2009年度

卒業論文

インヤン格子上の磁気流体データ可視化プログラムの開発

神戸大学情報知能工学科

吉田　真人

指導教員　陰山　聡

2010年2月23日

要旨

宇宙天気予報プロジェクトの一環として Yin-Yang 格子上の MHD データを可視化するプログラム SV4 (Spherical data Visualizer in Fortran) の開発した。

Yin-Yang 格子とは球ジオメトリで計算機シミュレーションを効率的に行うために最近考案された計算格子である。SV4 は、Yin-Yang 格子上で定義されたシミュレーションデータ— 特に磁場 —を、他の格子系に変換せず、直接可視化処理する。ユーザーが指定した任意の点（複数）から出発した磁力線を Yin-Yang 格子上で数値的に積分し、OpenGL を用いて滑らかな曲線として可視化する機能を実現した。SV4 は GLUT を用いたクラフィカルユーザーインターフェースを備えており、対話的に様々な角度や位置から表示した磁力線を観察することが可能である。SV4 の一つの特徴は Fortran90 言語で書かれているという点である。これは宇宙天気予報プロジェクトに参加するシミュレーション研究者が C/C++ 言語よりも Fortran90 言語になじみがあり、この言語で書かれた可視化ツールをソースコードごと提供して自由に改変したいという強い要望があるためである。そのため SV4 では Fortran 用の OpenGL モジュール “f90gl” を利用した。

目次

1 緒論

人工衛星は現代社会になくてはならない存在である。その人工衛星がさらされる環境は地球上に比べて大変厳しい。特に太陽表面でフレアや CME(Coronal Mass Ejection) などの爆発現象が起こると、衛星軌道上の人工衛星が高エネルギーの荷電粒子にさらされて故障の原因となる。また、この種の激しい太陽活動がいわゆる磁気嵐として地球上でも大きな影響を及ぼし、電波障害や送電線の異常電流などを引き起こすことがある。これらの悪影響を回避するため、太陽フレアなどを予測する宇宙天気予報の必要性が高まっている。

現在、日本国内においては名古屋大学の草野完也教授を中心として宇宙天気予報のプロジェクト（宇宙天気モデリング共同プロジェクト）が進められている [1]。宇宙天気予報に必要とされるのは太陽コロナ、太陽風、地球磁気圏、電離層シミュレーション等のシミュレーションコードを結合した極めて大規模なシミュレーションである。これらのコードを国内各地（名古屋、横浜、京都、広島等）に分散した多数の研究者が分担して開発を進めている。この宇宙天気予報プロジェクトの進行と共に急速に必要性が高まってきたものの一つとして専用の可視化ツールがある。

宇宙天気予報シミュレーションでは大量のシミュレーションデータが生産される。そのデータを解析するために、数値データを画像化するいわゆる可視化のプロセスは必要不可欠である [2]。現在、シミュレーションデータの可視化を目的とした “可視化ソフト” と呼ばれる市販あるいは無償のソフトウェアが手に入る。しかしそれらを宇宙天気予報プロジェクトの標準可視化ツールとして採用することはできない。その理由は以下の通りである。

- VTK[3] に代表される無償の可視化ソフトは、ユーザーインターフェースが貧弱で使いにくい。
- 洗練されたユーザーインターフェースを備えた AVS/Express 等の市販ソフトは, 高価であるだけでなく、宇宙天気予報プロジェクトには不要な機能が多すぎるためかえって使いにくい。

宇宙天気予報プロジェクトを構成する様々なシミュレーションコンポーネントの中で最も重要な中心部分は太陽コロナシミュレーションである。そのデータ可視化で必要とされるのは主に

- ユーザーが指定する任意の初期点からの磁力線の計算と表示
- ユーザーが指定する任意の位置での断面表示
- 可視化オブジェクトの回転、移動、拡大縮小機能

である。

ここで必要とされている可視化ツールには通常の可視化ソフトウェアには見られない特徴が二つある。一つは高速性、もう一つはソースコードの提供である。

宇宙天気予報プロジェクトのシミュレーションは出力データサイズが大きく、一つのシミュレーションデータが 1GB のオーダーに達することも珍しくない。このような大規模なデータをリアルタイムで可視化処理することが求められている。ここでリアルタイムとは、可視化パラメーターの視点情報を PC 画面上で指定 · 変更したときに瞬時にその変更が可視化画像に反映されることや、磁力線の出発点をマウス等で指定したときに瞬時に計算し、磁力線オブジェクトを表示することなどである。

ソースコードの提供は宇宙天気予報プロジェクトの参加研究者から特に強い要請のある点である。彼らシミュレーション研究者は、可視化ソフトウェアのソースコードを自分の手で自由に変更 · 改訂したいという希望を持っている。研究の遂行上生じる様々な可視化機能への新たな必要性に対して素早く対応するためである。

ここで特別な注意となるのは、宇宙天気モデリング共同プロジェクトに参加している研究者は Fortran 言語を “母語” としているので通常可視化ソフトウェアで用いられる C/C++ 言語にはなじみが少ないという点がある。したがって提供すべき可視化ソースコードは Fortran 言語 [4] で書かれている必要がある。

宇宙天気モデリング共同プロジェクトでは太陽のコロナのシミュレーションにおいて、インヤン（Yin-Yang）格子と呼ばれる新しい格子を計算格子として採用している。Yin-Yang 格子は陰山 [5] によって考案された球面上のキメラ格子の一種である。大野と陰山 [6] は、Yin-Yang 格子上で定義されたデータを Yin-Yang 格子上でそのまま可視化するプログラムを開発した。しかしこのプログラムはレイ · キャスティング [7] という計算コストのかかるレンダリング方式なので、すばやい応答が要求される宇宙天気予報プロジェクトの可視化には向かない。

Yin-Yang 格子で定義される太陽コロナデータは磁気流体力学（Magnetohydrodynamics, MHD）によって計算されている。磁気流体力学とは電導性流体の流体力学である。電導性流体の運動は磁場の変化と電流を生じ、その結果生じる力（ローレンツ力）が流体自身の運動を変化させるので、普通の流体よりも複雑である。磁気流体力学のシミュレーションでは磁場、速度場、圧力場、温度場等が計算結果として出力される。太陽コロナ中

では磁場エネルギーが運動エネルギーより圧倒的に強いため、太陽コロナのシミュレーションデータの解析では磁場の可視化が最も重要である。磁場は三次元のソレノイダルな（発散のない）ベクトル場であり、その可視化には磁力線表示が有効である。

本研究の目的は Yin-Yang 格子上で定義された太陽コロナデータを対話的に可視化するプログラムを Fortran 言語で開発し、宇宙天気モデリング共同プロジェクトに提供することである。このプログラムを SV4 (Spherical data Visualizer in Fortran) と名付けた。

2 Yin-Yang 格子

宇宙天気モデリング共同プロジェクトでは Yin-Yang 格子が用いられており、本研究の目的はこの格子上で定義されたデータを可視化することである。そこでこの章では Yin-Yang 格子について解説する。

コンピューターシミュレーションでは場の量を離散的に扱う。計算機シミュレーションでは解くべき数値モデル（太陽コロナシミュレーションの場合は MHD 方程式）を離散化して解く。離散化の方法は有限差分法、有限体積法、有限要素法、境界要素法、スペクトル法等、数多く存在する。その中で、宇宙天気モデリング共同プロジェクトでは有限差分法を採用している。有限差分法とは、微分方程式で定式化された問題を計算機で解くにあたり、微分を差分で近似する方法である。そのために空間を順序付けされた格子点で離散化する。格子点全体を計算格子と呼ぶ。計算格子のデザインはシミュレーションの精度と速度を決める重要な要因である。計算結果としての数値データは格子点上で定義される。

2.1 球座標

三次元領域を対象とした計算機シミュレーションをする際には、一点（原点）で直交する 3 本の直線を座標軸にしたカーテシアン座標を用いるのが一般的であるが、今回のシミュレーションのように太陽の表面などの球状の領域を扱うモデルは球座標を用いるのが便利である。球座標とは半径 r、余緯度 θ、経度 ϕ により位置を定義する座標系である。

球座標に基いて空間を離散化した球座標格子（緯度経度格子）が球状領域を対象とした計算機シミュレーションではしばしば用いられているが、この球座標格子には余緯度が 0 度もしくは 180 度の点（即ち北極点と南極点）の近傍において格子点の分布が非常に密になってしまうという問題がある (Fig. 1)。この格子間隔の過度な集中のために極の近傍において計算コストが大きくなるなどの問題が生じる。

