VISUALIZATION FOR SPACE WEATHER PREDICTION USING FINITE-TIME LYAPUNOV EXPONENT IN 3D MAGNETIC FIELD

Sara Binazir

Supervisors: Rickard Englund, Alexander Bock
Examiner: Ingrid Hotz

Tuesday 13th September, 2016

Linköpings universitet

Acknowledgements

I would like to many thanks to proffesor Ingrid Hotz and my supervisors Rickard Englund and Alexander Bock. And a great Thanks to my parents and my sister.

Abstract

In this work, we visualize the magnetic field of the Sun wherein the data was taken from the National Aeronautics and Space Administration (NASA). We use Finite-Time Lyapunov Exponent (FTLE) in 3D Which is characterized by Coherent Lagrangian Structures providing a powerful framework for the analysis and visualization of wrapped technical flows. Its definition is easy and intuitive and has a profound theoretical foundation. While the associated computational cost is fundamentally prohibitive, the application of this approach shows straightforward in theory. We use this method to consider divergence of neighboring field lines to get a topological segmentation of the surface. Our purpose in to support the scientist to understanding and analyzing the data resulting from these simulation. We apply FTLE method to help them easily find which streamlines are more divergence and vice versa and compare FTLE Values for Selected Points and FTLE high resolution to find the accurate one and faster computation time between them.

Contents

1 INTRODUCTION **6**

1.1 Visualization . . . 7

1.2 Space Weather . . . 7

1.2.1 Space Weather and Technology . . . 8

1.2.2 Effects of Space Weather . . . 9

1.3 Goal and Challenge . . . 12

1.4 Data . . . 12

2 RELATED WORK **16**

3 CONCEPTS **18**

3.1 Transformation . . . 18

3.2 Local Frame of reference . . . 20

3.3 Streamline . . . 21

3.3.1 Trilinear Interpolation . . . 23

3.3.2 Euler method . . . 23

3.3.3 Runge-Kutta 4th Order method . . . 25

3.3.4 Compare Euler and Runge-Kutta 4th Order methods . . . 28

3.4 Sphere and Disc . . . 28

4 METHODS **31**

4.1 Finite-Time Lyapunov Exponents . . . 31

4.1.1 FTLE Values for Selected Points . . . 32

4.1.2 FTLE High Resolution . . . 32

4.2 Transfer Function . . . 35

5 IMPLEMENTATION IN INVIWO **36**

5.1 Inviwo . . . 36

5.2 Processors . . . 36

5.3 Properties: . . . 38

6 RESULTS **41**

6.1 FTLE Values for Selected Points and FTLE High Resolution 41

6.2 Dot between normal and vector 43

6.3 Color Lines 43

6.4 Comparison between FTLE Values for Selected Points and FTLE High Resulotion 48

7 CONCLUSION AND FUTURE WORK **49**

8 APPENDIX **50**

8.1 Transformation function 50

8.2 Euler function 51

8.3 Runge kutta 4th order function 51

8.4 FTLE function 53

1 INTRODUCTION

Fluid flows are necessary objects of study in a broad range of Scientific, medical and engineering applications. In particular the optimization of numerous industrial processes need the accurate understanding and the control of flows. Combustion, turbomachinery, automotive and aeronautics comprise important application areas. important application areas. The convolution of the considered flow phenomena and the refined accuracy applied for their investigation yield numerical flow datasets to derive the insight essential for the specific task at hand.

To address the challenge embossed by the size and the qualitative convolution of the corresponding vector field datasets, scientific visualization probing has explored various approaches that have in joint the target of characterizing, extracting and visually demonstrating salient flow structures across spatial and temporal scales and these methods are distributed into topological and feature-based approaches. The topological technique is sometimes unable to capture necessary flow patterns like vortices and its Eulerian viewpoint and absence of Galilean invariance make the commentary of topological structures problematic in the transient setting. In comparison feature-based visualization techniques usage problem-driven and heuristic feature explanation to recognize interesting patterns in an application-specific fashion. In this context, the notion of Lagrangian Coherent Structures (LCS) to make a portrait of the flow and its correspond to ridges of the Finite-Time Lyapunov Exponent (FTLE), a scalar field that specify the amount of stretching of the trajectory of a point over a finite time interval given a definite starting time.

In this work, we visualize the magnetic field of the sun. Sun is our closest star and include of hot gas and some of them right toward the Earth and therefore the Earth's weather affected from the Sun's weather and because of that space weather and magnetic field of the Sun are important for human's life.

An important question is whether magnetic field lines originating from the sun are closed or open. Brute force method would compute field lines for regular sampling on the solar surface but it turns out that the results are not accurate enough even for high sampling density [1]. So we used FTLE method to consider divergence of neighboring field lines to get a topological segmentation of the surface.

The underlying software framework used in this work is Interactive Visualization Workshop (Inviwo) which is an extensible C++ framework for simple prototyping of intractive applications to loading and visualize the Data.

In the following pages, we explain the background and requirements such as Visualization ,Space

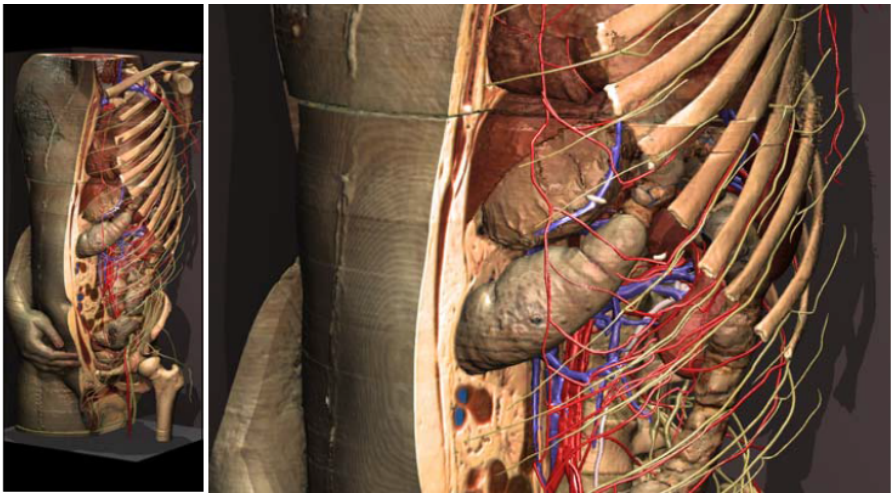

Figure 1: Visible Human Project [23].

