M. Ashour-Abdalla, A global magnetohydrodynamic simulation of the response of the magnetosphere to a northward turning of the interplanetary magnetic field, J. Geophys. Res., Vol.99, No.A6, 11,027-11,042 (1994).

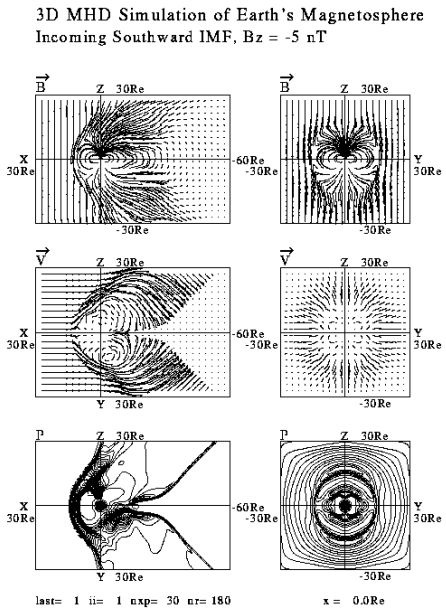


Fig.4-1. gm150b.gif
(converted from gm150b.ps)

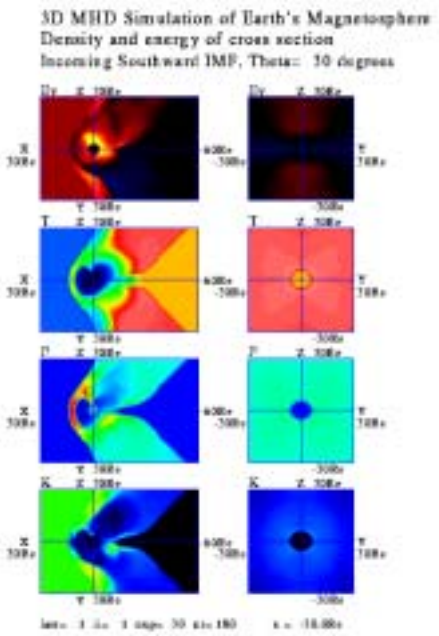


Fig.4-2. gm220b.gif
(converted from gm220b.ps)

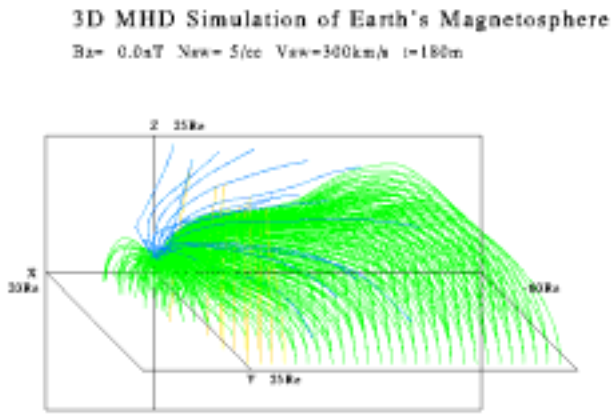


Fig.4-3. gm480b.gif (converted from gm480b.ps)

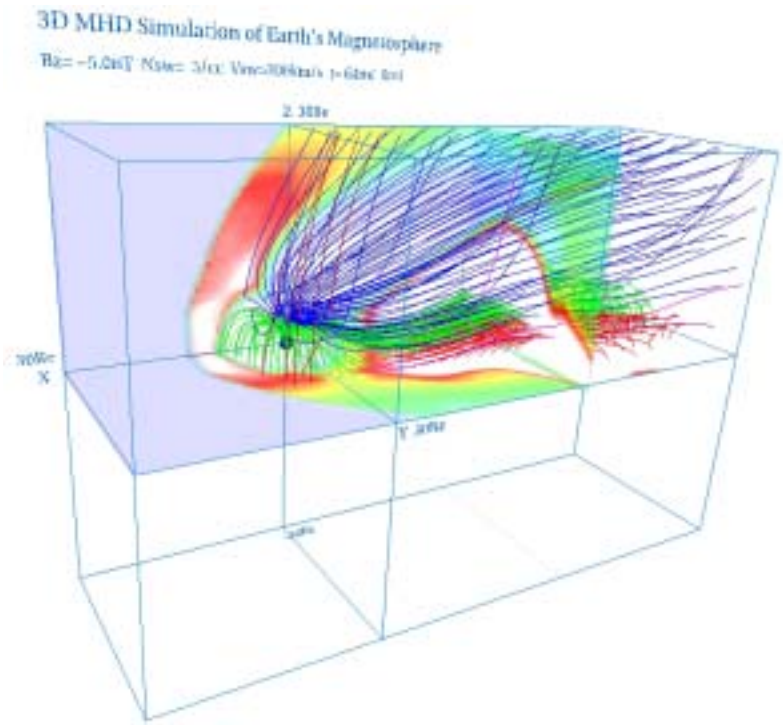


Fig.4-4. zvrml1b.jpg (made from zvrml1b.wrl)

5. Visualization to Various 3-Dimensional MHD Models of Earth's Magnetosphere (English)

We will show how to use the graphics programs to make VRML files of various 3-Dimensional MHD models of Earth's Magnetosphere. The output binary files produced by various MHD simulation codes by using modified leap-frog scheme are used as input.

<<execution of VRML graphics program>>

1. f77 -c -O zvrsubb.f
2. f77 -O zvrmagb.f zvrsubb.o

```
  where file must be defined in open statement like
c          open(10,file='earthb10.data',
c       1             access='sequential',form='unformatted')
           open(11,file='earthb11.data',
        1             access='sequential',form='unformatted')
c
```

3. a.out & : output is written in fort.10
4. mv fort.10 fort.102

1. f77 -c -O zvrsubb.f
5. f77 -O zvrcrob.f zvrsubb.o
6. a.out & : output is written in fort.10
7. mv fort.10 fort.101
8. cat fort.101 fort.102 > zvrml01.wrl

<<Summary of parameters in MHD Simulation Code>>

(nx,ny,nz) = (180,60,60)	grid number without boundary
nxp=30	parameter to determine earth position
(xxl,yyl,zzl) = (90.5,30.5,30.5)Re	length in each direction
hx=xxl/float(nx+1)=0.5Re	grid interval in x direction
hy=yyl/float(ny+1)=0.5Re	grid interval in y direction
hz=zzl/float(nz+1)=0.5Re	grid interval in z direction

1-dimensional array variable f(i1)=f(i,j,k,m)
i1=i+n1*(j-1)+n2*(k-1)+n3*(m-1)

m=1 : rho, plasma density
m=2 : Vx, x component of velocity
m=3 : Vy, y component of velocity
m=4 : Vz, z component of velocity
m=5 : P, plasma pressure
m=6 : Bx, x component of magnetic field
m=7 : By, y component of magnetic field
m=8 : Bz, z component of magnetic field