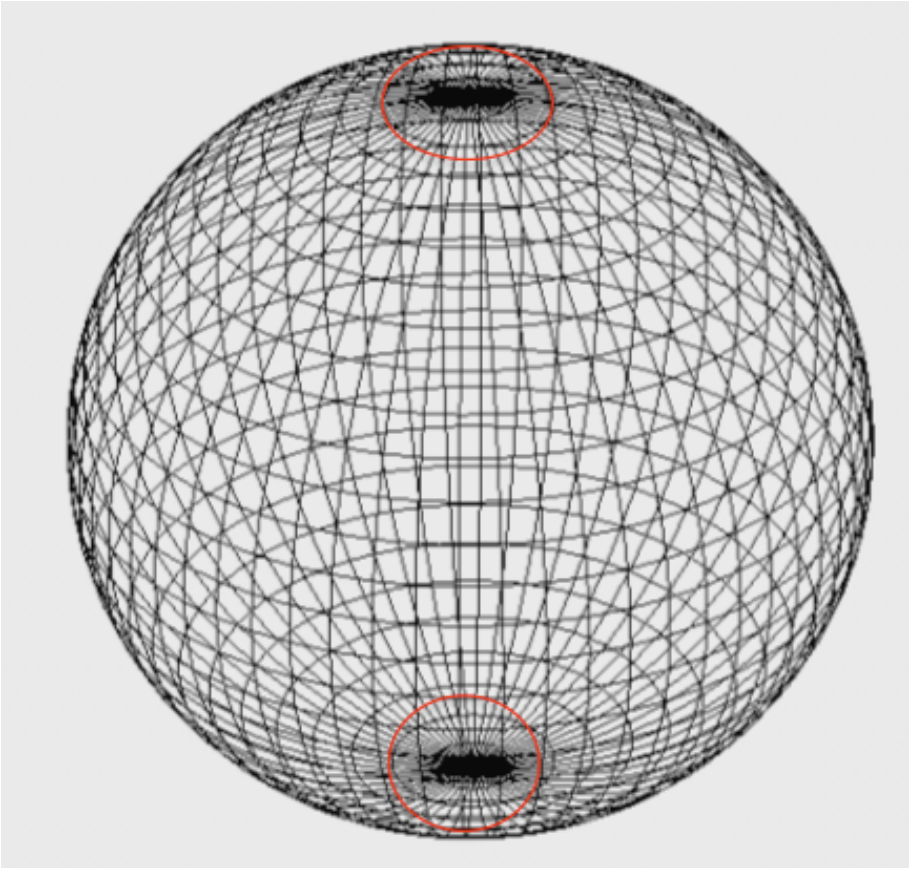

Fig. 1　Spherical coordinate grid.
球座標格子において赤丸で囲った部分が格子間隔が密になってる部分である

2.2　Yin-Yang 格子

2.2.1　Yin-Yang 格子の構造

上記の球座標格子の欠点を解消するために、球座標で格子間隔の分布がほぼ均一となる、余緯度 θ が $\frac{\pi}{4} \leqq \theta \leqq \frac{3\pi}{4}$, 且つ経度 ϕ が $\frac{-3\pi}{4} \leqq \phi \leqq \frac{3\pi}{4}$ の格子部分を抜き出す。そして同じものをもう一つ作り、その二つを組わせて球全体を覆うようにしたのが Yin-Yang 格子である。Yin-Yang 格子を Fig. 2 に示す。

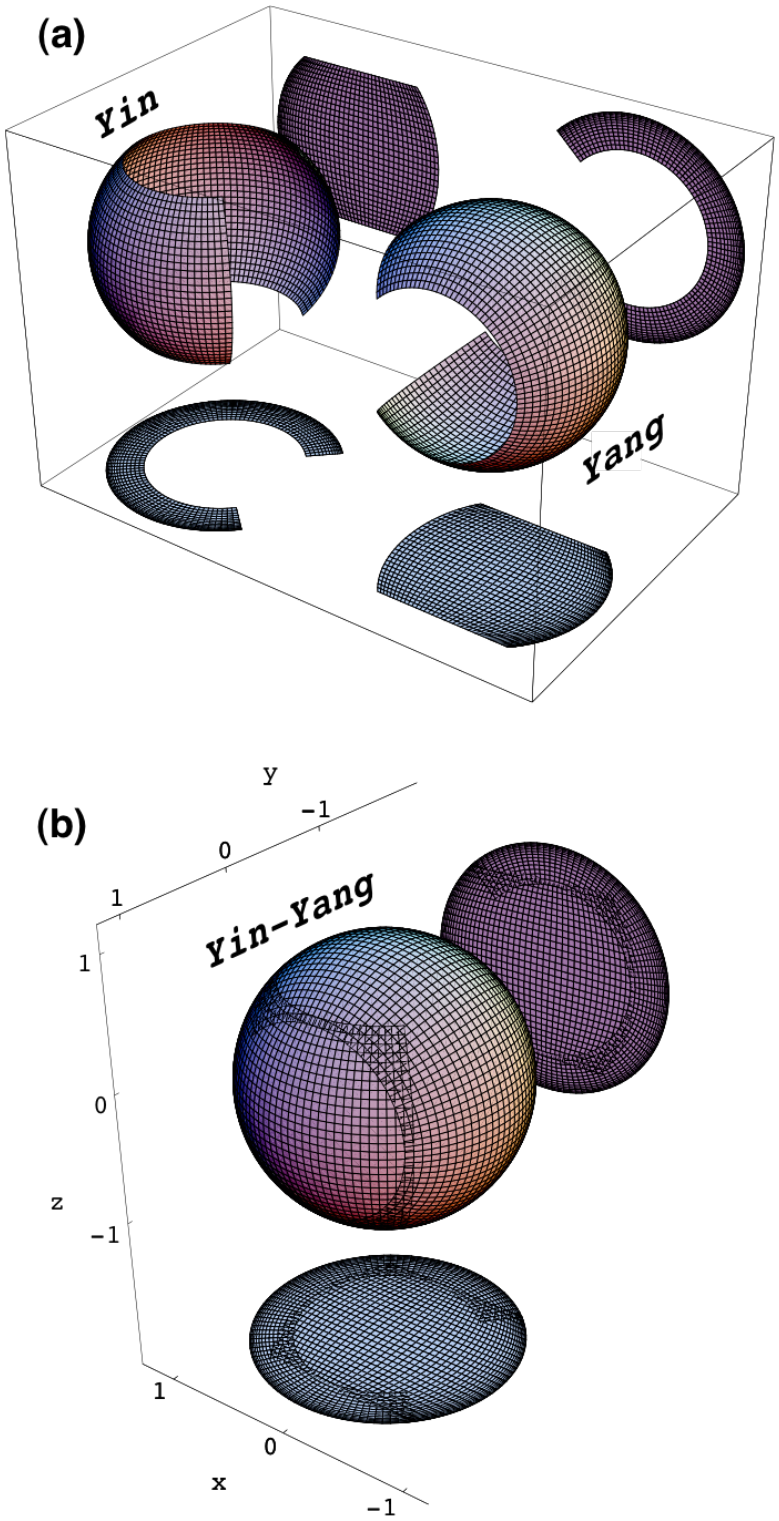

Fig. 2　Yin-Yang grid.
(a) は二つの格子は合同で球座標の $\frac{\pi}{4} \leqq \theta \leqq \frac{3\pi}{4}$ 且つ $\frac{-3\pi}{4} \leqq \phi \leqq \frac{3\pi}{4}$ の領域である。(*b*) 二つの格子を組み合わせると球全体をおおうことができる。これが $Yin - Yang$ 格子である。

2.2.2 Yin-Yang 格子の特徴

Yin-Yang 格子は二つの合同な格子を組み合わせたものである。それぞれを yin と yang と呼ぶ。yin 領域を普通の球座標と同じ r、θ、ϕ を定義した場合、yang 領域は球座標系を回転させた座標系になるので yin 領域と yang 領域では θ と ϕ の定義が違う。半径 r については yin 領域、yang 領域どちらも同じである。

yin 領域の θ は正の z 軸上が 0 度、そこから xy 平面におろす方向が正の向きであり、xy 平面上が 90 度、負の z 軸上が 180 度である。yin 領域の ϕ は xy 平面上の角度であり正の x 軸上が 0 度、そこから正の y 軸へ向かう方向が正である。正の y 軸上で ϕ が 90 度、負の y 軸上が-90 度、負の x 軸上が 180 度もしくは-180 度である。それに対して yang 領域の θ は負の y 軸上が 0 度、そこから xz 平面におろす方向が正の向きである。xz 平面上が 90 度、正の y 軸上が 180 度である。yang 領域の ϕ は xz 平面上の角度であり負の x 軸上が 0 度、そこから負の z 軸へ向かう方向が正である。負の z 軸上で ϕ が 90 度、正の z 軸上が-90 度、正の x 軸上が 180 度もしくは-180 度である。これらの関係を Fig. 3、Fig. 4 の xy 平面の図と xz 平面の図にて示す。また yin 領域の θ を θ^i、yin 領域の ϕ を ϕ^i、yang 領域の θ を θ^a、yang 領域の ϕ を ϕ^a とする。