Weather, our goal and challenge and our data to help the reader to better understand work. The content of this work is organized as follows: we review previous works in Section two. The underlying basics used for this visualization is presented in Section three. We describe our methods in Section four. Implementation our visualization in Inviwo shall be explained in Section five. We present our findings in Section six. Finally we conclude our presentation by negotiating possible extensions of this work in Section seven.

1.1 Visualization

Visualization is the process of showing abstract business or scientific data as images. It means that lots of data come from something that is not visible such as inside of the human body and visualization converts them to visible and helps the scientist in understanding and analyzing the data results as shown in Fig. 1 .

1.2 Space Weather

The Sun is the most energetic object in the solar system and closest star to the Earth and like a ball of fire containing plasma rather than a hot gas (most hydrogen is ionized). Magnetic field of the Earth can be observed in Fig. 2 it protects us from most of the particles the Sun throws at us. Solar activity drives out atomic particles and radiation from the Sun all along solar flares and coronal mass ejectionstimes when the Sun also discharge billions of tons of plasma. Space

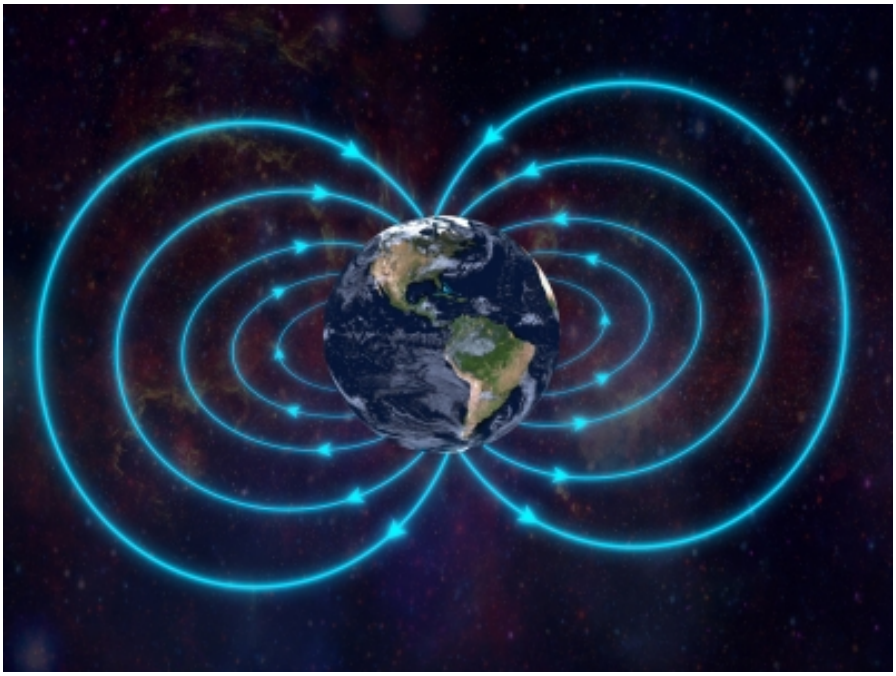

Figure 2: Earth magnetic field. Lines exit near the South Pole and re-enter near the North Pole [10].

Weather is how we return to the variations on local space environment with rejected radiation and particles and how these variations influence human society and Earth such as electronic failures in satellites, connection and navigation difficulty in airplanes and miss satellites in atmospheric drag. The solar wind that hits the Earth deforms the magnetic field and it thereby induces an electric field which can be difficult to handle by long-distance electric power lines.

1.2.1 Space Weather and Technology

Our captivation with technology are endless such as plow and printing press and some of those technologies are efficacy to Space Weather.
Due to all the changes, and indeed technological advances, that have come about in our world today, we have made our lives quite susceptible to space weather. The development of thermosphere and heating at solar maximum increments the drag on satellites and leads to them to fall out of orbit. Suns energetic protons can damage equipment such as astronauts and injure aircrew. Airlines fly polar routes to save fuel and time but space weather may cause relevant

radio outages making them divert to more costly routes.
Fig. 3 shows a timeline of technology and sunspot number. It was more than 170 years ago that we found out about the 11-year sunspot cycle. As time goes by, there are more and more electronics in orbit around our Earth and our dependence on the them increases.

1.2.2 Effects of Space Weather

Fig. 4 shows a few ways that modern technology can be affected by Space weather. These effects are for systems adjacent to the Earth and whenever we travel further, we will discover that all planets will see their own version of Space Weather. For instance, a small magnetic field of the Mars is much less wide than the Earth's and the effects of magnetic storms become tiny at Mars. Now we will briefly explain how different technologies are affected by Space Weather.

Satellites: Solar storms can cause cutting in satellites signals, increment the frictional drag on satellites in LEO and lead satellites to re-enter the Earth's atmosphere more quickly than expected and degrading their orbits. Since satellites orbit the Earth they attract electrons from the magnetosphere and ionosphere and when solar X-rays impact the surfaces, electrons are lose. If a satellite has plenty or few electrons a spark can occur, annihilating electronic parts.

Astronauts: Sun's high-energy particles can harm living subjects. We are preserved by the composed act of the atmosphere and magnetosphere. Astronauts are exposed to much higher amounts of radiation when they are outside the protective atmosphere of the Earth. Space walks are sometimes rescheduled because of the threat of a solar storm to save astronauts from the Space Shuttle.

Aircraft: Airlines are affected by space weather in different ways. Aircrew are exposed to greater levels of radiation than at the surface when they are on polar routes(latitude above 78 degree). Flights at higher latitudes should use HF radio communications and at latitudes below 82 degree may use satellites to connection with the ground. The main reason limiter of the number of flights is worrying about cutting of HF connections when higher number of airplanes are flying polar routes.