5-a. Half volume model of earth's magnetosphere with IMF By and Bz components

<<example data file used in this section>>

gg220410.data

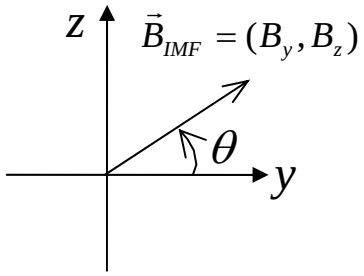
(nx,ny,nz) = (180,60,120)

nxp = 30

time = 410m

$\vec{B}_{IMF} = (B_y, B_z)$

theta = 30 deg



<<3-dimensional graphics program by VRML files>>

1. 3-dimensional magnetic field lines
   zvrmaga.f (main) + zvrsuba.f (subroutine) > fa410m.wrl
2. cross sectional pattern by pixel image
   zvrcroa.f (main) + zvrsuba.f (subroutine) > fa410c.wrl
3. cross sectional pattern by velocity vectors
   zvrvela.f (main) + zvrsuba.f (subroutine) > fa410v.wrl

   a synthesis file with above three files using "cat" > fa410w.wrl

4. 3-dimensional pixel image with multi-planes
   zvrmpxa.f (main) + zvrsuba.f (subroutine)
   - zx planes selected in parameter > fa410zx.wrl
   - yz planes selected in parameter > fa410yz.wrl
   - xy planes selected in parameter > fa410xy.wrl

   a synthesis file with above three files using "cat" > fa410xyz.wrl

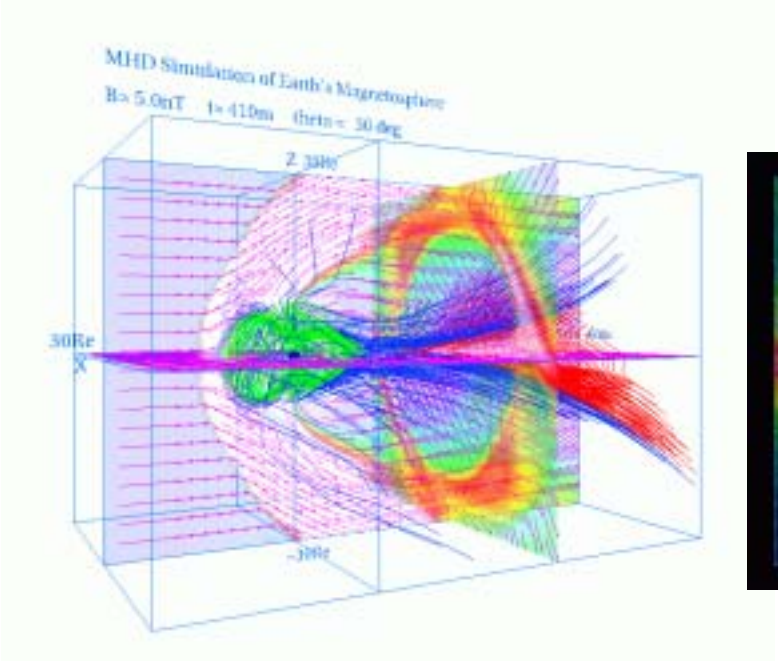


fig.5-1 : fa410w.gif (made from fa410w.wrl)
Whole image of magnetosphere

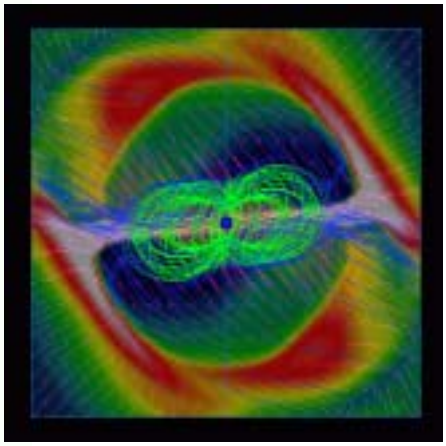


fig.5-2 : fa410s.gif
(made from fa410w.wrl)
View from the sun

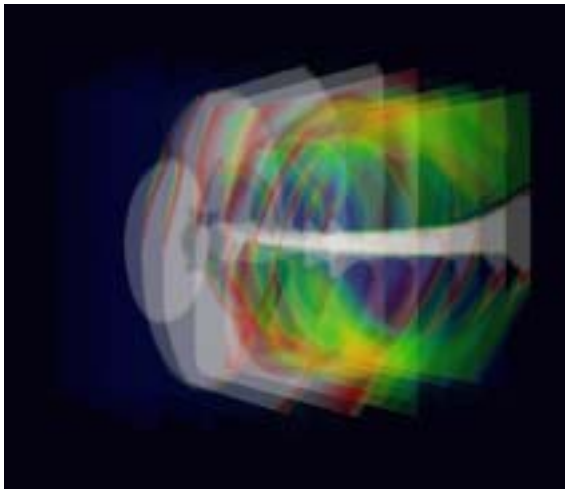


fig.5-3 : fa410yz.gif (made from fa410yz.wrl)
Image of multiple yz planes

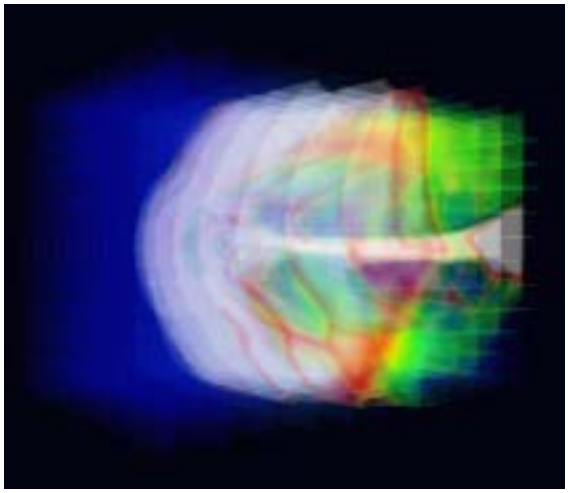


fig.5-4 : fa410xyz.gif (made from fa410xyz.wrl)
Synthesis image with multiple planes
in three directions

5-b. Quarter volume model of earth's magnetosphere

<<example data file used in this section>>
ggswb240.data
(nx,ny,nz) = (180,60,60)
nxp = 30
time = 240m
IMF *Bz* (*By* = 0)

<<3-dimensional graphics program by VRML files>>

1. 3-dimensional magnetic field lines
   zvrmagb.f (main) + zvrsubb.f (subroutine) > fb240m.wrl
2. cross sectional pattern by pixel image
   extended pattern to whole volume
   zvrcrob.f (main) + zvrsubb.f (subroutine) > fb240c.wrl
   quater volume pattern
   zvrcrobq.f (main) + zvrsubb.f (subroutine) > fbq240c.wrl
3. cross sectional pattern by velocity vectors
   extended pattern to whole volume
   zvrvelb.f (main) + zvrsubb.f (subroutine) > fb240v.wrl
   quater volume pattern
   zvrvelbq.f (main) + zvrsubb.f (subroutine) > fbq240v.wrl

a synthesis file with fb240m.wrl , fb240c.wrl and fb240v.wrl using "cat" > fb240w.wrl
a synthesis file with fb240m.wrl , fbq240c.wrl and fbq240v.wrl using "cat" > fbq240w.wrl

4. 3-dimensional pixel image with multi-planes
   zvrmpxb.f (main) + zvrsubb.f (subroutine)
   zx planes selected in parameter > fb240zx.wrl
   yz planes selected in parameter > fb240yz.wrl
   xy planes selected in parameter > fb240xy.wrl