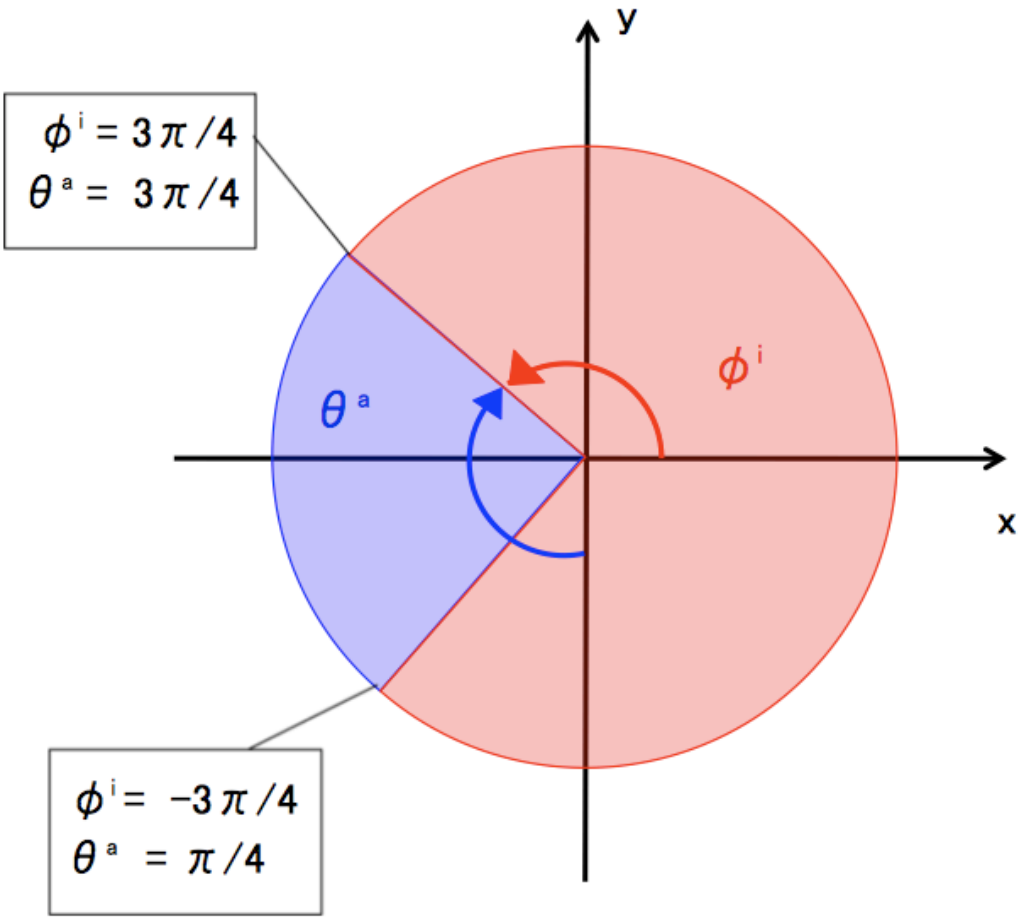

Fig. 3　Definition of θ^a and ϕ^i.
赤は $\frac{-3\pi}{4} \leqq \phi^i \leqq \frac{3\pi}{4}$ の領域で、青は $\frac{\pi}{4} \leqq \theta^a \leqq \frac{3\pi}{4}$ の領域である。また赤い矢印は ϕ^i の 0 度から正方向に向かう矢印であり、青い矢印は θ^a の 0 度から正方向に向かう矢印である。

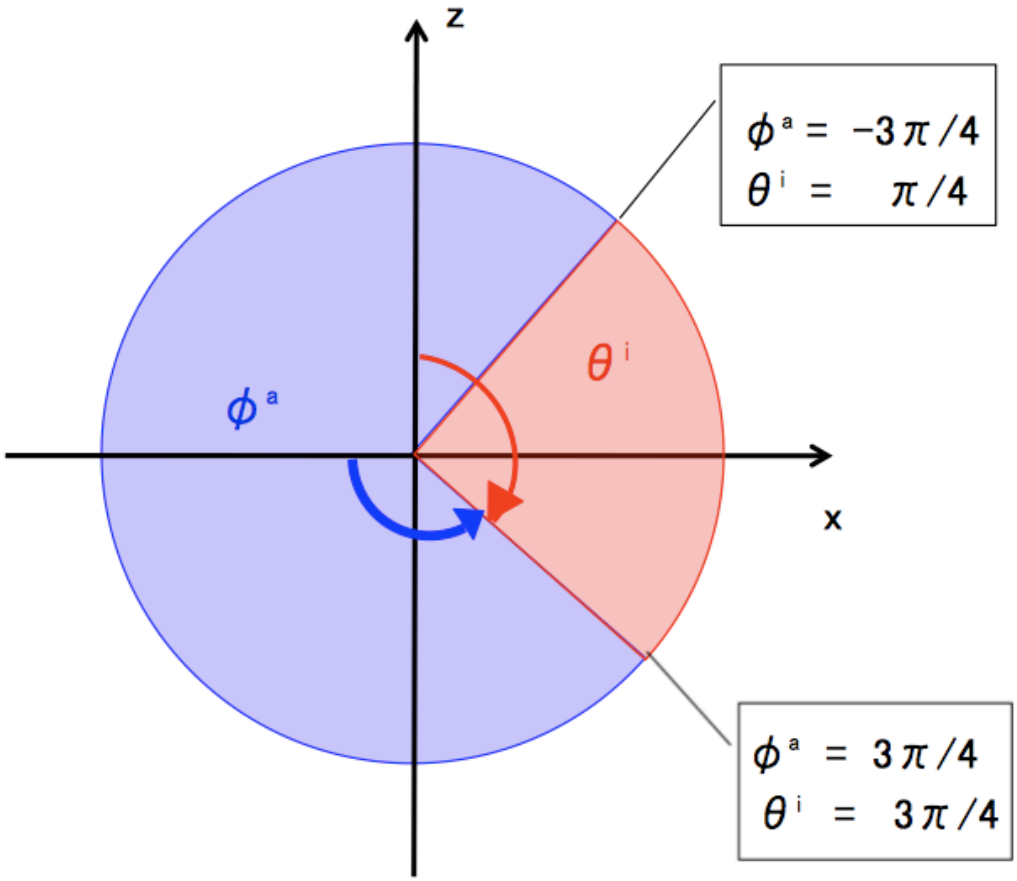

Fig. 4　Definition of θ^i and ϕ^a.
赤は $\frac{\pi}{4} \leqq \theta^i \leqq \frac{3\pi}{4}$ の領域で、青は $\frac{-3\pi}{4} \leqq \phi^a \leqq \frac{3\pi}{4}$ の領域をである。また赤い矢印は θ^i の 0 度から正方向に向かう矢印であり、青い矢印は ϕ^a の 0 度から正方向に向かう矢印である。

3　データ構造

この章では前章で述べた Yin-Yang 格子上のデータを可視化するために本研究で開発した SV4 プログラムで用いた基本データ構造とその利用法を説明する。

3.1　Yin-Yang データの構造

半径 r、余緯度 θ、経度 ϕ のそれぞれ格子間隔 Δr、$\Delta\theta$、$\Delta\phi$ を以下の式で定義する。

$$\Delta r = \frac{r_{max} - r_{min}}{N_r - 1} \tag{1}$$

$$\Delta\theta = \frac{\theta_{max} - \theta_{min}}{N_\theta - 1} = \frac{\frac{3\pi}{4} - \frac{\pi}{4}}{N_\theta - 1} = \frac{\frac{\pi}{2}}{N_\theta - 1} \tag{2}$$

$$\Delta\phi = \frac{\phi_{max} - \phi_{min}}{N_\phi - 1} = \frac{\frac{3\pi}{4} - \left(-\frac{3\pi}{4}\right)}{N_\phi - 1} = \frac{\frac{3\pi}{2}}{N_\phi - 1} \tag{3}$$

ここで N_r、N_θ、N_ϕ はそれぞれ半径 r、余緯度 θ、経度 ϕ 方向の格子点数である。それぞれの N を大きくとればとるほど格子の目は細かくなるのでより精密なシミュレーションが可能になるが、逆に計算コストは大きくなる。次に r_i、θ_j、ϕ_k を以下の式で定義する。

$$r_i = r_{min} + \Delta r * i \qquad (0 \leqq i < N_r) \tag{4}$$

$$\theta_j = \theta_{min} + \Delta\theta * j \qquad (0 \leqq j < N_\theta) \tag{5}$$

$$\phi_k = \phi_{min} + \Delta\phi * k \qquad (0 \leqq k < N_\phi) \tag{6}$$

Yin-Yang 格子は yin 領域と yang 領域に分かれているので、識別子 l を以下のように定義する。

$$l = \begin{cases} 0 & \text{(yin 領域のとき)} \\ 1 & \text{(yang 領域のとき)} \end{cases} \tag{7}$$

これら r_i、θ_j、ϕ_k、l を使って Yin-Yang 格子のすべての格子点を定義することができる。格子点 (i, j, k, l) は識別子 l の領域における半径 r_i、余緯度 θ_j、経度 ϕ_k の格子点である。

シミュレーションで得られた各格子点の位置上での様々な物理量がそれぞれ個別の四次元配列として定義されている。この四次元配列データを読み込んで様々な可視化処理を行うのが SV4 プログラムである。

3.2 Yin-Yang データの利用

3.2.1 座標変換

本節ではカーテシアン座標と Yin-Yang 座標との関係についてまとめる。座標変換の式は yin と yang では形が異なることに注意が必要である。

まず yin 領域において半径 r、余緯度 θ、経度 ϕ からカーテシアン座標 x、y、z に変換する式を下に示す。

$$x = r \sin\theta^i \cos\phi^i \tag{8}$$

$$y = r \sin\theta^i \sin\phi^i \tag{9}$$

$$z = r \cos\theta^i \tag{10}$$

これは yin 領域では半径 r、余緯度 θ、経度 ϕ の定義は球座標と同じなので球座標からカーテシアン座標への変換式と同じである。

一方 yang 領域では余緯度 θ、経度 ϕ の定義が球座標と異なるので変換式も異なる。Fig. 3、Fig. 4 の定義により yang 領域において半径 r、余緯度 θ、経度 ϕ からカーテシアン座標 x、y、z に変換する式は下記のようになる。

$$x = -r \sin\theta^a \cos\phi^a \tag{11}$$

$$y = -r \cos\theta^a \tag{12}$$

$$z = -r \sin\theta^a \sin\phi^a \tag{13}$$

次にカーテシアン座標 x、y、z から yin 領域の半径 r、余緯度 θ、経度 ϕ に変換する式を以下に示す

$$r = \sqrt{x^2 + y^2 + z^2} \tag{14}$$

$$\theta^i = \tan^{-1} \frac{\sqrt{x^2 + y^2}}{z} \tag{15}$$

$$\phi^i = \tan^{-1} \frac{y}{x} \tag{16}$$

式 (15)、式 (16) では $\tan^{-1}$ 関数の定義に注意する必要がある。$\tan^{-1}$ は多価函数であり、一般には $\frac{-\pi}{2} < \tan^{-1}\alpha < \frac{\pi}{2}$ を主値とした定義が採用される場合が多い。しかし Yin-Yang 格子法では余緯度 θ の範囲は $\frac{\pi}{4} \leqq \theta \leqq \frac{3\pi}{4}$ であり、経度 ϕ の範囲は $\frac{-3\pi}{4} \leqq \phi \leqq \frac{3\pi}{4}$ であるので上記の定義の $\tan^{-1}$ では不便である。そこで $\tan^{-1}\frac{y}{x}$ に対して以下の定義を採用する。