Navigation and Communications: Due to its use of long electrical conductors, the telegraph was affected by magnetic disturbances as the first system created by our species. Each time aurora

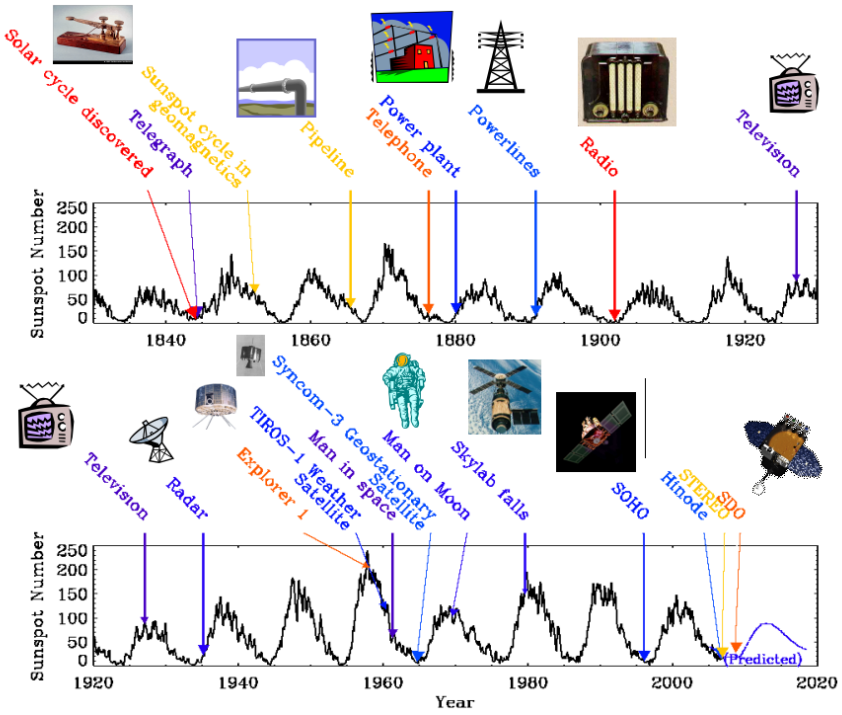

Figure 3: Space Weather and Technlogy [11].

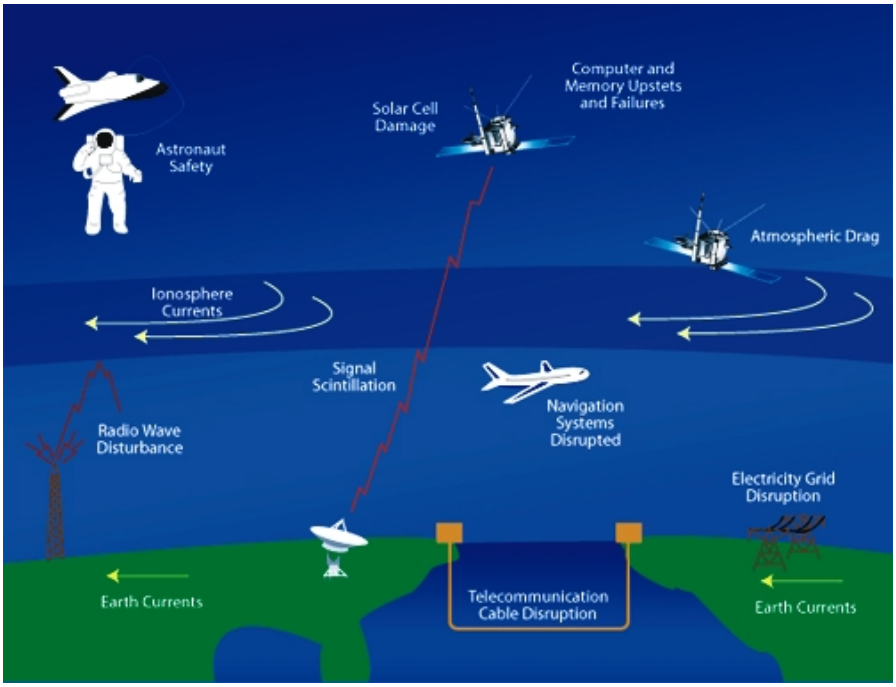

Figure 4: Effects of Space Weather [11].

appeared there were disruptions in the functionality of the telegraph system. Most recently, GPS the very ubiquitous Global Positioning System has been quite susceptible to space weather. Not only the GPS, but in practice every man-made system that uses ionosphere for transmitting or receiving its waves, is affected by space weather. Recent research has represented that explosion of energy from the Sun may interrupt mobile phone connections, lead to calls to be dropped or to include noise.

Powerlines and Pipelines: Every year, our power grid becomes rather wrapped and interrelated and this lets electricity to transferred over great distances from producers to consumers. If space weather interrupts the power transmission grid the outage can cascade through a great area of the energy grid. Electricity is a special merchandise since it may not be simply stored, so it should be use quickly after it is generated. Pipelines transfer material that may keep at both ends of the pipe. Defect of the electrical grid may lead to a falling of the grid and defect of pipelines may cause a mass failure, but the latter only affects the customers of that particular pipeline.

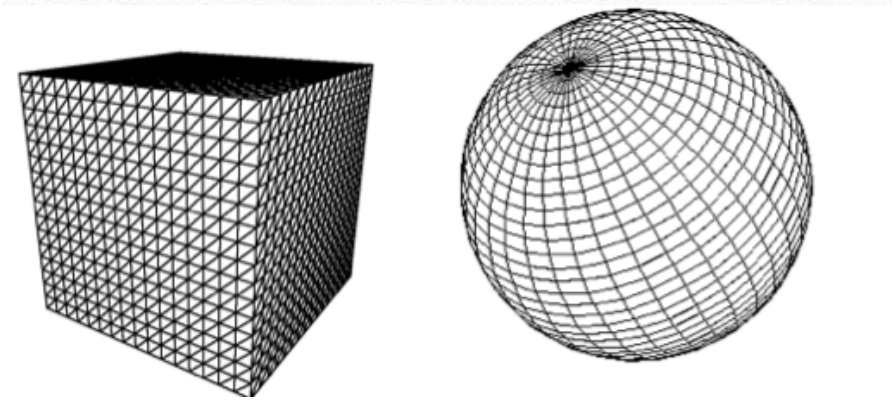

Figure 5: The size of the samples in Cartesian coordinate are the same but in Spherical coordinate are not the same.