a synthesis file with above three files using "cat" > fb240xyz.wrl

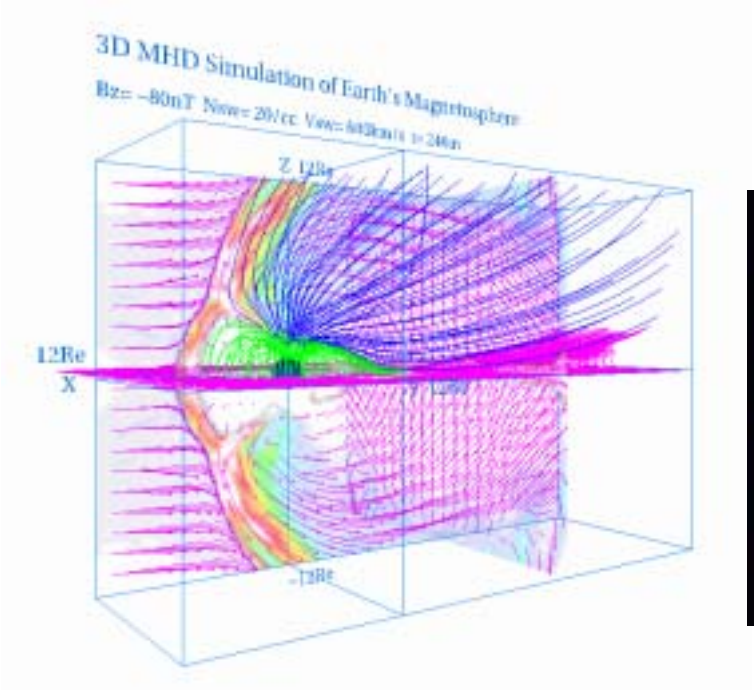


fig.5-5 : fb240w.gif (made from fb240w.wrl)
Whole image of magnetosphere

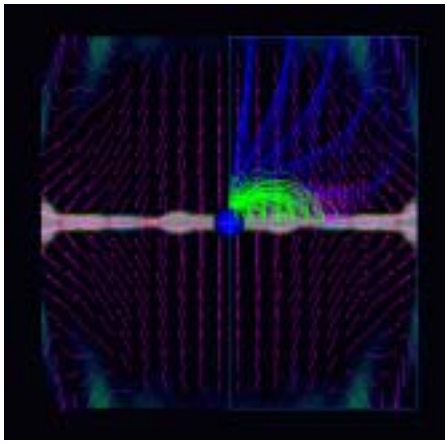


fig.5-6 : fb240s.gif
(made from fb240w.wrl)
View from the sun

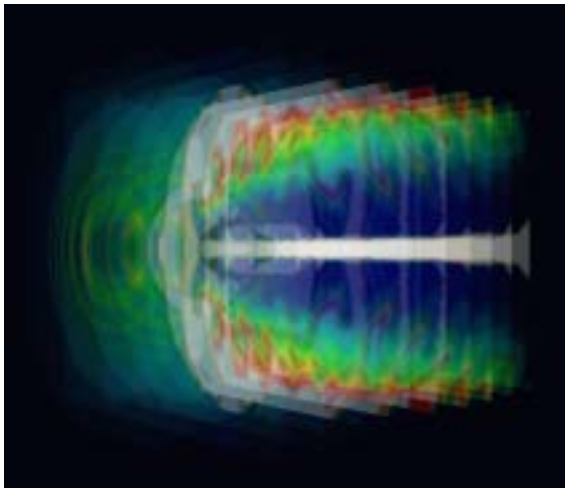


fig.5-7 : fb240yz.gif (made from fb240yz.wrl)
Image of multiple yz planes

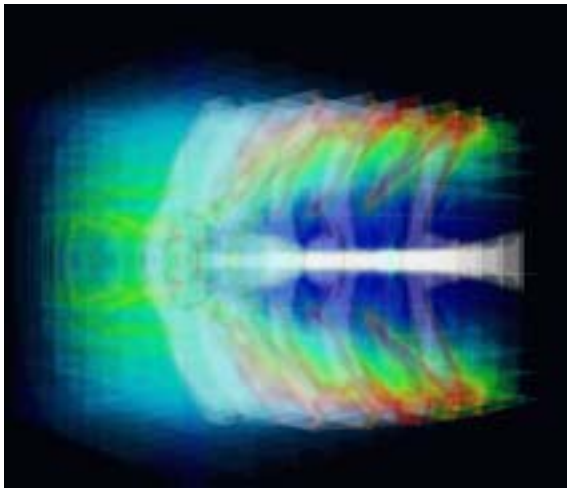


fig.5-8 : fb240xyz.gif (made from fb240xyz.wrl)
Synthesis image with multiple planes
in three directions

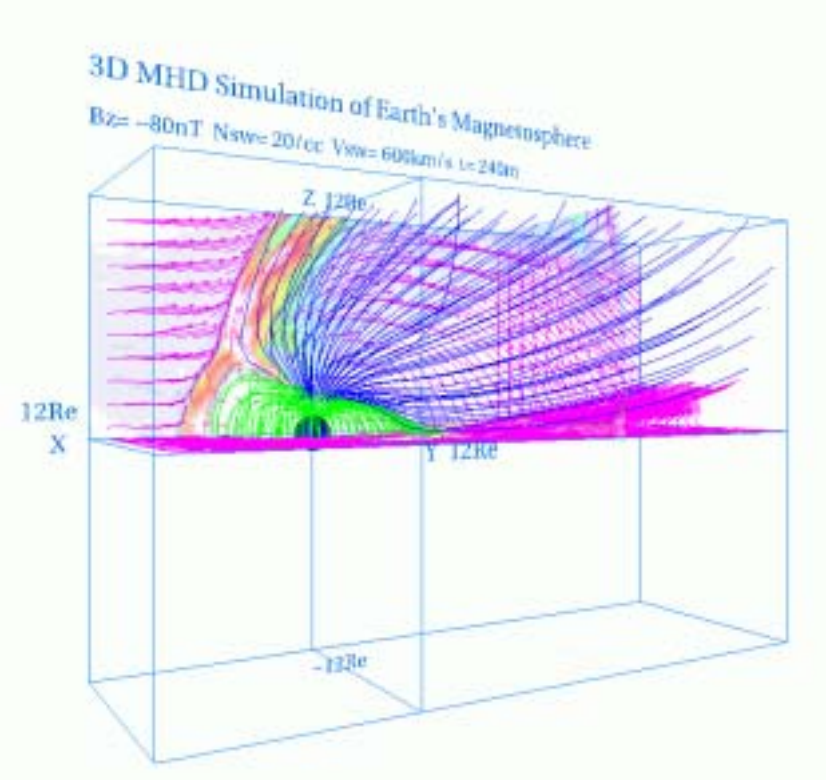


fig.5-9 : fbq240w.gif (made from fbq240w.wrl)
Whole image of magnetosphere

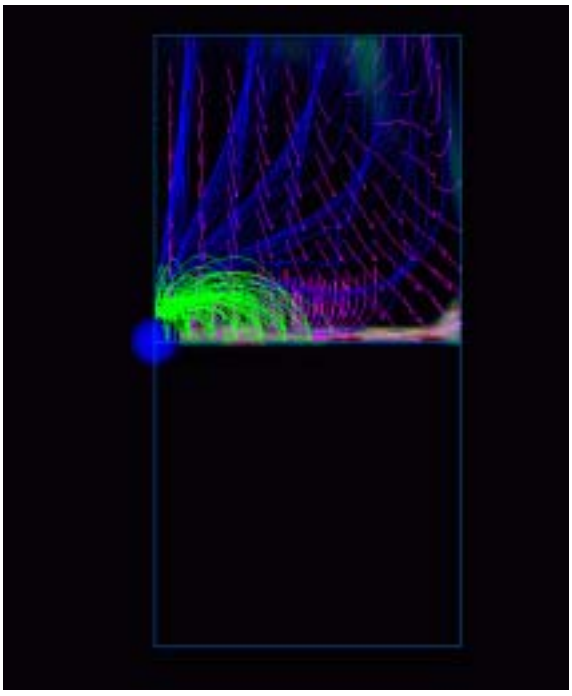


fig.5-10 : fbq240s.gif (made from fbq240w.wrl)
View from the sun

5-c. Half volume model of earth's magnetosphere with dipole tilt

<<example data file used in this section>>
gcc1a360.data
(nx,ny,nz) = (180,60,120)
nxp = 30
time = 360m
dipole tilt + IMF Bz ($By = 0$)

<<3-dimensional graphics program by VRML files>>
1. 3-dimensional magnetic field lines
zvrmagc.f (main) + zvrsubc.f (subroutine) > fc360m.wrl
2. cross sectional pattern by pixel image
zvrcroc.f (main) + zvrsubc.f (subroutine) > fc360c.wrl
3. cross sectional pattern by velocity vectors
zvrvelc.f (main) + zvrsubc.f (subroutine) > fc360v.wrl

a synthesis file with above three files using "cat" > fc360w.wrl

4. 3-dimensional pixel image with multi-planes
zvrmpxc.f (main) + zvrsubc.f (subroutine)
zx planes selected in parameter > fc360zx.wrl
yz planes selected in parameter > fc360yz.wrl
xy planes selected in parameter > fc360xy.wrl
a synthesis file with above three files using "cat" > fc360xyz.wrl