- $\tan^{-1}\frac{y}{x}$ の範囲は $-\pi < \tan^{-1}\frac{y}{x} \leqq \pi$
- $y > 0$ のとき $\tan^{-1}\frac{y}{x}$ の値は正
- $y < 0$ のとき $\tan^{-1}\frac{y}{x}$ の値は負
- $y = 0$ で $x > 0$ のとき $\tan^{-1}\frac{y}{x}$ の値は 0
- $y = 0$ で $x < 0$ のとき $\tan^{-1}\frac{y}{x}$ の値は π
- $x = 0$ のとき $\tan^{-1}\frac{y}{x}$ の絶対値は $\frac{\pi}{2}$

Fortran90 言語ではこの拡張された $\tan^{-1}$ が組み込み関数 ATAN2 として用意されており、SV4 もこの ATAN2 を使って実装している。なお、この論文中に登場する $\tan^{-1}$ は上記の定義で拡張された $\tan^{-1}$ とする。

カーテシアン座標 x、y、z から yang 領域の半径 r、余緯度 θ、経度 ϕ に変換する式を以下に示す 。

$$r = \sqrt{x^2 + y^2 + z^2} \tag{17}$$

$$\theta^a = \tan^{-1} \frac{\sqrt{x^2 + z^2}}{-y} \tag{18}$$

$$\phi^a = \tan^{-1} \frac{-z}{-x} \tag{19}$$

Fig. 3、Fig. 4 の定義の通りに yang 領域の余緯度 θ、経度 ϕ を求めるためには $\tan^{-1}$ の分子分母の符号をそれぞれ式 (18)、式 (19) のようにしなければならない。なぜなら ATAN2 は分子の符号と分母の符号をもとに角度を一意的に定めるからである。よってプログラム中では式 (18) の分母 y のマイナスを分子に持ってきたり、式 (19) のマイナスを約分することはできないことに注意しなければならない。

3.2.2 データ補間

カーテシアン座標で点 $\mathrm{P}(x, y, z)$ におけるデータの値がほしいとき、P の座標値 (x, y, z) から式 (14)、式 (15)、式 (16) を使い yin 領域の半径 r、余緯度 θ、経度 ϕ に変換する。求めた θ と ϕ が $\frac{\pi}{4} \leqq \theta \leqq \frac{3\pi}{4}$ 且つ $\frac{-3\pi}{4} \leqq \phi \leqq \frac{3\pi}{4}$ だった場合、カーテシアン座標 x、y、z は yin 領域にあるということなので添字 l を $l = 0$ とする。そうでなかった場合カーテシアン座標 x、y、z は yang 領域にあるということなので l を $l = 1$ とし、式 (18)、式 (19) を使い θ、経度 ϕ を計算しなおす。これでカーテシアン座標 x、y、z が Yin-Yang 格子のどちらの領域にあるかを判別し、またその領域における半径 r、余緯度 θ、経度 ϕ に変換することができる。次にこうして求めた r、θ、ϕ から以下の条件に当てはまる r_i、θ_j、ϕ_k を見つける。

$$r_i \leqq r < r_{i+1} \ , \ \theta_i \leqq \theta < \theta_{i+1} \ , \ \phi_i \leqq \phi < \phi_{i+1} \tag{20}$$

点 P は、yin 格子または yang 格子中の八つの 格子点 (i, j, k, l)、$(i+1, j, k, l)$、$(i, j+1, k, l)$、$(i+1, j+1, k, l)$、$(i, j, k+1, l)$、$(i+1, j, k+1, l)$、$(i, j+1, k+1, l)$、$(i+1, j+1, k+1, l)$ に囲まれている。そこで点 P とそれを取り囲む八つの格子点の相対位置に基づいて重みをつけた補間を行う。データの値がほしい点に近い格子点ほど重みが大きい。本研究では以下のような三次元線形補間を採用した。

$$f_{r1} = \frac{r_{i+1} - r}{r_{i+1} - r_i} \ , \ f_{r2} = 1 - f_{r1} \tag{21}$$

$$f_{\theta 1} = \frac{\theta_{i+1} - \theta}{\theta_{i+1} - \theta_i} \ , \ f_{\theta 2} = 1 - f_{\theta 1} \tag{22}$$

$$f_{\phi 1} = \frac{\phi_{i+1} - \phi}{\phi_{i+1} - \phi_i} \ , \ f_{\phi 2} = 1 - f_{\phi 1} \tag{23}$$

$$w_1 = f_{r1} * f_{\theta 1} * f_{\phi 1} \ , \ w_2 = f_{r2} * f_{\theta 1} * f_{\phi 1} \tag{24}$$

$$w_3 = f_{r1} * f_{\theta 2} * f_{\phi 1} \ , \ w_4 = f_{r2} * f_{\theta 2} * f_{\phi 1} \tag{25}$$

$$w_5 = f_{r1} * f_{\theta 1} * f_{\phi 2} \ , \ w_6 = f_{r2} * f_{\theta 1} * f_{\phi 2} \tag{26}$$

$$w_7 = f_{r1} * f_{\theta 2} * f_{\phi 2} \ , \ w_8 = f_{r2} * f_{\theta 2} * f_{\phi 2} \tag{27}$$

点 P における値を $value$ とし 格子点 (i,j,k,l) のデータを $data(i,j,k,l)$ とすると $value$ は以下の式で書ける。

$$\begin{aligned} value = {} & w_1 * data(i,j,k,l) + w_2 * data(i+1,j,k,l) + w_3 * data(i,j+1,k,l) \\ & + w_4 * data(i+1,j+1,k,l) + w_5 * data(i,j,k+1,l) + w_6 * data(i+1,j,k+1,l) \\ & + w_7 * data(i,j+1,k+1,l) + w_8 * data(i+1,j+1,k+1,l) \end{aligned} \tag{28}$$

以上のようにして Yin-Yan 格子で定義されたデータ空間を利用することができる。SV4 中では定義されたデータそれぞれに対して yy_xx_load（xx は取り出すデータの名前）という関数を実装した。例えば yy_bx_load は引数 x、y、z が与えられたとき、そのカーテシアン座標位置における磁場の x 方向の値を返す関数である。

3.2.3 データマップ

yy_xx_load（xx は取り出すデータの名前）を応用することで Yin-Yang 格子で定義されたデータをカーテシアン座標格子で定義されたデータに変換することができる。例えばカーテシアン座標格子点すべてにおいて yy_bx_load を使うことですべて格子点の磁場の x 方向の値が定義でき、それはまさしくカーテシアン座標格子で定義された磁場の x 方向データである。その手法により Yin-Yang 格子上で定義されたシミュレーションデータをカーテシアン格子にマップすることでカーテシアン格子上で定義されたデータを可視化する一般的な可視化ソフトウェアを利用することが可能である。実際そのようなデータマッププログラム YdToCd を作り、カーテシアン格子上で可視化できることを確認した。ただし Yin-Yang 格子からカーテシアン格子にマップしてから可視化する方法はマップの処理に時間がかかるため高速な処理が要求される宇宙天気予報プロジェクトの可視化という目的には合わない。そこで本研究では Yin-Yang 格子上で定義されたデータを Yin-Yang 格子上で直接可視化するソフトウェアを開発した。

4　可視化プログラムの基本ツール

コンピューターグラフィックスの基本 API として Open Graphics Library（OpenGL）[8] が有名である。OpenGL では三次元空間上に点や点列を指定し、その座標値から線や多角形（ポリゴン）を構成することでオブジェクトを作り出す。オブジェクトの法線ベクトルを指定することで光源により照らされたオブジェクトの陰影も自動的に処理される。各オジェクトの光に対するさまざまな反射属性を設定することで色や質感を細かく決めることができる。光源も、どのような光でどの位置から照らしている等、細かく設定することができる。またどの視点からオブジェクトを見ているかというカメラに関する指定も OpenGL で行う。このように設定されたオブジェクトの位置、属性、光源、カメラの設定により、どのような画像が出来るかが自動的に計算される。通常その計算には専用のハードウェア (Graphics Processing Unit, GPU) が使われる。

OpenGL はウインドウマネージメントやマウスイベントの取得等、ユーザーインターフェースの機能は持たない。そこで、OpenGL Utility Toolkit（GLUT）というライブラリがよく用いられる。SV4 でも GLUT を使っている。GLUT を使えばキーボードやマウスによりオブジェクトを回転させるようなことが可能になる。またメニューを作ることもでき、例えばマウスやキーボードの役割をメニューにより変えるといったことも可能である。このようなユーザーインターフェースに関するさまざま関数が GLUT には用意されている。SV4 の主な目的として対話性のある 3D グラフィックスであるので GLUT は必要不可欠である。

しかし GLUT は C 言語のライブラリとして実装されているので Fortran 言語では使えない。そこで SV4 では f90gl[9] を採用した。f90gl は GLUT の機能を Fortran 言語のモジュールとして提供しており、このモジュールを使用することで Fortran 言語でも GLUT が使用可能になる。