1.3 Goal and Challenge

Our goal within this work is to support the scientist to understanding and analyze the data resulting from these simulations. We apply FTLE method to help them easily find which streamlines are more divergence and vice versa and compare FTLE Values for Selected Points and FTLE high resolution to find the accurate one and faster computation time between them.
In this work converting polar coordinates to cartesian coordinates or vice versa is our challenge because in cartesian coordinate all the samples are the same size but in polar coordinate they are not the same size, in north pole or south pole, size of the samples become smaller and when moving from them toward the middle they become larger as shows in Fig. 5.

1.4 Data

The original model is ENLIL with Cone Model. ENLIL is a time-dependent 3D MHD model of the heliosphere and solves equations for magnetic fields, plasma mass, momentum and energy density, apply a Flux-Corrected-Transport (FCT) algorithm. The outer radial boundary can be corrected to include planets or spacecraft of interest and the inner radial boundary is placed beyond the sonic point, typically at 21.5 or 30 solar radii. It coverage 360 degrees in azimuth and 60 degrees north to 60 degrees south in latitude.
ENLIL cone model forecasts CME diffusion from the ENLIL inner boundary to the interesting point as shows in Fig. 6. The based idea of the cone model is, close to the sun CME diffusion pervade with constant angular and radial velocity and has a cone shape. ENLIL catch the cone

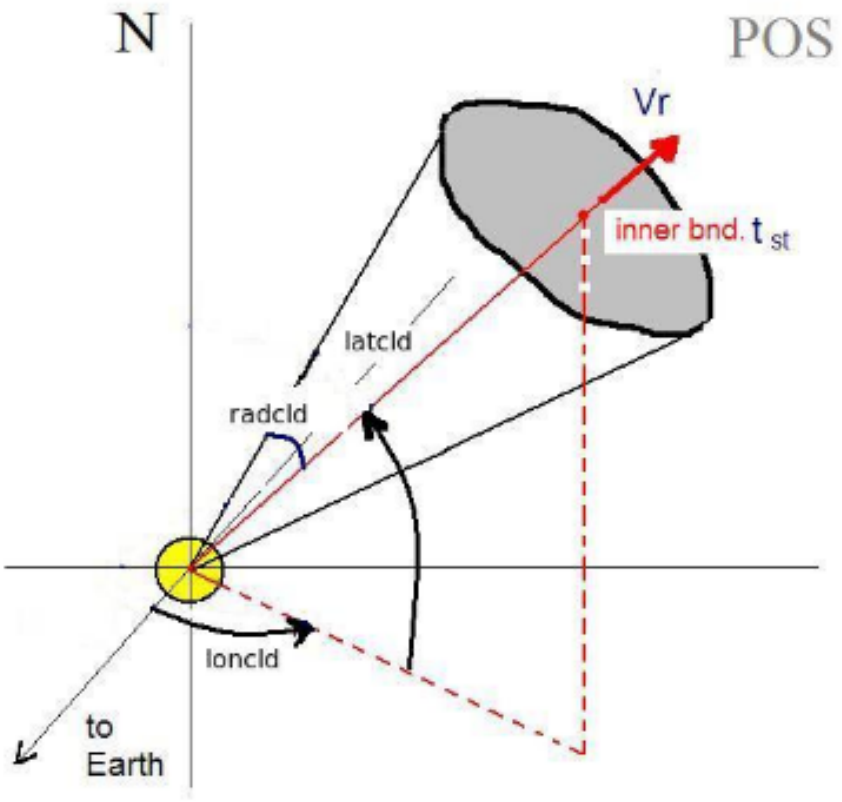

Figure 6: ENLIL cone model forecasts CME diffusion from the ENLIL inner boundary to the interesting point [25].

model input parameters at its inner boundary as following: start date, start time, cone latitude, cone longitude, cone radius and radial velocity at the ENLIL inner boundary.
The model published by D. Odstrcil [26] and it is run by the CCMC (Community Coordinated Modeling Center) at NASA Goddard.
Fig. 7 and Fig. 8 represent our data in cartesian coordinate and polar coordinate respectively. In this data, theta is between -π and π and phi is between 0 and 2 π. The important issue is we visual the data in cartesian coordinate and compute it in polar coordinate.