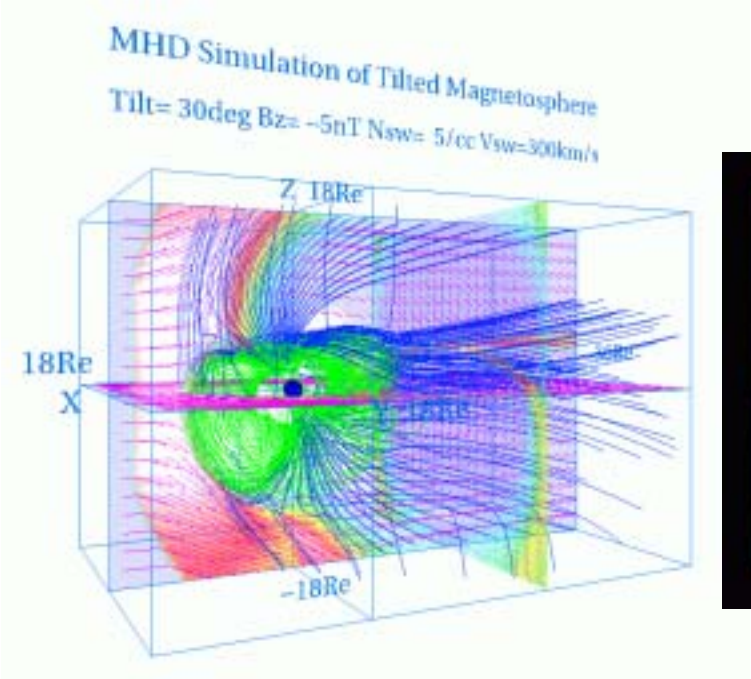


fig.5-11 : fc360w.gif (made from fc360w.wrl)
Whole image of magnetosphere

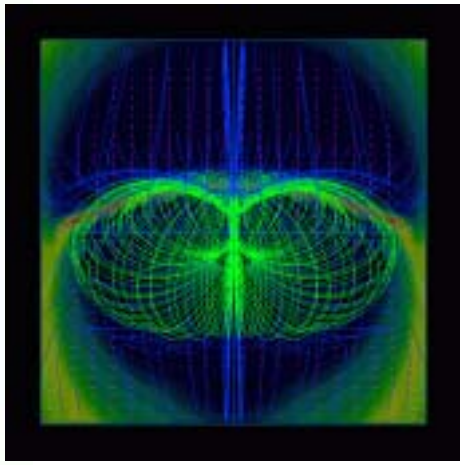


fig.5-12 : fc360s.gif
(made from fc360w.wrl)
View from the sun

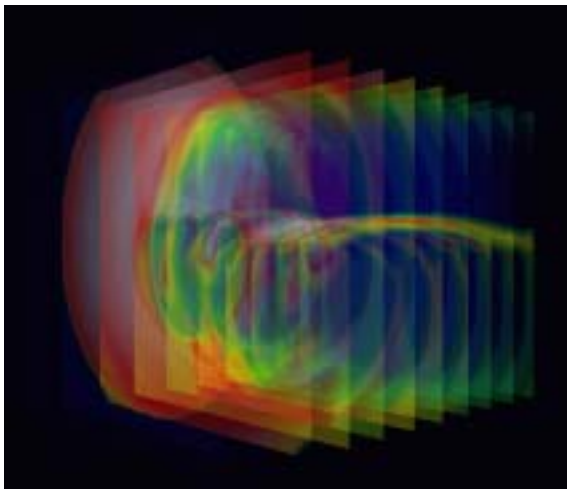


fig.5-13 : fc360yz.gif (made from fc360yz.wrl)
Image of multiple yz planes

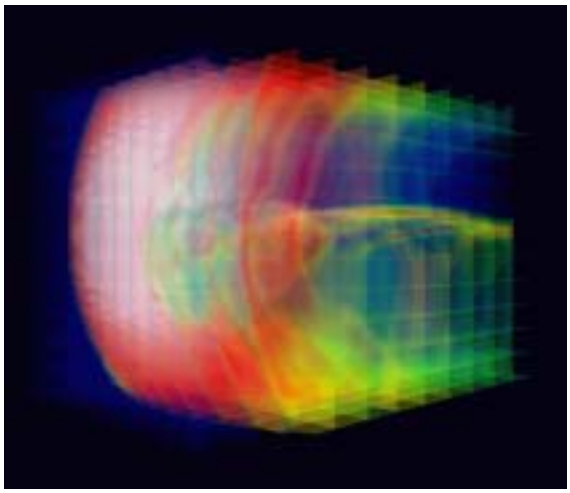
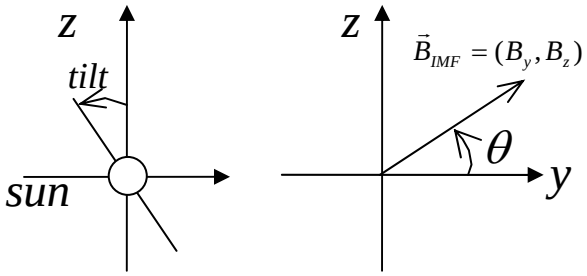


fig.5-14 : fc360xyz.gif (made from fc360xyz.wrl)
Synthesis image with multiple planes
in three directions

5-d. Whole volume model of earth's magnetosphere

<<example data file used in this section>>
gdd1e180.data
(nx,ny,nz) = (180,120,120)
nxp = 30
time = 180m
dipole tilt + IMF $\vec{B}_{IMF} = (B_y, B_z)$
tilt = 30 deg
theta = 45deg



<<3-dimensional graphics program by VRML files>>

1. 3-dimensional magnetic field lines
   zvrmagd.f (main) + zvrsubd.f (subroutine) > fde180m.wrl
2. cross sectional pattern by pixel image
   zvrcrod.f (main) + zvrsubd.f (subroutine) > fde180c.wrl
3. cross sectional pattern by velocity vectors
   zvrveld.f (main) + zvrsubd.f (subroutine) > fde180v.wrl

   a synthesis file with above three files using "cat" > fde180w.wrl

4. 3-dimensional pixel image with multi-planes
   zvrmpxd.f (main) + zvrsubd.f (subroutine)
   zx planes selected in parameter > fde180zx.wrl
   yz planes selected in parameter > fde180yz.wrl
   xy planes selected in parameter > fde180xy.wrl