5　磁力線の可視化

この章では太陽コロナシミュレーションの解析において重要な役割を担う磁力線の可視化について説明する。

5.1　磁力線の計算

磁力線は以下の式 (29) で定義される。

$$\frac{d\mathbf{x}(s)}{ds} = \frac{\mathbf{B}(\mathbf{x}(s))}{\|\mathbf{B}\|} \tag{29}$$

ここで $\mathbf{x}(s)$ は磁力線に沿った長さ s における空間中の位置ベクトルを表す。$\mathbf{B}(\mathbf{x}(s))$ は位置 $\mathbf{x}(s)$ における磁場である。$\|\mathbf{B}\|$ は磁場ベクトルの絶対値を表す。式 (29) は微少量 Δs を使えば以下のように書くことができる。

$$\Delta x = \frac{\mathbf{B_x}(\mathbf{x}(s))}{\|\mathbf{B}\|}\Delta s, \tag{30}$$

$$\Delta y = \frac{\mathbf{B_y}(\mathbf{x}(s))}{\|\mathbf{B}\|}\Delta s, \tag{31}$$

$$\Delta z = \frac{\mathbf{B_z}(\mathbf{x}(s))}{\|\mathbf{B}\|}\Delta s. \tag{32}$$

これらの式が意味するのは以下の通りである。シミュレーション空間内に描きたい磁力線の初期点を決める。その初期点における磁場の x、y、z 成分を 3 章の補間により Yin-Yang 格子上の磁場データから導き、次にそのベクトルを絶対値が 1 になるよう規格化する。磁場ベクトル方向に Δs だけ進んだ点を次の点とする。その操作を次の条件が満足されるまで続ける。その条件とは、磁場が極端に小さな値になる、シミュレーション空間から外れる、もしくは点の数があらかじめ設定した最大数になるまで、というものである。こうして得られた点列を繋いだものがシミュレーション空間内における磁力線である。

ここで気をつけなければならないのは Δs の大きさである。磁力線を正確に描くには Δs は格子間隔より小さくとるべきである。そうでなければ磁力線追跡の誤差が大きくなってしまう。その様子を Δs が適切な場合の磁力線（緑)、Δs が大きい場合の磁力線

（赤）として Fig. 5 に示す。ここでは簡単ためシミュレーション空間が二次元として磁力線を描いている。

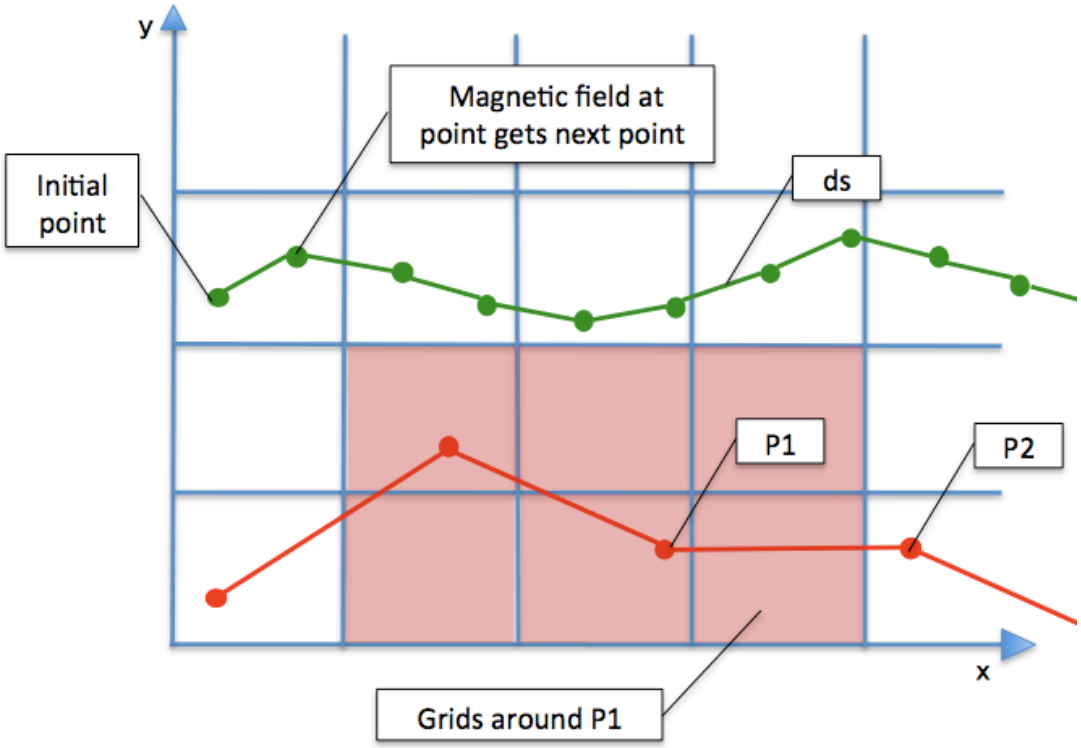

Fig. 5　Magnetic field line tracing.
緑が ds が適切な場合の磁力線、赤は ds が大きい場合の磁力線、それぞれの一番左の点が初期点である。赤く塗りつぶされている格子は P1 の近傍の格子である。ds が格子間隔より大きいため P1 の次の点である P2 が P1 の近傍の格子の外に出てしまっている。

上では磁力線の定義式である微分方程式 (29) を数値的に解く方法として 1 次オイラー法による数値積分法に基づいて説明した。しかしながらこの方法は $O[(\Delta s)]$ の精度しかない。SV4 では精度を上げるため 4 次精度のルンゲ＝クッタ法で式 (29) を数値積分している。つまり精度は $O[(\Delta s^4)]$ である。なお、Δs はシミュレーションデータの定義された Yin-Yang 格子の全格子間隔のなかで最も短い格子間隔の 20 %に設定した。

5.2 磁力線の可視化

前節で述べた磁力線の点列の座標から OpenGL で磁力線を描く方法をここでは説明する。初期点から次の点に向かうベクトルを $\mathbf{V}$ とする、そのとき $\mathbf{V}$ を中心軸とした円柱を描きたい。まず $\mathbf{V}$ とは異なる任意のベクトルをとり、そのベクトルと $\mathbf{V}$ との外積を計算し、その方向へ向かう単位ベクトルを $\mathbf{U}$ とする。更に $\mathbf{V}$ と $\mathbf{U}$ の外積単位ベクトルを $\mathbf{W}$ とする。$\mathbf{V}$ と $\mathbf{U}$ と $\mathbf{W}$ は互いに直交する。ここで初期点の位置ベクトルを $\mathbf{P}_1$、次の点の位置ベクトルを $\mathbf{P}_2$ としたとき

$$\mathbf{A}_1(\theta) = \mathbf{P}_1 + r(\sin\theta\mathbf{W} + \cos\theta\mathbf{U}) \quad (0 \leqq \theta < 2\pi) \tag{33}$$

$$\mathbf{A_2}(\theta) = \mathbf{P}_2 + r(\sin\theta\mathbf{W} + \cos\theta\mathbf{U}) \quad (0 \leqq \theta < 2\pi) \tag{34}$$

で決まる $\mathbf{A}_1(\theta)$ と $\mathbf{A}_2(\theta)$ は $\mathbf{V}$ に垂直な面上で $\mathbf{P}_1$ と $\mathbf{P}_2$ を中心とした半径 r の円周上の点である。θ の値により円周上の任意の点をとることができる。$\mathbf{P}_1(\theta)$ の点と $\mathbf{P}_2(\theta)$ の点の θ ごとの座標とその法線ベクトルである $(\sin\theta\mathbf{W} + \cos\theta\mathbf{U})$ を第 4 章で説明した OpenGL の関数に引数として渡す。その関数にそれらの点に結ぶ面を表示させることにより $\mathbf{V}$ を中心軸とした円柱が描くことができる (Fig. 6)。

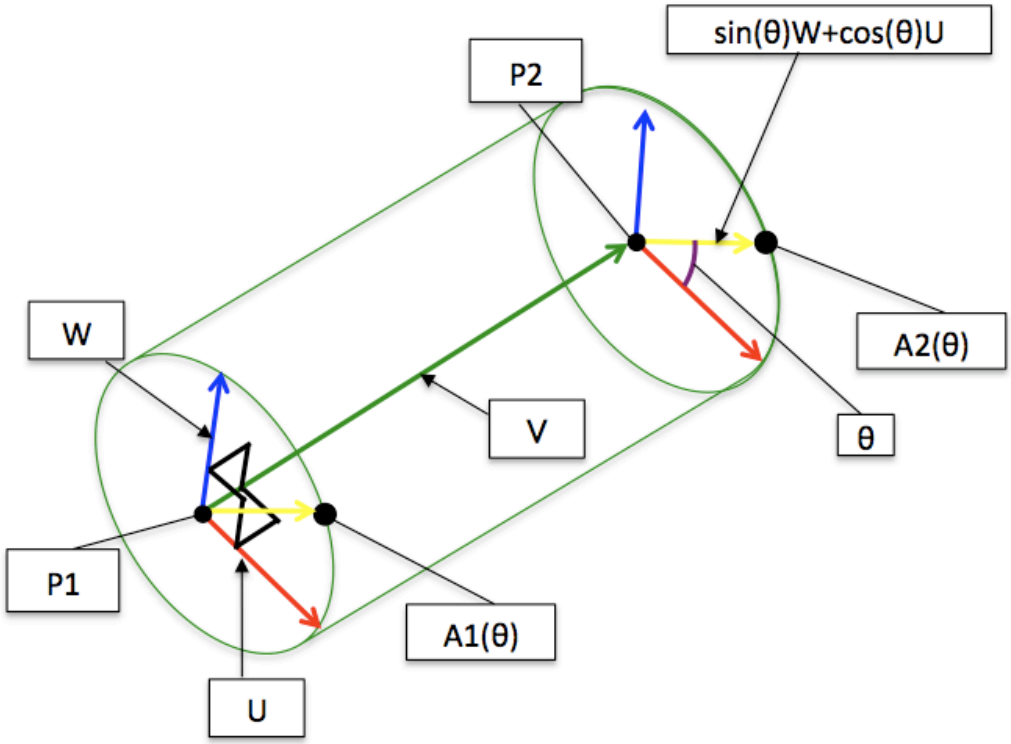

Fig. 6　Cylinder construction for magnetic field line tube.
同じ色の矢印は同じ方向大きさを持つベクトルを意味する