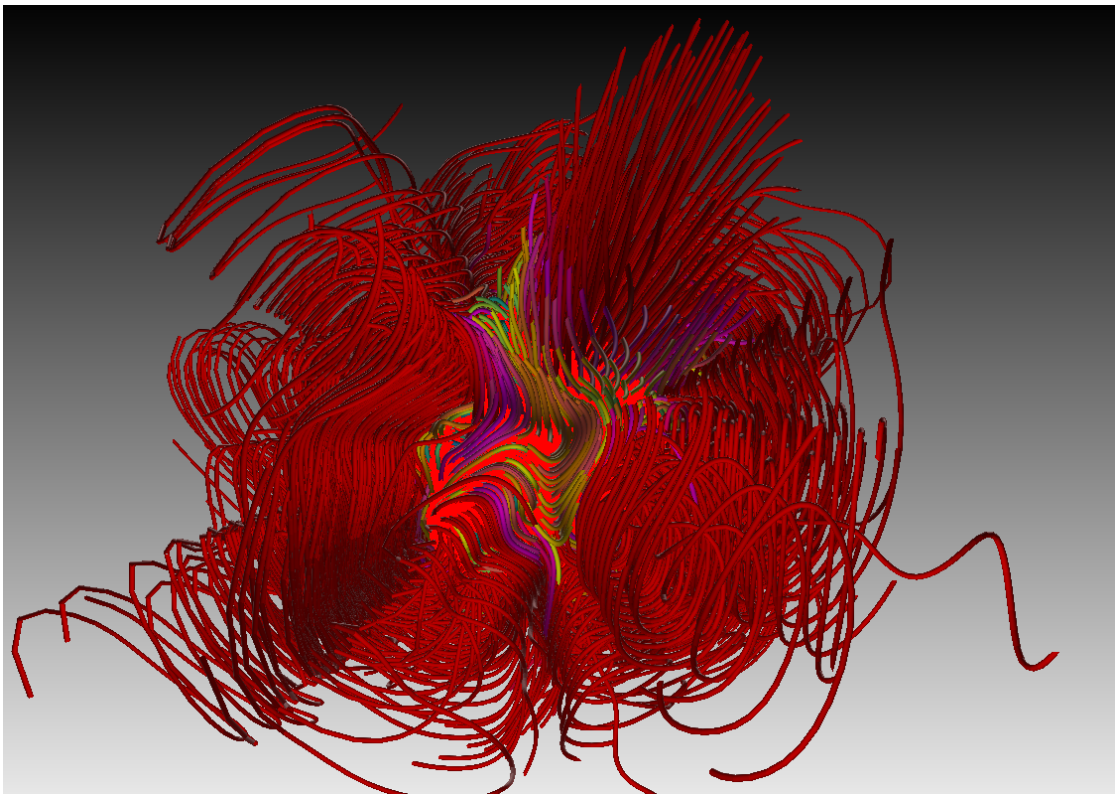

Figure 7: Volume in cartesian coordinate. Length applied for streamlines color.

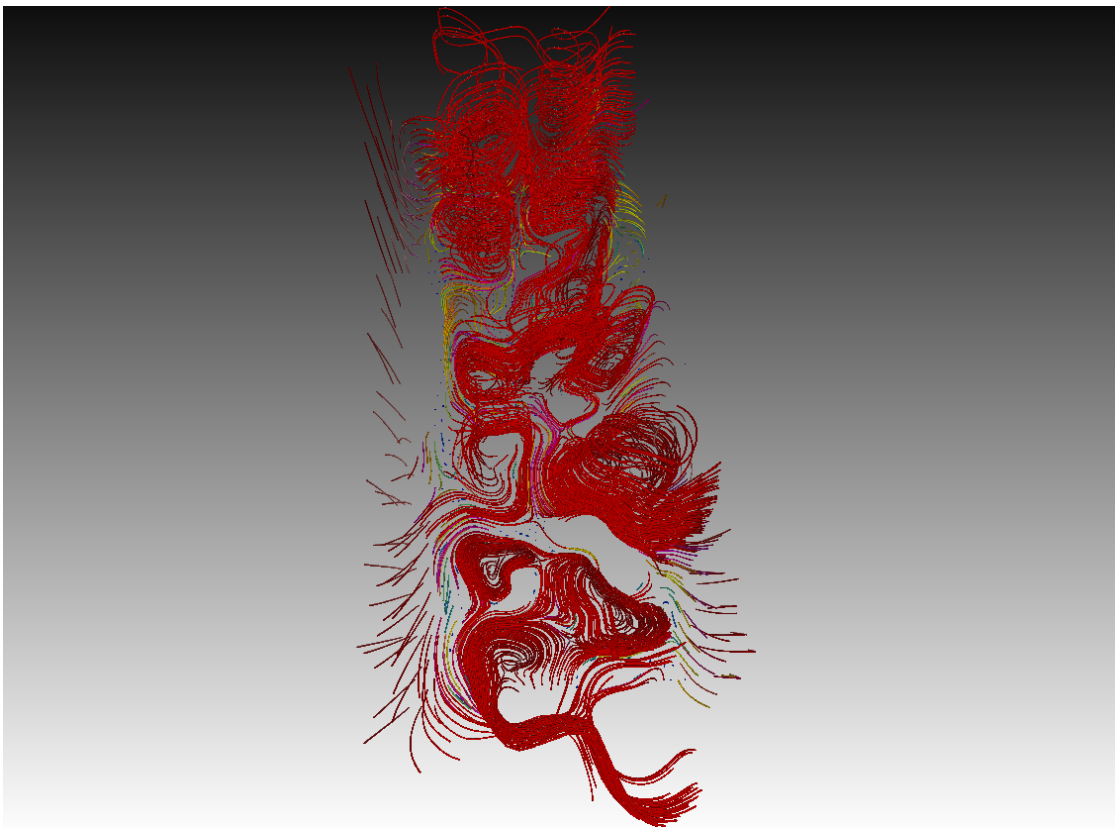

Figure 8: Volume in spherical coordinate. Length applied for streamlines color.

2 RELATED WORK

In this section, we will discuss the previous work and how it is related to our approach.
As introduction to the subject of rendering solar magnetic structures, using magnetic field extrapolations to infer the 3-dimensional structure of coronal loops began with the computer code devised by Schmidt in the early 1960s [[29], [30]]. Rust was the first one use Schmidts extrapolation scheme to evaluate the 3D magnetic field structure with the coronal loops [[31], [32], [33]]. Rusts analogies demonstrated a fine correlation of the current-free field lines with the direction of the coronal loop structures.
Field-line configurations cater a unique analysis method to derive data from solar magnetograms. Field-lines configurations used to specify consistency and interconnection of the photospheric and chromospheric magnetic fields. The extrapolations of magnetograms applied current-free fields situated on active regions [[29], [34]].
Due to the absence of sensing techniques that provide 3D data of the Suns magnetic field, various approaches that analogize such data from line of sight magnetograms. More realistic approach use of a full 3D global magnetohydrodynamics (MHD) model(Mackay and Yeates [27]). Machado et al. presented a technique for visualize the stremline-based mapping between the boundary of a simply-connected subregion of arbitrart 3D vector fields and streamlines are seeded on one part of the boundary and the residual part is a escape border. The approach aims at topologically consistent extraction of their boundary.
Streamline and pathline are the most popular visualization techniques for 3D flow. The main problem for line rendering is occlusion and clutter so we need filter methods. FTLE is a method that is usually developed for flow and though we are not concerned with flow data some of the questions are similar so therefore i will shortly summarize some related work from FTLE method. Previously, Finite-Time Lyapunov Exponent (FTLE) method has been applied in visualizing and analyzing. This approach is a scalar quantity to extend the stretching induced by the flow. The method was introduced by Haller [2] in his seminal paper in 2001 as a means to characterize coherent Lagrangian structures in transient flows and practiced an intensive research since [[3], [4], [5], [6]]. He introduced Finite-Time Lyapunov Exponent as a geometric approach, contrasting it with an analytic criterion that Haller proposed as well. The target of both approaches is to characterize coherent structures in terms of maintenance of certain stability types of the velocity slope along the path of a particle. This seminal share generated a considerable interest in FTLE and its utilizations to the structural analysis of glancing flows in the fluid dynamics community.