   a synthesis file with above three files using "cat" > fde180xyz.wrl

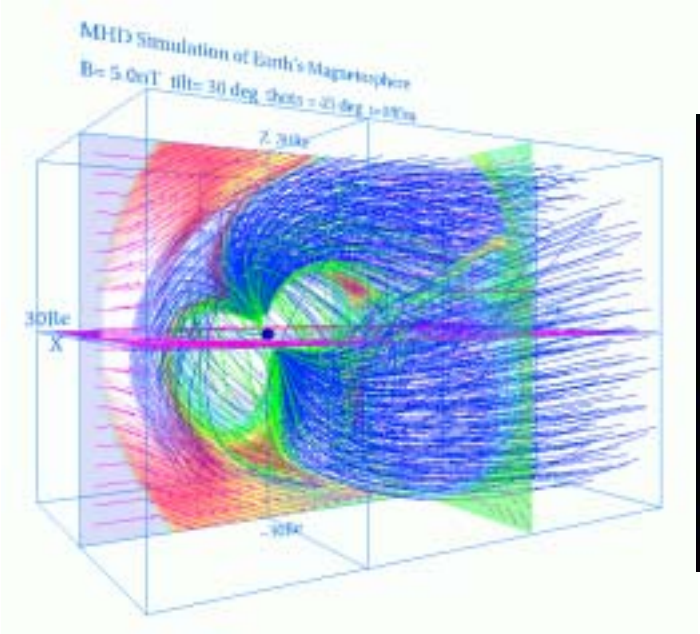


fig.5-15 : fde180w.gif (made from fde180w.wrl)
Whole image of magnetosphere

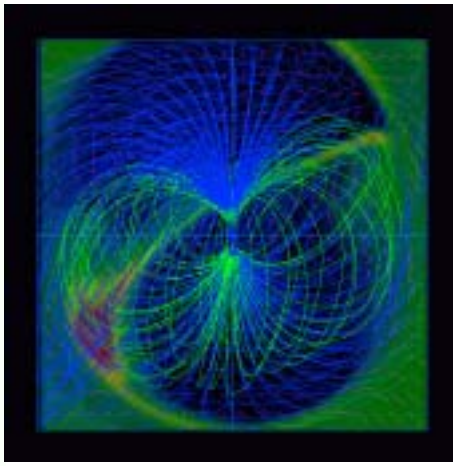


fig.5-16 : fde180s.gif
(made from fde180w.wrl)
View from the sun

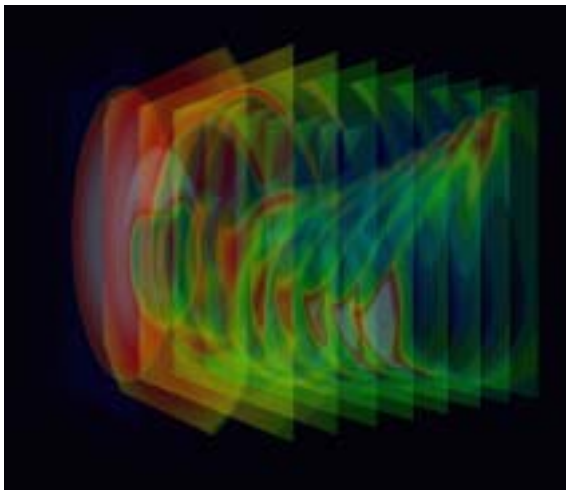


fig.5-17 : fde180yz.gif
(made from fde180yz.wrl)
Image of multiple yz planes

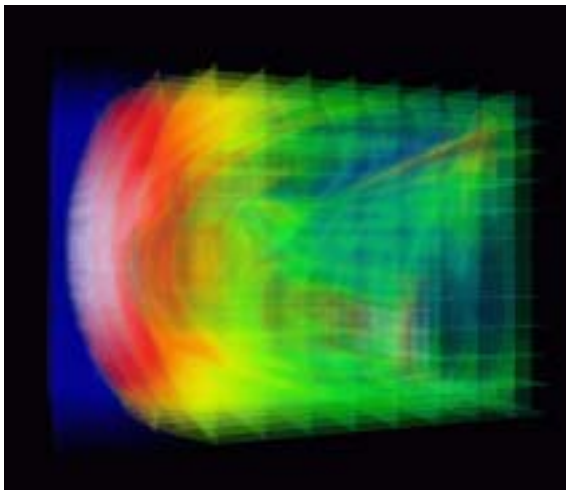


fig.5-18 : fde180xyz.gif
(made from fde180xyz.wrl)
Synthesis image with multiple planes in three directions

6. Application of VRML to 3-Dimensional MHD Models of Earth's Magnetosphere (English)

6-a. 3-dimensional visualization of earth's magnetosphere with multiple planes

<<example data file used in this section>>

idd1i540.data

(nx,ny,nz) = (268,134,134)

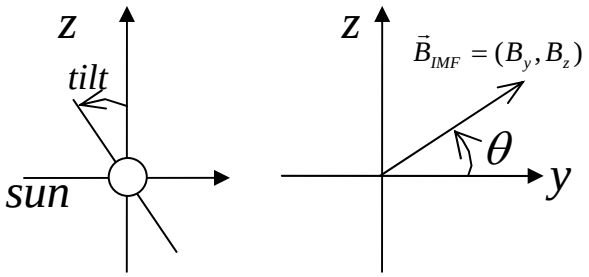
nxp = 30

time = 540m

dipole tilt + IMF $\vec{B}_{IMF} = (B_y, B_z)$

tilt = 30deg

theta = 0deg



<<3-dimensional graphics program by VRML files>>

1. 3-dimensional magnetic field lines

   zvrmagd.f (main) + zvrsubd.f (subroutine) > zvrmagdc2.idd1i540.wrl

2. Plasma velocity vectors in cross section

   zvrveld.f (main) + zvrsubd.f (subroutine) > zvrveldc.idd1i540.wrl

3. Pixel image of plasma temperate with multiple planes

   zvrmpxd.f (main) + zvrsubd.f (subroutine)

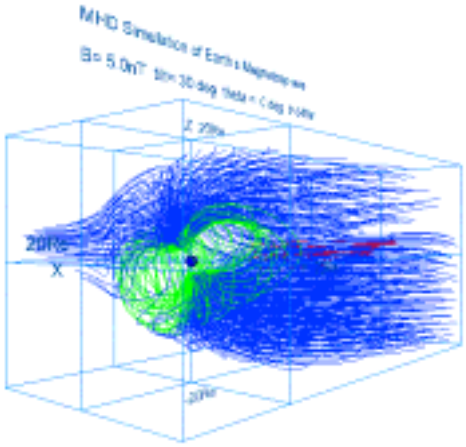
   zx planes selected in parameter > pxdzx. idd1i540.wrl

   yz planes selected in parameter > pxdyz. idd1i540.wrl

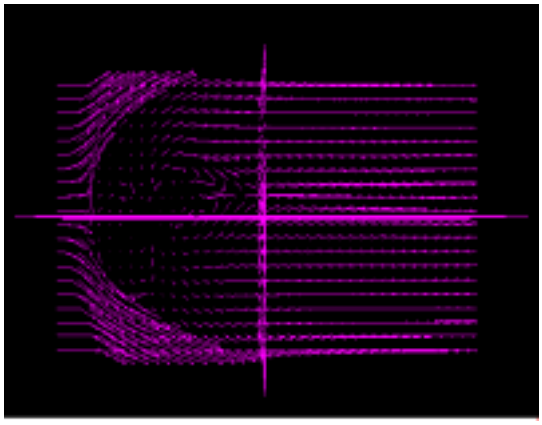
   xy planes selected in parameter > pxdxy idd1i540.wrl

   a synthesis file with above three files using "cat" > pxd4. idd1i540.wrl

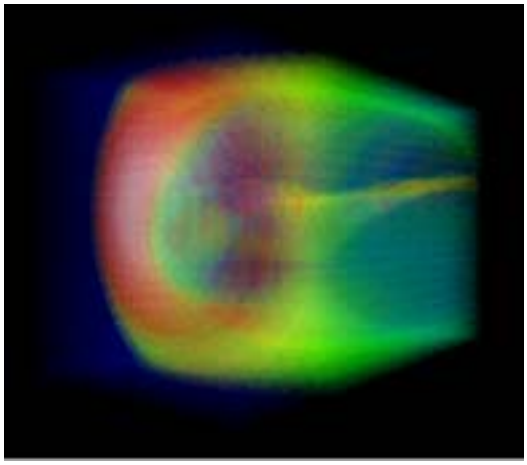
4. a synthesis file with above three files using "cat" > pxd4mdc2v.idd1i540.wrl



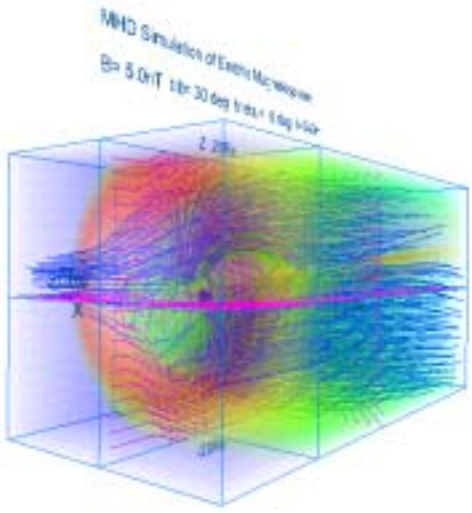
zvrmagdc2.idd1i540.gif
(made from zvrmagdc2.idd1i540.wrl)
3-dimensional magnetic field lines



zvrveldc.idd1i540.gif
(made from zvrveldc.idd1i540.wrl)
Plasma velocity vectors in cross section



pxd4.idd1i540.gif
(made from pxd4.idd1i540.wrl)
Pixel image of plasma temperate
with multiple planes



pxd4mdc2v.idd1i540.gif
(made from pxd4mdc2v.idd1i540.wrl)
Synthesis image with multiple planes
in three directions