次に $\mathbf{P}_2$ の次の点を $\mathbf{P}_3$ としたとき、$\mathbf{P}_2$ と $\mathbf{P}_3$ において上記と同じ導出で $\mathbf{A}_3(\theta)$ を求める。その $\mathbf{A}_3(\theta)$ と前回の $\mathbf{A}_2(\theta)$ を結ぶ面を表示することでその次の円柱を作る。これを最後の点まで繰り返すことにより磁力線を表示する。SV4 においては磁力線の向きを分かりやすくするため, 最後の点だけあえて周りの点を使わずにそのままの点を使うことで磁力線の最後が円錐となるようにした (Fig. 7)。Fig. 8 に実際に SV4 で表示した磁力線の例を示す。

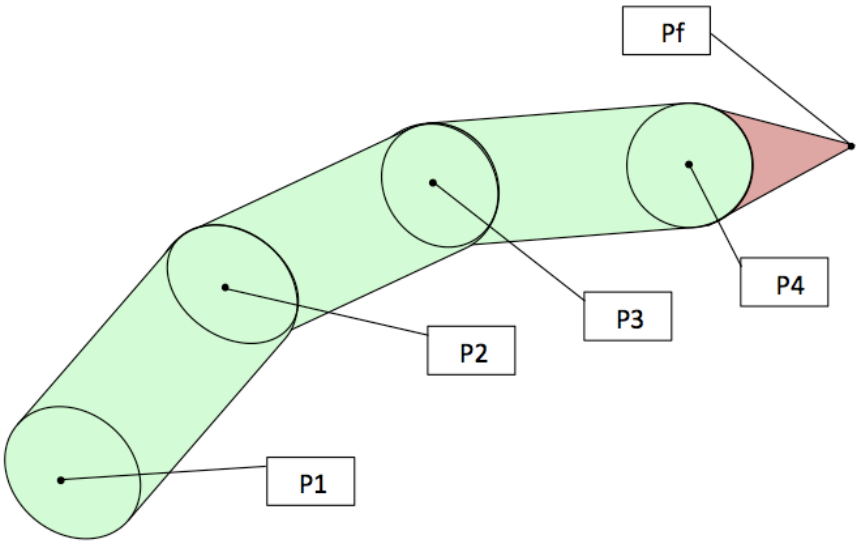

Fig. 7　Magnetic filed line tube.
Pf は磁力線の最後の点である

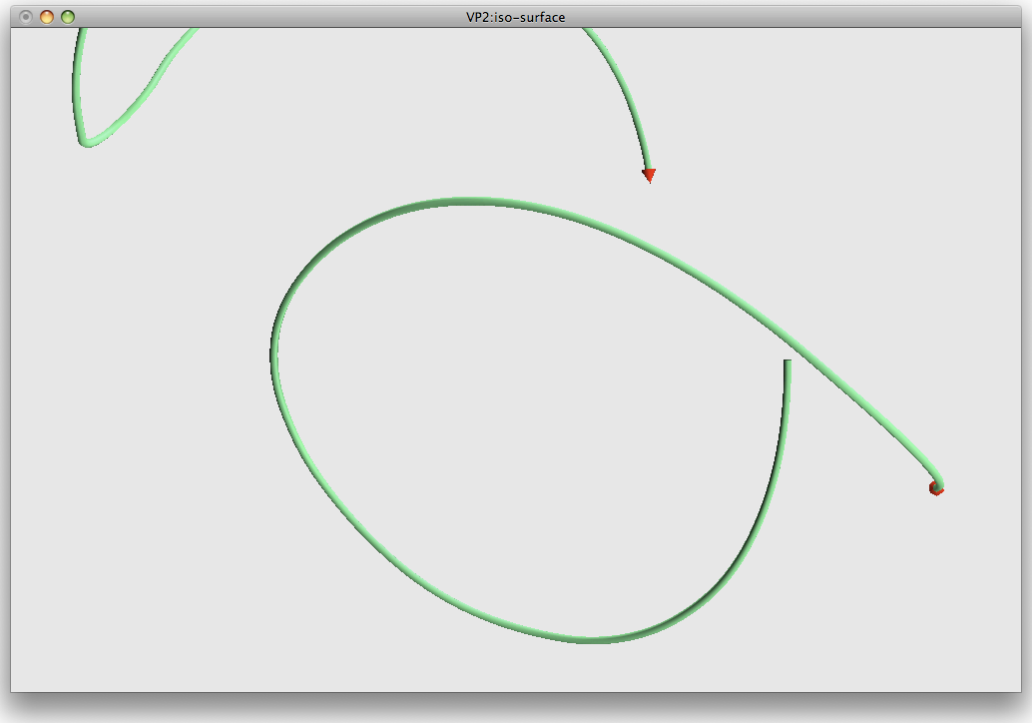

Fig. 8　Magnetic field line tubes.

6　SV4（Yin-Yang データ可視化プログラム）

この章では本研究で開発したプログラム SV4 について説明する。SV4 はグラフィカルユーザーインターフェースにより可視化したオブジェクトを回転、拡大縮小、平行移動することができる。Fig. 9 はフレア発生時の太陽コロナ中の磁力線を Yin-Yang データから可視化した例である。

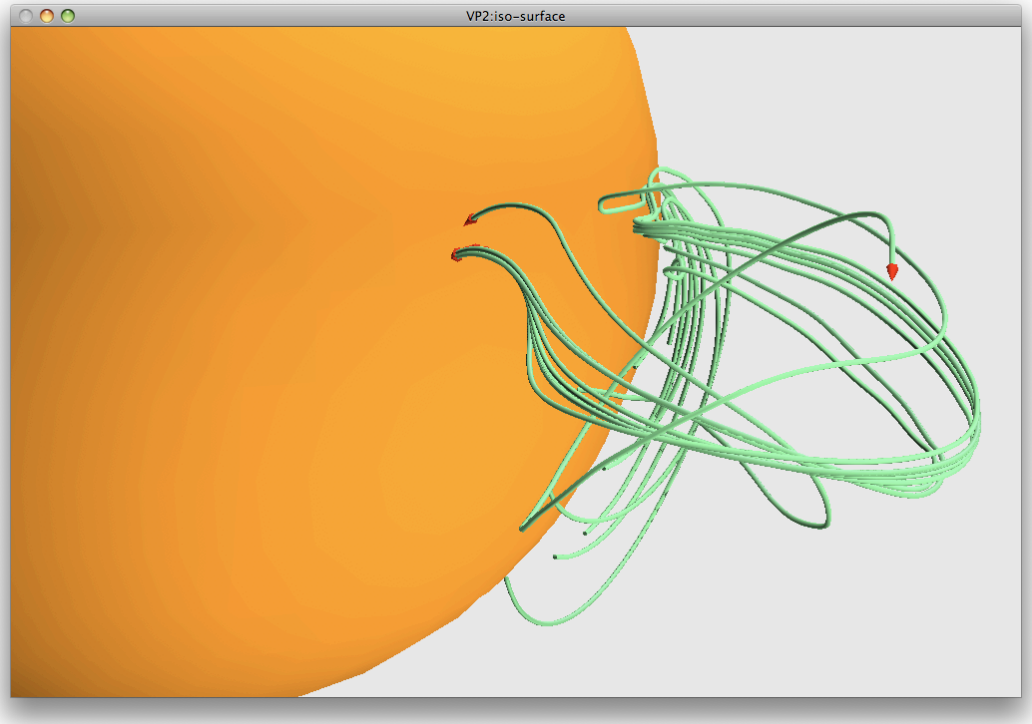

Fig. 9　A snapshot of magnetic field lines of the solar corona.
オレンジ色のオブジェクトが太陽の表面であり、緑のチューブが磁力線である。

以下に SV4 に実装したユーザーインターフェースのデザインについて説明する。マウスの右クリックによりメニューを表示できる。Fig. 10 が SV4 のメニューである。