6-b. 3-dimensional visualization of earth's magnetosphere with equivalence planes

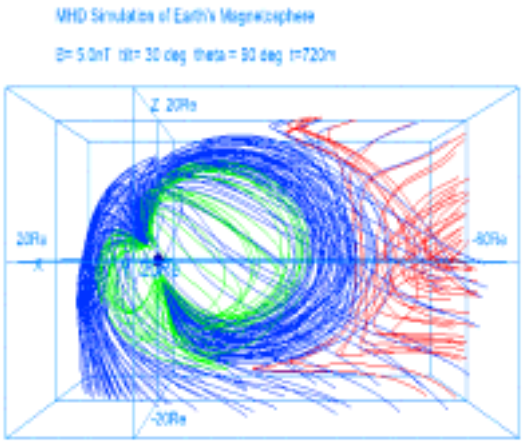
<<example data file used in this section>>
idd1n720.data
(nx,ny,nz) = (268, 134, 134)
nxp = 30
time = 720m
dipole tilt + IMF
tilt = 30deg
theta = 90deg

<<3-dimensional graphics program by VRML files>>
1. 3-dimensional magnetic field lines
zvrmagd.f (main) + zvrsubd.f (subroutine) > zvrmagd. idd1n720.wrl

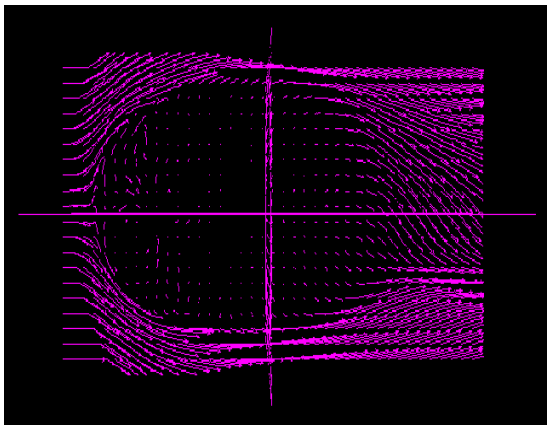
2. Plasma velocity vectors in cross section
zvrveld.f (main) + zvrsubd.f (subroutine) > zvrveld. idd1n720.wrl

3. 3-dimensional visualization with equivalence planes
mcubvrm.f (main) + mcubsub.f (subroutine) > mcub.idd1n720.wrl

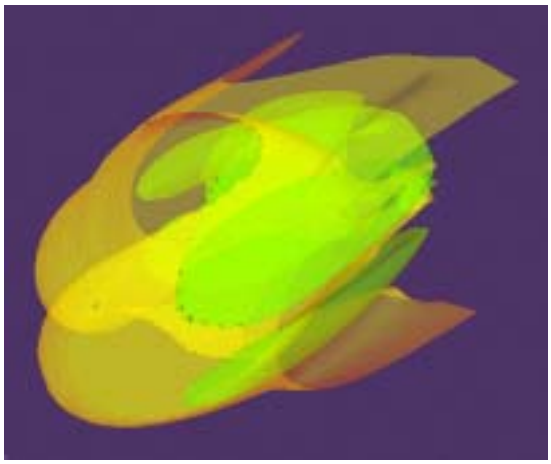
4. a synthesis file with above three files using "cat" > mcubmagdv.idd1n720.wrl



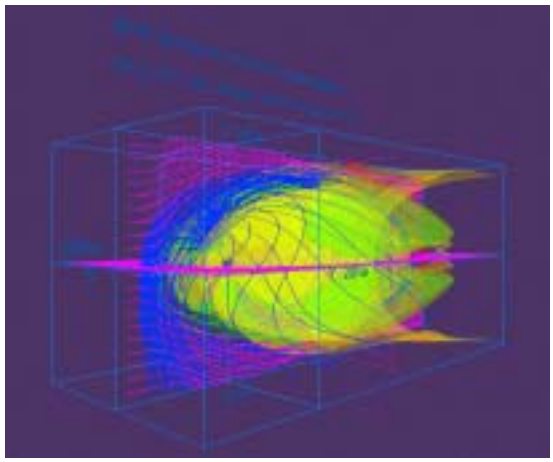
zvrmagdc2.idd1n720.gif
(made from zvrmagdc2.idd1n720.wrl)
3-dimensional magnetic field lines



zvrveldc.idd1n720.gif
(made from zvrveldc.idd1n720.wrl)
Plasma velocity vectors in cross section



mcub.idd1n720.gif
(made from mcub.idd1n720.wrl)
3-dimensional visualization
with equivalence planes



mcubdc2v.idd1n720.gif
(made from mcubmagdv.idd1n720.wrl)
Synthesis image with equivalence planes
in three directions

6-c. Cropping of 3-dimensional data

main program mbikxyz.f

Input variable :

nx : Data number of the x direction
ny : Data number of the y direction
nz : Data number of the z direction
x0 : The x coordinate of the origin
y0 : The y coordinate of the origin
z0 : The z coordinate of the origin
mb : The number of cropping

Description : This program produce a reduced array, fb(jx,jy,jz) from the original away, fa(ix,iy,iz), where mb is the number of cropping.

ix = 1+mb*(jx-1)
iy = 1+mb*(iy-1)
iz = 1+mb*(iz-1)

7. C による等値面作成法を用いたスカラー物理量の３次元可視化

C プログラムによる等値面作成法を用いて、VRML2.0 型式のスカラー物理量の３次元可視化画像ファイルを作成する方法と基本的な例題を示す。具体的には、mcube001.c をコンパイル・実行して VRML2.0 型式の画像ファイルを作成する。

使用方法：

1. cc mcube001.c
   メインプログラム（mcube001.c etc.）をコンパイルし、実行ファイル（a.out）を作成する。
2. a.out
   output file : mcube001.wrl
   実行ファイル（a.out）を実行し、「output file :」に続いて出力ファイル名(mcube001.wrl etc.) を入力し、VRML 2.0 ファイル(mcube001.wrl etc.)を作成する。

（１）marching-cubes 法による等値面作成

main program　mcube001.c
subroutine　mcube000.h

説明：３次元のデータ v[nX][nY][nZ]を与えて等値面を描くプログラム

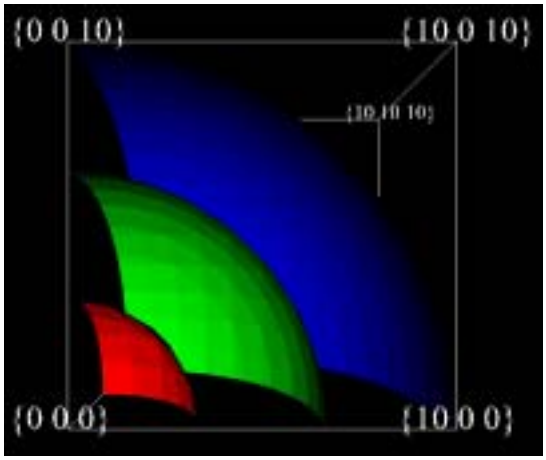
入力変数：

ヘッダファイル(mcube000.h)