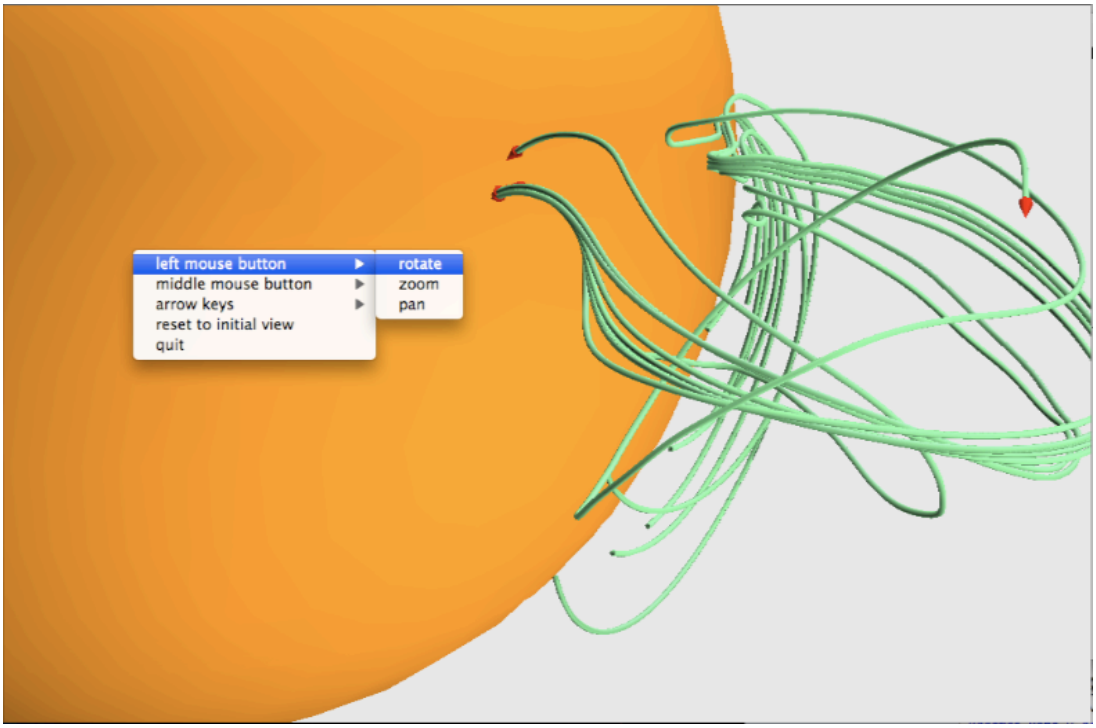

Fig. 10　Menu of SV4
ウインドウで右クリックするとメニューが表示される。

メニューのそれぞれの意味を下に示す。

- left mouse button　マウスの左ボタンの役割設定
  - rotate　オブジェクトの回転操作
  - zoom　オブジェクトの拡大縮小操作
  - pan　オブジェクトの平行移動操作
- right mouse button　マウスの中ボタンの役割設定
  - rotate　オブジェクトの回転操作
  - zoom　オブジェクトの拡大縮小操作
  - pan　オブジェクトの平行移動操作
- arrow keys　キーボードの矢印キーの役割設定
  - rotate　オブジェクトの回転操作
  - zoom　オブジェクトの拡大縮小操作
  - pan　オブジェクトの平行移動操作
- reset initial view　視点を初期状態に戻す
- quit　プログラムの終了

メニューをクリックすることでマウスやキーボードの矢印キーの役割を設定し、その役割通りにマウスやキーボードの矢印キーで操作することできる。

次にマウスによるそれぞれの操作を説明する。マウスの操作はカーソルが対象ウインドウ内にある時にマウスボタンと組合わせて作用するように設計した。ウインドウを xy 平面とみなし、ウインドウの左下隅を原点とし、右方向を x の正方向、上方向を y の正方向とする。

マウスボタンを押し込んだ座標を (x_i, y_i) とし、現在のカーソルの位置を (x_n, y_n) とする。マウスの回転操作は z 軸のまわりに $x_n - x_i$ 度、x 軸のまわりに $y_n - y_i$ 度だけ回転する。マウスの拡大縮小操作は $y_n - y_i$ に比例させた。マウスの平行移動操作はウインドウの x、y 方向それぞれに $(x_n - x_i)$、$(y_n - y_i)$ だけ作用する。

次にキーボードの矢印キーよるそれぞれの操作を説明する。矢印キーの回転操作は左右のキーが z 軸を軸とた回転で、上下のキーが x 軸を軸とした回転である。矢印キーの拡大縮小操作は上キーが拡大であり。下キーが縮小である。矢印キーの平行移動はウインドウの x 方向の移動が左右キーで y 方向の移動が上下キーである。

メニューコマンドの “reset initial view” を押すと視点を初期状態に戻すことができる。初期状態は xy 平面を真上から見下ろす視点となっている。メニューコマンドの “quit” を押すとプログラム終了である。SV4 のメニュー以外の機能としてプログラムを終了して、次に起動するとき同じ視点から始めることができるようにした。