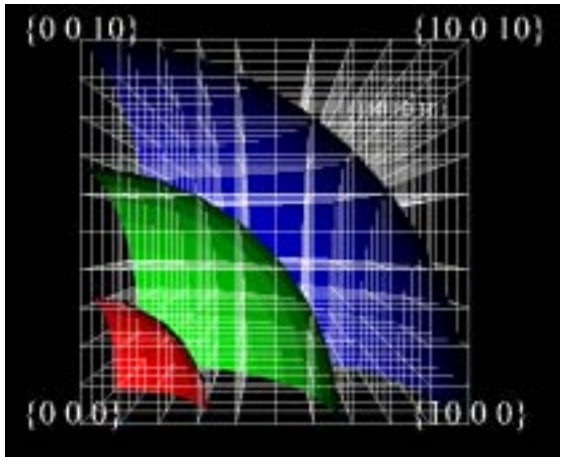
NX　：x 方向の格子数
NY　：y 方向の格子数
NZ　：z 方向の格子数
nX =NX+1：x 方向のデータ数
nY =NY+1：y 方向のデータ数
nZ =NZ+1：z 方向のデータ数

メインプログラム(mcube001.c)

Viewpoint(fp)：視点を決める関数　fp：出力ファイル
dr_grid(fp, gc)：格子を描く関数　　fp：出力ファイル　gc[3]：線の色
dr_scale(fp, gc)：目盛を描く関数　　fp：出力ファイル　gc[3]：線の色
mk_equ_face001(fp, th, v, c, tr)：等値面を１面描く関数
fp：出力ファイル th：閾値 c：等値面の色　v：３次元入力データ　tr：透明度



描画例：mcube0011.wrl



描画例：mcube0012.wrl(grid 有り)

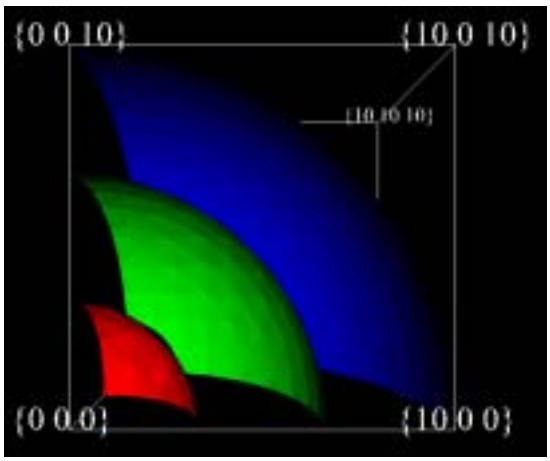
（1）四面体格子による等値面作成

main program　mcube002.c
subroutine　mcube000.h

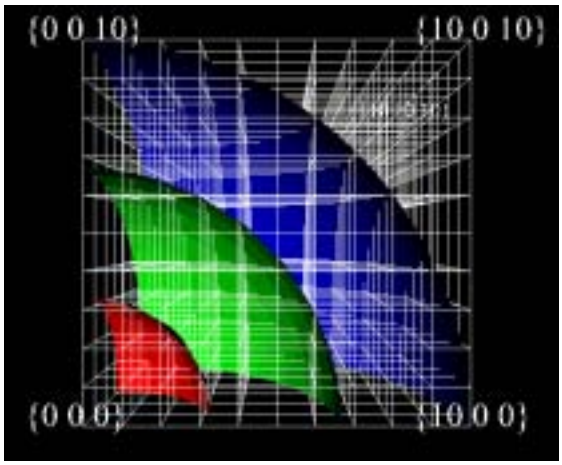
説明：３次元のデータ v[nX][nY][nZ]を与えて等値面を描くプログラム

入力変数：
　ヘッダファイル(mcube000.h)
　　NX　：x 方向の格子数
　　NY　：y 方向の格子数
　　NZ　：z 方向の格子数
　　nX =NX+1：x 方向のデータ数
　　nY =NY+1：y 方向のデータ数
　　nZ =NZ+1：z 方向のデータ数

　メインプログラム(mcube001.c)
　　Viewpoint(fp)：視点を決める関数　fp：出力ファイル
　　dr_grid(fp, gc)：格子を描く関数　　fp：出力ファイル　gc[3]：線の色
　　dr_scale(fp, gc)：目盛を描く関数　　fp：出力ファイル　gc[3]：線の色
　　mk_equ_face002(fp, th, v, c, tr)：等値面を１面描く関数
　　　fp：出力ファイル th：閾値 c：等値面の色　v：３次元入力データ　tr：透明度



描画例：mcube0021.wrl



描画例：mcube0022.wrl(grid 有り)

参考図書

VRML2.0　3D サイバースペース構築言語、三浦憲二郎著、朝倉書店、1996 年

Web3D グラフィックス、広内哲夫著、ピアソン・エデュケーション、2001 年

The VRML 2.0 Handbook, by Jed Hartman and Josie Wernecke, Silicon Graphic, Inc,
Addison-Wesley Publishing Co., 1996 年

注解 VRML 2.0 リファレンスマニュアル、by Rikk Carey and Gavin Bell,
Addison-Wesley Publishers Japan, 1998 年

PostScript リファレンスマニュアル 第２版、by Adobe Systems,アスキー出版局,1995 年

PostScript by Example, by H. McGilton and M. Campione,
Addison-Wesley Publishing Co., 1992 年

Fortran プログラムと画像ファイルリスト

１.基本的な例題1

No.	Contents	Fortran Program	VRML file	gif file
	サブルーチンパッケージ	pk1subrtn.f		
1	立方体	box1.f	box1.wrl	box1.gif
2	直方体	box2.f	box2.wrl	box2.gif
3	三角柱	cone1.f	cone1.wrl	cone1.gif
4	円柱	cylinder1.f	cylinder1.wrl	cylinder1.gif
5	球	sphere1.f	sphere1.wrl	sphere1.gif
6	背景色	backgrd4.f	backgrd4.wrl	backgrd4.gif
7	文字列	text1.f	text1.wrl	text1.gif
8	折れ線：単色	lineset1.f	lineset1.wrl	lineset1.gif
9	折れ線：グラデーション	lineset2.f	lineset2.wrl	lineset2.gif
10	点線：単色	pointset1.f	pointset1.wrl	pointset1.gif
11	点線：グラデーション	pointset2.f	pointset2.wrl	pointset2.gif
12	太い矢印	arrow2.f	arrow2.wrl	arrow2.gif
13	太さのある曲線	linemesh1.f	linemesh1.wrl	linemesh1.gif
14	面１：単色三角メッシュ	mesh1.f	mesh1.wrl	mesh1.gif
15	面２：三角ベルト	triangb1.f	triangb1.wrl	Triangb1.gif
16	面３：多色三角メッシュ	triangm1.f	triangm1.wrl	Triangm1.gif
17	立体：三角メッシュで構成	defusem1.f	defusem1.wrl	defusem1.gif
18	テクスチャ画像の貼りつけ	image1.f	image1.wrl	image1.gif
19	複雑な表面：三角メッシュ	ebmesh3d.f	ebmesh3d.wrl	Ebmesh3d.gif

２.基本的な例題2

No.	Contents	Fortran Program	VRML file	gif file	jpg file	rgb file
	サブルーチンパッケージ	zvrsuba.f				
1	文字を描く	msymbol.f	msymbol.wrl	msymbol.gif		
2	点を描く	mpoint.f	mpoint.wrl	mpoint.gif		
3	線を描く	mline.f	mline.wrl	mline.gif		
4	三角メッシュを描く	Mtriangm.f	mtriangm.wrl	mtriangm.gif	mtriangm.jpg	mtriangm.rgb
5	ピクセルイメージを描く	mzpt03.f	mzpt03.wrl	mzpt03.gif	mzpt03.jpg	mzpt03.rgb
6	枠とピクセルイメージと文字を描く	mpix015.f	mpix015.wrl	mpix015.gif	mpix015.jpg	mpix015.rgb
7	等値面を描く	mcube301.f	mcube301.wrl	mcube301.gif	mcube301.jpg	mcube301.rgb
8	外部磁気圏の３次元格子	outline3d.f	outline3d.wrl	outline3d.gif	outline3d.jpg	
9	内部磁気圏の３次元格子	inline3d.f	inline3d.wrl	inline3d.gif	inline3d.jpg	
10	電離圏の３次元格子	denline3d.f	denline3d.wrl	denline3d.gif	denline3d.jpg	

3.応用

No.	Contents	Fortran Program	VRML file	gif file
	サブルーチンパッケージ	zvrsuba.f		
1	地球磁気圏のプラズマ温度などの断面図をピクセルイメージで描く • 最大値と最小値を利用してピクセルイメージを描く	zvrcroa.f	zvrcroa.wrl	zvrcroa.gif
2	地球磁気圏の３次元構造を描く • 磁力線を予め描いて判定 • ３次元の磁力線を極域から出発して描く • ３次元の磁力線を赤道面から出発して描く • オープンとクローズド領域の境界の検定	zvrmaga.f	zvrmaga.wrl	zvrmaga.gif
3	ピクセルイメージと磁力線の３次元画像の合成		zvr01.wrl	zvr01.gif
	サブルーチンパッケージ	zvrsuba.f		
4	地球磁気圏のプラズマ温度分布を 多重ピクセル面イメージを使って描く	zvrmpxa.f	fa410zx.wrl fa410yz.wrl fa410xy.wrl fa410xyz.wrl	fa410zx.gif fa410yz.gif fa410xy.gif fa410xyz.gif

4.3-Dimensional MHD Simulation of Earth's Magnetosphere (Example to execute the MHD Code and Graphic programs)

No.	Contents	Fortran Main Program	Subroutine Package	PostScript file	VRML file or gif file
3D MHD simulation code of 1/4 earth's magnetosphere		earthb10.f			
Graphics program to make PostScript files					
1	noon-midnight meridian and equatorial plots (black and white)	gm150b.f	gsub150.f	gm150b.ps	gm150b.gif
2	energy distribution of cross section	gm220b.f	gsub220.f	gm220b.ps	gm220b.gif
3	3-dimensional magnetic field lines	gm480b.f	gsub480.f	gsub480.ps	gsub480.gif
3-dimensional graphics program by VRML files (Virtual Reality Modeling Language)					
1	3-dimensional magnetic field lines	zvrmagb.f	zvrsubb.f		zvrml1b.wrl
2	cross sectional pattern by pixel image	zvrcrob.f	zvrsubb.f		zvrml1b.jpg

5.Visualization to Various 3-dimensional MHD Models of Earth's Magnetosphere

5-a. Half volume model of earth's magnetosphere with IMF By and Bz components

No.	Contents	Fortran Main Program	VRML file	gif file
	Subroutine Package	zvrsuba.f		
1	3-dimensional magnetic field lines	zvrmaga.f	Fa410w.wrl	fa410w.gif fa410s.gif
2	cross sectional pattern by pixel image	zvrcroa.f		
3	cross sectional pattern by velocity vectors	zvrvela.f		
4	3-dimensional pixel image with multi-planes	zvrmpxa.f	Fa410xyz.wrl fa410yz.wrl	fa410xyz.gif fa410yz.gif

5-b. Quarter volume model of earth's magnetosphere

No.	Contents	Fortran Main Program	VRML file	gif file
	Subroutine Package	zvrsubb.f		
1	3-dimensional magnetic field lines	zvrmagb.f	fb240w.wrl fbq240w.wrl	fb240w.gif fb240s.gif fbq240w.gif fbq240s.gif
2	cross sectional pattern by pixel image	zvrcrob.f zvrcrobq.f		
3	cross sectional pattern by velocity vectors	zvrvelb.f zvrvelbq.f		
4	3-dimensional pixel image with multi-planes	zvrmpxb.f	fb240xyz.wrl fb240yz.wrl	fb240xyz.gif fb240yz.gif

5-c. Half volume model of earth's magnetosphere with dipole tilt

No.	Contents	Fortran Main Program	VRML file	gif file
	Subroutine Package	zvrsubc.f		
1	3-dimensional magnetic field lines	zvrmagc.f	fc360w.wrl	fc360w.gif fc360s.gif
2	cross sectional pattern by pixel image	zvrcroc.f		
3	cross sectional pattern by velocity vectors	zvrvelc.f		
4	3-dimensional pixel image with multi-planes	zvrmpxc.f	fc360xyz.wrl fc360yz.wrl	fc360xyz.gif fc360yz.gif

5-d. Whole volume model of earth's magnetosphere

No.	Contents	Fortran Main Program	VRML file	gif file
	Subroutine Package	zvrsubd.f		
1	3-dimensional magnetic field lines	zvrmagd.f	fde180w.wrl	fde180w.gif fde180s.gif
2	cross sectional pattern by pixel image	zvrcrod.f		
3	cross sectional pattern by velocity vectors	zvrveld.f		
4	3-dimensional pixel image with multi-planes	zvrmpxd.f	fde180xyz.wrl fde180yz.wrl	fde180xyz.gif fde180yz.gif

6. Application of VRML to 3-Dimensional MHD Models of Earth's Magnetosphere

6-a. 3-dimensional visualization of earth's magnetosphere with multiple planes

No.	Contents	Fortran Main Program	VRML file	gif file
	Subroutine Package	zvrsubd.f		
1	3-dimensional magnetic field lines	zvrmagd.f	zvrmagdc2.idd1i540.wrl	zvrmagdc2.idd1i540.gif
2	Plasma velocity vectors in cross section	zvrveld.f	zvrveldc.idd1i540.wrl	zvrveldc.idd1i540.gif
3	Pixel image of plasma temperate with multiple planes	zvrmpxdc5.f	pxd4.idd1i540.wrl	pxd4.idd1i540.gif
4	Synthesis image with multiple planes in three directions		pxd4mdc2v.idd1i540.wrl	pxd4mdc2v.idd1i540.gif

6-b. 3-dimensional visualization of earth's magnetosphere with equivalence planes

No.	Contents	Fortran Main Program	Subroutine Program	VRML file
1	3-dimensional magnetic field lines	zvrmagd.f	zvrsubd.f	zvrmagdc2.idd1n720.gif
2	Plasma velocity vectors in cross section	zvrveld.f	zvrsubd.f	zvrveldc.idd1n720.gif
3	3-dimensional visualization with equivalence planes	mcubvrm.f	mcubsub.f	mcub.idd1n720.gif
4	Synthesis image with multiple planes in three directions			mcubdc2v.idd1n720.gif

6-c. Cropping of 3-dimensional data

No.	Contents	Fortran Main Program	Subroutine Program	VRML file
1	cropping of 3-dimensional data	mbikxyz.f		

7. 3-dimensional visualization of scalar physical quantity with isosurfaces by C language

No.	Contents	C MainProgram	Subroutine Program	VRML file
1	Production of Isosurfaces by Marching-Cubes Method	mcube001.c	mcube000.h	mcube001.wrl
2	Production of Isosurfaces by Tetrahedrons	mcube002.c	mcube000.h	mcube002.wrl

謝辞：

この「VRML の利用方法」は、名古屋大学太陽地球環境研究所の「VRML による可視化ツールの開発と標準化」のプロジェクトチームで 1997 年から個々のサブルーチンを開発し、科学技術振興事業団の計算科学技術活用型特定研究開発推進事業「宇宙シミュレーション・ネットラボラトリーシステムの開発」で最初のマニュアルとしてまとめ、さらに科学研究費基盤研究(A)(1)「共通並列計算電磁流体・粒子コードによる太陽風磁気圏電離圏ダイナミックスの研究」で継続開発して、科学研究費研究成果公開促進費「宇宙天気国際協同研究データベース」でマニュアルとしてまとめたものである。また、本稿のコンピュータシミュレーションは名古屋大学情報連携基盤センターを利用してなされたものである。

「VRML による可視化ツールの開発と標準化」のプロジェクトチーム

荻野　竜樹　（リーダー）

中尾　真季

門脇　優香

菅沼　美奈子

稲吉　真奈美

太田　幸一

深沢　圭一郎

大脇　大

澤田　和英

平成１６年　３月

名古屋大学太陽地球環境研究所

〒442-8507　豊川市穂ノ原 3-13

TEL:0533-89-5207　　FAX:0533-89-5090

E-mail:ogino@stelab.nagoya-u.ac.jp


JST-CS


STEL