オブジェクトの回転、拡大縮小、平行移動をしている様子を撮った画像を Fig. 11 から 13 に示す。

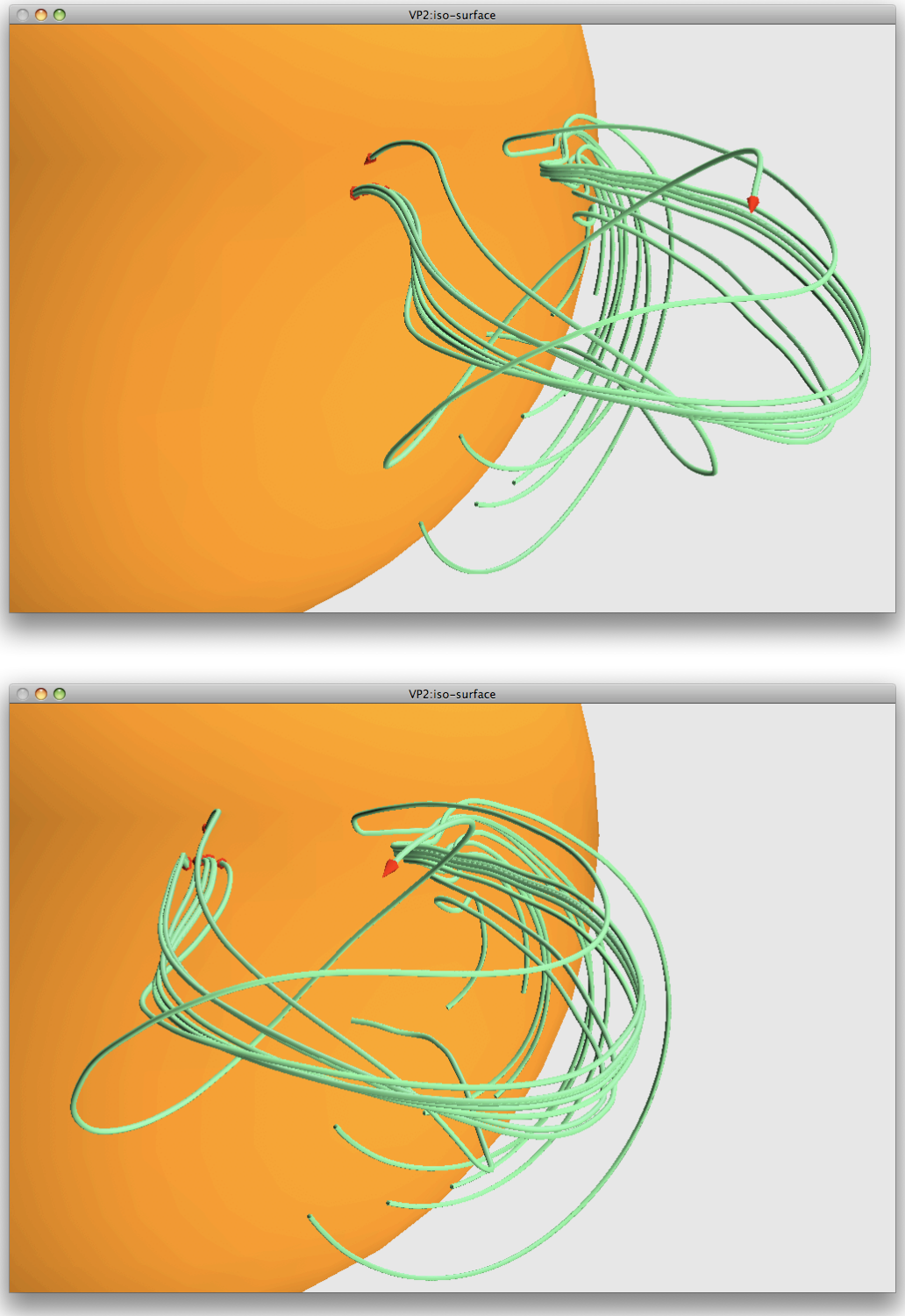

Fig. 11　Rotation of objects in SV4.
上の画像が回転前、下が回転後である。

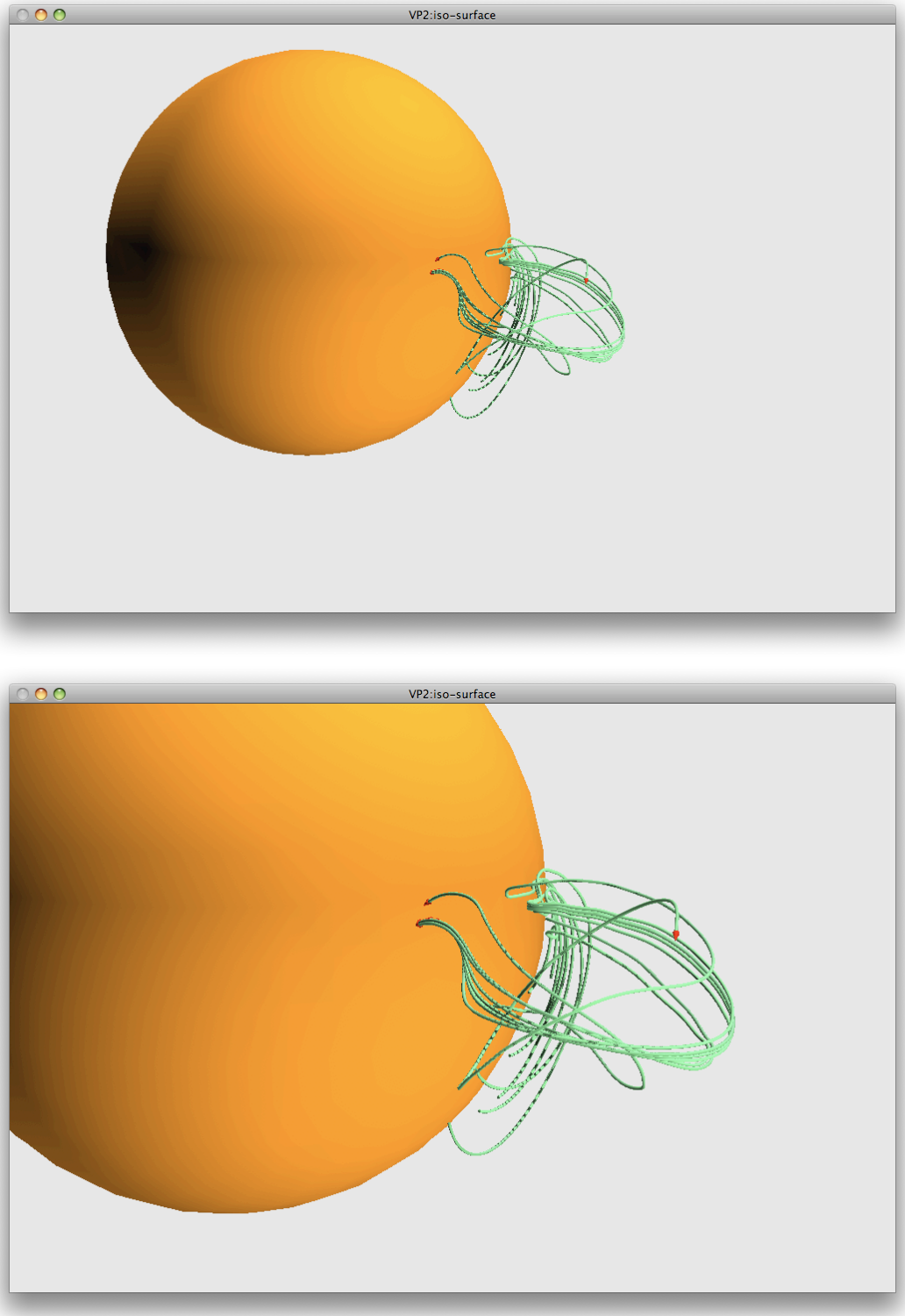

Fig. 12　Zoom in view by SV4.
上の画像が拡大前、下が拡大後である。

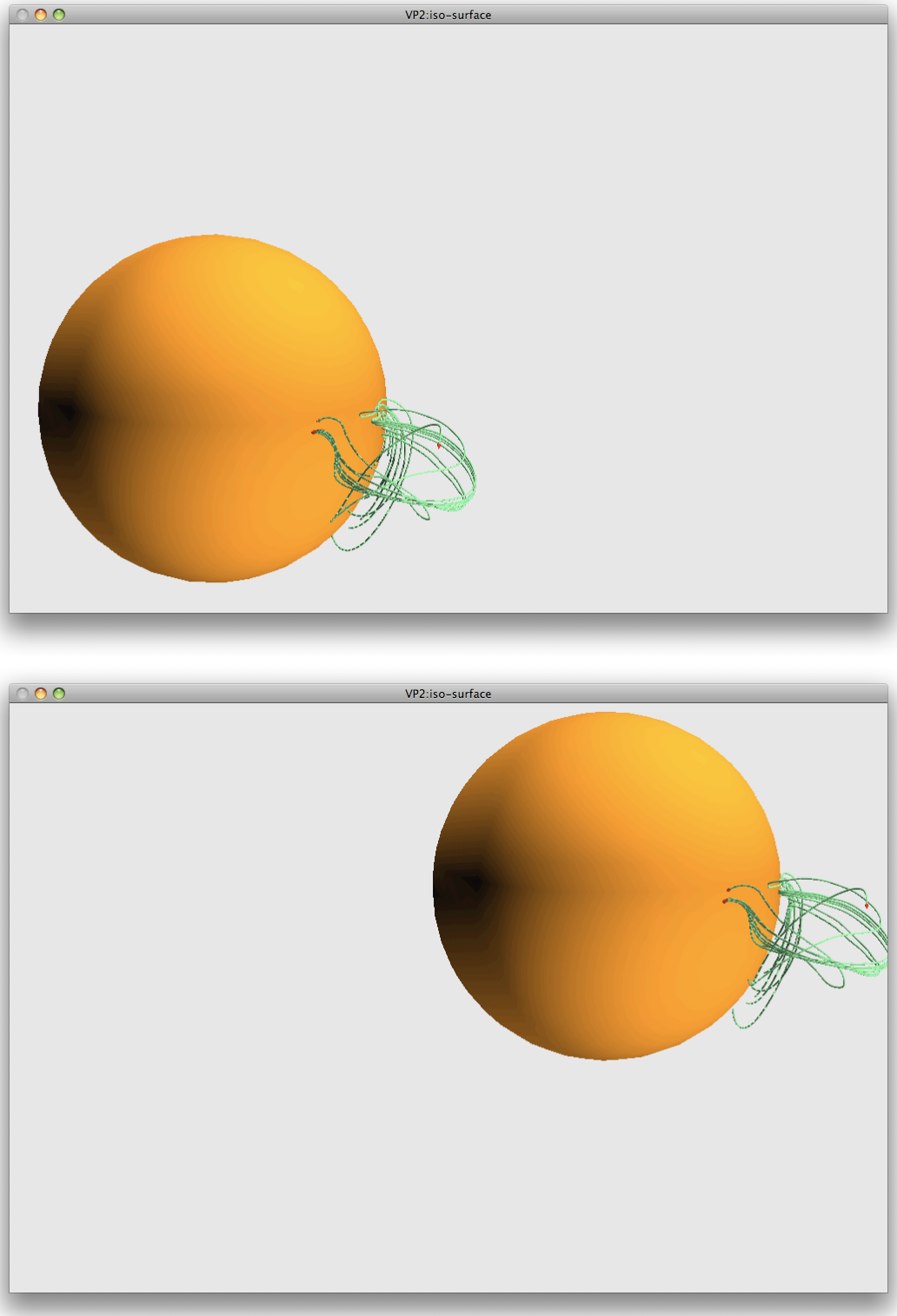

Fig. 13　Transration.
上の画像が平行移動前、下が平行移動後である。

7　まとめと課題

宇宙天気予報プロジェクトの一環として Yin-Yang 格子上の MHD データを可視化するプログラム SV4 を開発した。SV4 は Yin-Yang 格子上で定義された磁場のベクトル場データを読み込み、Yin-Yang 格子上で直接可視化処理する。ユーザーが指定する点群から複数の磁力線を追跡し、OpenGL を用いて描画する機能を持つ。磁力線の積分には４次精度のルンゲ＝クッタ法を用い、空間位置は Yin-Yang 格子上で線形補間を行う。GLUT を用いたクラフィカルユーザーインターフェースを備えており、対話的に様々な角度や位置から表示した磁力線を観察することが可能である。

SV4 の特徴は Fortran90 言語で書かれているという点である。これは宇宙天気予報プロジェクトに参加するシミュレーション研究者が C/C++ 言語よりも Fortran90 言語になじみがあり、この言語で書かれた可視化ツールをソースコードごと提供して自由に改変したいという強い要望があるためである。そのために SV4 では f90gl を利用した。

太陽シミュレーション研究における可視化ツールとして、SV4 プログラムにはマウスによる磁力線の出発点の決定、断面図の表示、等値面の表示などの追加機能が必要である。また今のところ表示に時間はかからないが、プロジェクトの進行と共に太陽コロナシミュレーションの規模が大きくなると、解析すべきデータは今使用しているデータより大きくなるので、それにも対応できるように高速化が必要となる。これらを実現できるよう今後も SV4 の開発を続けていきたい。

8 謝辞

陰山聡教授には１年間を通して様々なご指導頂き、各地の研究機関で、最前線の研究現場をみる機会も与えて頂きましたことを深く感謝致します。名古屋大学、草野完也教授には太陽現象の基礎についてご教授頂ました。また太陽コロナシミュレーションの研究に必要とされる可視化機能について様々な示唆をして頂き、f90gl ライブラリとその使い方についても教えて頂きました。本研究で用いた太陽シミュレーションデータは名古屋大学の塩田大幸博士から提供して頂いたものです。私のためにわざわざデータを作り直して頂き、さらにそのデータ構造をわかりやすくを教えて頂きましたことを深く感謝致します。同じ研究室の古田敦哉氏と村田歌織氏には研究上様々なアドバイスをもらいましたことを深く感謝致します。

参考文献

[1] 宇宙モデリングプロジェクト, http://www.jamstec.go.jp/ifree/space_earth/jswm/ja/

[2] 陰山 聡, 大規模シミュレーションデータの可視化, システム／制御／情報, vol52, pp51-57.(2010)

[3] Will Schroeder, Ken Martin, Bill Lorensen. The Visalization Toolkit, 2nd Edition. Prentice Hall PTR (1998).

[4] 牛島 省, 数値計算のための Fortran90/95 プログラミング入門, 森北出版株式会社 (2007)

[5] Akira Kageyama and Tetsuya Sato. "Yin-Yang grid" An overset grid in spherical geometry. *Geochemistry Geophysics Geosystems*. (2004)

[6] Akira Kageyama and Nobuki Ohno . Visualization of spherical data by Yin-Ynag grid. *Computer Physics Communicatons*. (2009)

[7] R.A. Drebin, L.Carpenter, P.Hanrahan. Volume rendering, *CompuerGraphics*, *Proc.SIGGRAPH'*88. vol.22 pp.65-74(1988)

[8] 床井 浩平, GLUT による OpenGL 入門, 工学社 (2005)

[9] William F. Mitchell. f90gl. http://math.nist.gov/f90gl/