He examined the robustness of the structures characterized by FTLE [4]. He demonstrated that they stay valid covered by approximation errors in the velocity field and offered to identify ridge lines of the FTLE field in stable and unstable manifolds [12].
Shadden et al. researched the theory of FTLE and Lagrangian coherent structure (LCS) in 2D [7]. He displayed that the flux across ridges of FTLE field is negligible and little. Arbitrary dimantions for an extension of LCS was proposed [13]. These notions have been used in the analysis of vortex ring flows [14] and turbulent flows [15], [16], [17]. Sadlo and Peikert offered an extraction of LCS method through ridge-driven adaptive refinement of the FTLE field [18] and after that extended it to support the tracking of LCS over time [19].
The acceleration of FTLE computations for 2D flows was previously considered by Garth et al. [9] and mapping pathline integration to the GPU and their method does not extend to 3D flows. Garth et al. also explained an adaptive FTLE computation for 3D flows [8]. Brunton et al. offered a method that exploits the similarities between trajectories of a sequence of flow maps over time to accelerate the computation [20]. Hlawatsch et al. introduced a new hierarchical calculation scheme for integral curves and explained a GPU implementation [21]. Survey to fine scales can become acutely expensive, particularly for time-dependent data sets and when finer resolutions are only related in a small section of the volume, inessential computations are performed. Recently, Englund et al. [28] used FTLE method for controll blood flow clustering and FTLE not focus on separating extremal lines but on local minima and regions of low FTLE intensities to extract coherent flow.
In the visualization community, several approaches have been proposed to increment the performance, accuracy and usefulness of FTLE method. In this work we aim to consider the FTLE method to control the visualization of particles to display divergent of neighboring field lines in 3D flows which is similar to Burger et al. [22] and Garth et al. [8]. But we apply FTLE method without time parameter and do this method in values for Selected Points and high resolution and then compare these two methods.

3 CONCEPTS

This part includes general concepts concerning the rendering of line like structures, discrete data, as well as dealing with specific challenges related to our data. For challenge clutter, FTLE as filtering challenge data given in local coordinates. In the following pages we explain these concepts in detail.

3.1 Transformation

For any calculation within the same space a kind of parametrisation is necessary which means assigning coordinates to the space, this lets us specify locations in space, measure distances and allocate values. Parametrisation is not unique and depending on the difficulty one or another parametrisation may be helpful. We must always keep in mind that intrinsic physics or geometric properties should not depend on the selected parametrisation but when moving among different parametrisation the particular representation may change. In this work we will only consider domains that are subsets of the 3D observation space.

Euclidean coordinates: The most usual parametrisation for the 3D space are Eudlidean coordinates. This coordinates mention to a globally defined orthonormal frame of reference. The point's position is specified by its x, y and z coordinate with regard to a global orthonormal frame of reference with origin O and unit vectors $\overrightarrow{e}_x$, $\overrightarrow{e}_y$, $\overrightarrow{e}_z$ spanning the space. A point P $\in$ D is given as:

$$P(x,y,z) = O + x.\overrightarrow{e}_x + y.\overrightarrow{e}_y + z.\overrightarrow{e}_z \tag{1}$$

Spherical coordinates: A helpful alternative for spherically symmetric data are polar coordinates or spherical coordinates. The point's position is specified by radial distance r $\in$ R $^+$ from origin O, its polar angle $\theta \in [0, \pi]$ and it's azimuth angle $\phi \in [0, 2\ \pi]$ as shows in Fig. 9.

The transformation between the two parametrizations is given by:

Transformation Euclidean coordinates To Spherical coordinates:

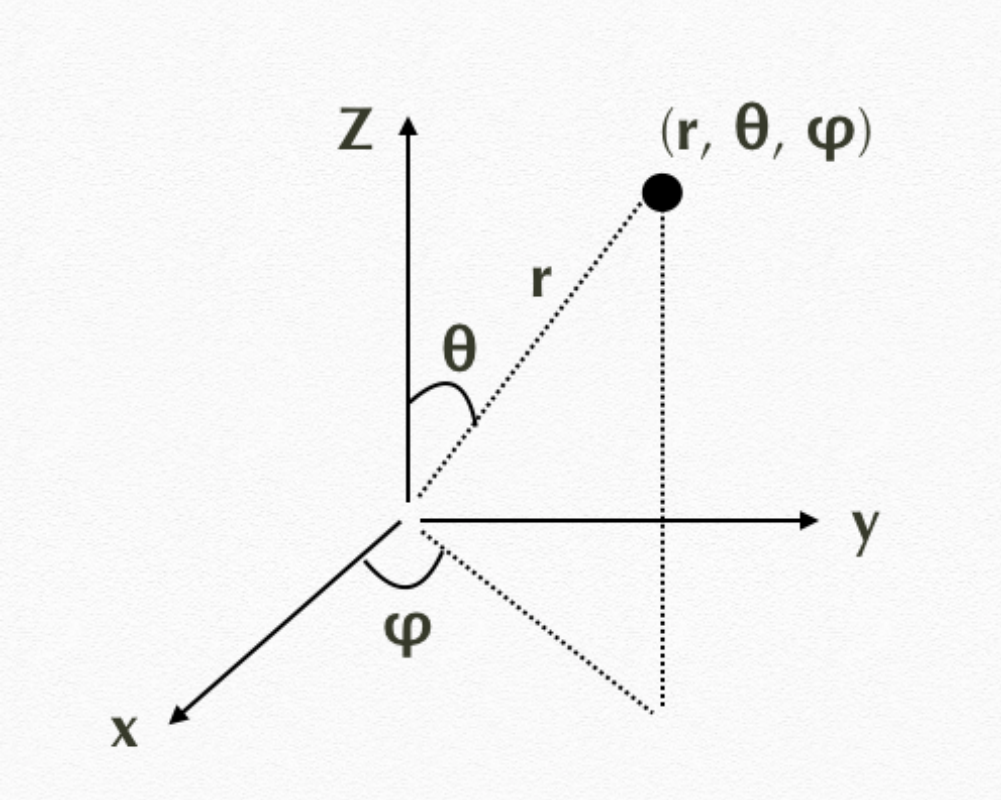

Figure 9: Spherical coordinate with radial distance r, polar angle θ (theta) and azimuth angle ϕ (phi).

$$
\begin{aligned}
r &= \sqrt{x^2 + y^2 + z^2}, \\
\theta &= \arctan\left(\frac{z}{\sqrt{x^2 + y^2}}\right), \\
\phi &= \arctan\left(\frac{y}{x}\right)
\end{aligned}
\tag{2}
$$

Transformation Spherical coordinates To Euclidean coordinates:

$$
\begin{aligned}
x &= r\cos(\theta)\cos(\phi), \\
y &= r\cos(\theta)\sin(\phi), \\
z &= r\sin(\theta)
\end{aligned}
\tag{3}
$$

The Jacobian matrix of the transformation from Spherical to Euclidean coordinates are:

$$
J = \begin{pmatrix}
\cos\theta\cos\phi & -r\sin\theta\cos\phi & -r\cos\theta\sin\phi \\
\cos\theta\sin\phi & -r\sin\theta\sin\phi & r\cos\theta\cos\phi \\
\sin\theta & r\cos\theta & 0
\end{pmatrix}
\tag{4}
$$

J matrix convert a vector in Spherical coordinates to a vector in Cartesian coordinates at position [r, θ, ϕ]. You can observe the tranformation function we used in this work in Section 8.1.

3.2 Local Frame of reference

Local reference frame defined a velocity field in spherical coordinates and is a matrix constructed of the unit vectors dr, dθ, dϕ. This can be seen as a rotation of the Cartesian coordinates. To convert any vector at point [r, θ, ϕ] from local coordinates to Cartesian coordinates we use matrix A which is defined as:

$$
A = \begin{pmatrix}
\cos\theta\cos\phi & -\sin\theta\cos\phi & -\sin\phi \\
\cos\theta\sin\phi & -\sin\theta\sin\phi & \cos\phi \\
\sin\theta & \cos\theta & 0
\end{pmatrix}
\tag{5}
$$

Matrix A use to converted a vector defined at [r, θ, ϕ] in the local frame to Cartesian coordinates with the following transformation:

$$v_{spherical} = A.v_{local} \tag{6}$$

The matrix A is a Jacobian with its column vector normalized to have magnitude of 1. To convert local frame to Spherical coordinates we do the following multiplication:

$$v_{spherical} = J^{-1}.A.v_{local} \tag{7}$$

A is a normalized version of J so the matrix multiplication J ^{1} . A is a linear scaling as following:

$$J^{1}.A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{r} & 0 \\ 0 & 0 & \frac{1}{r\cos(\theta)} \end{pmatrix} \tag{8}$$

The transformation in equation 7 can be rewritten as a component wise scaling:

$$v_{rspherical} = v_{rlocal} \tag{9}$$

$$v_{\theta spherical} = \frac{v_{\theta local}}{r} \tag{10}$$

$$v_{\phi spherical} = \frac{v_{\phi local}}{r\cos(\theta)} \tag{11}$$

The radial component remains the same, the θ and ϕ component is scaled down as r is increasing, in addition, ϕ is incrementing as it the elevation angle from the equator is incrementing.

3.3 Streamline

More fluids like air, water and etc are clear, so their flow patterns are undercover to us and we use flow visualization to make flow patterns sightly and then we can visually qualitative and quantitative flow information. The most common way to visualize a flow field is to describe the paths that fluid elements will follow at each point in time. For steady flow field these paths are called streamlines and for unsteady flow field these paths called pathlines. As shown in the Fig. 10 a streamline is a curve tangent to the flow field all over.

Streamline tracing is a critical part of each such method and include of solving the streamline ODE on several computational grid:

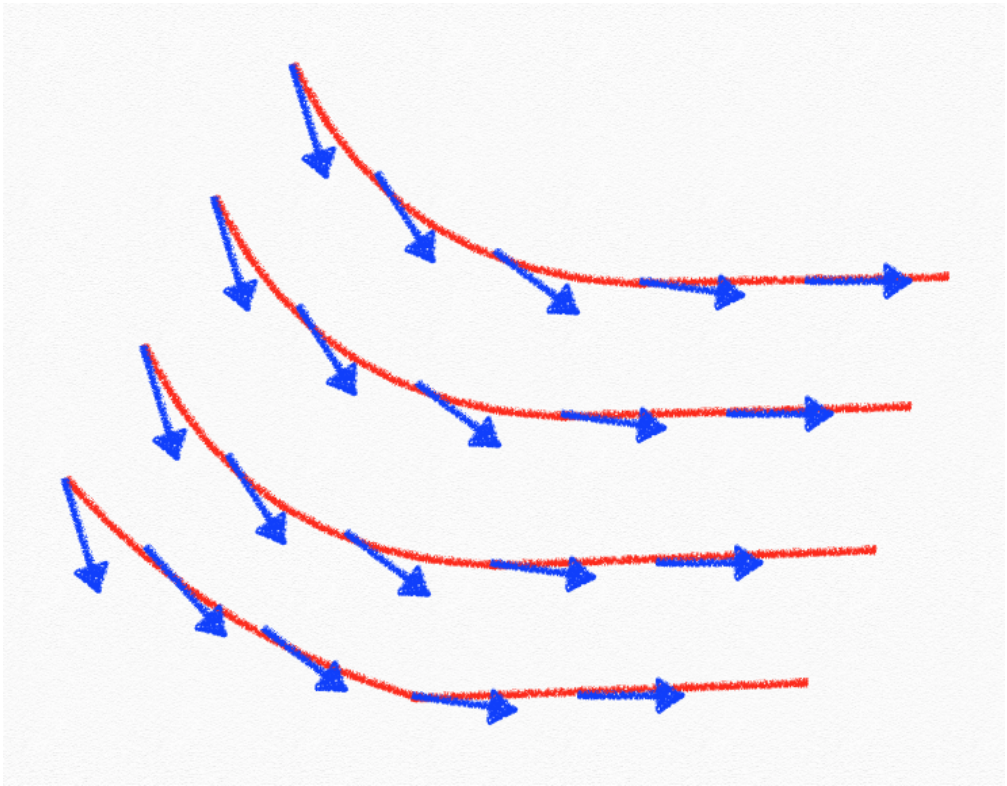

Figure 10: Streamline.

$$dX/d\tau = V(X) \tag{12}$$

where V is th velocity field and X is a point in space and τ is known as the streamline "time-of-flight". To create streamline first we create it with Trillinear interpolation and then used Euler method and Runge-Kutta 4th Order method. In the following we explain these methods.

3.3.1 Trilinear Interpolation

The Trilinear equation is derived by exerting the linear interpolation seven times As shown in the Fig. 11, three times each to specify the points f1 and f0 as showed in the 2D bilinear interpolation and then one more time to calculate the point f. The order of precision is 1 for all these interpolation schemes. The computation of the trilinear interpolation as follows:

4 linear interpolations:

$$\begin{aligned} f_{00} &= \alpha f_{001} + (1-\alpha) f_{000} \\ f_{01} &= \alpha f_{011} + (1-\alpha) f_{010} \\ f_{10} &= \alpha f_{101} + (1-\alpha) f_{100} \\ f_{11} &= \alpha f_{111} + (1-\alpha) f_{110} \end{aligned} \tag{13}$$

2 bilinear interpolations:

$$\begin{aligned} f_0 &= \beta f_{01} + (1-\beta) f_{00} \\ f_1 &= \beta f_{11} + (1-\beta) f_{10} \end{aligned} \tag{14}$$

1 trilinear interpolation:

$$f = \gamma f_1 + (1-\gamma) f_0 \tag{15}$$

where a = (x-xi) / (xi+1 - xi) and B = (y-yi) / (yi+1 - yi).

Fig. 12 shows our result when we use trilinear interpolation to create streamlines.

3.3.2 Euler method

The Euler method is a first-order numerical method for solving ordinary differential equations (ODEs) with a given primary value. It is the most basic explicit way for numerical integration of ordinary differential equations and is the easiest RungeKutta method. Eulers method computes the slope at a given point, and steps forward a given interval to specify a new point and this process repetition at the new point as shows in Fig. 13. The average slope between the initial

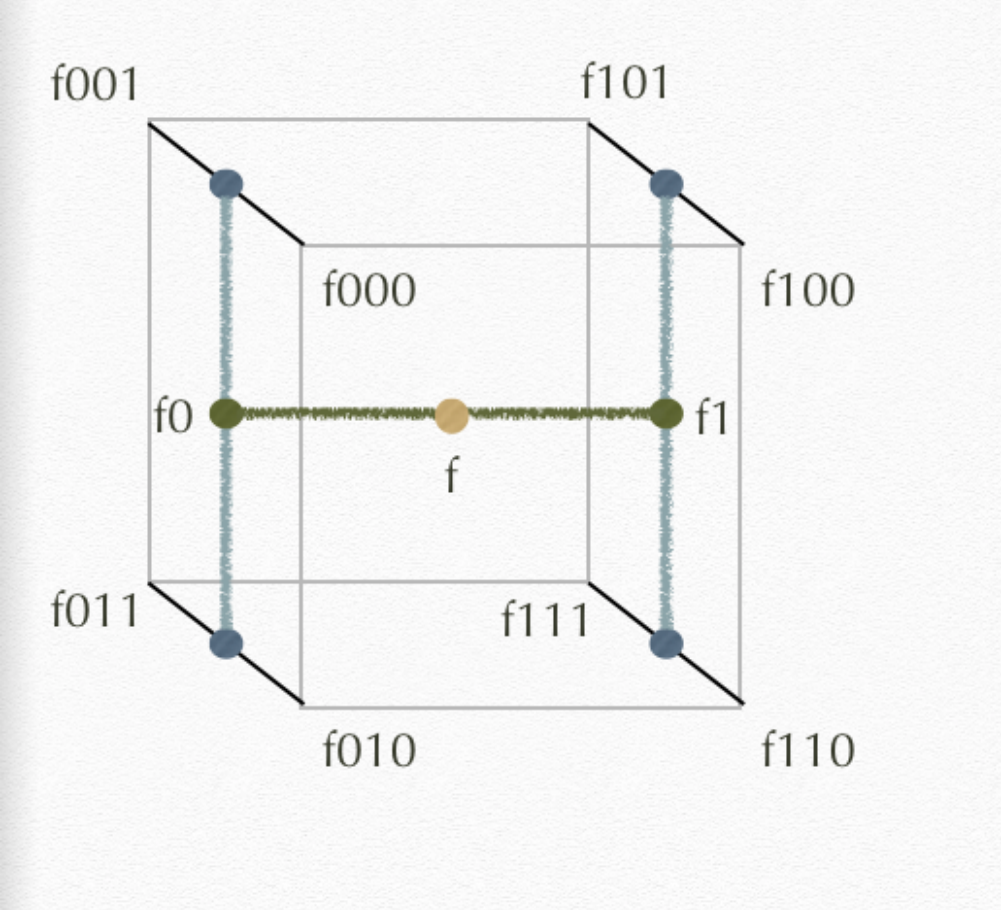

Figure 11: Trilinear Interpolation.

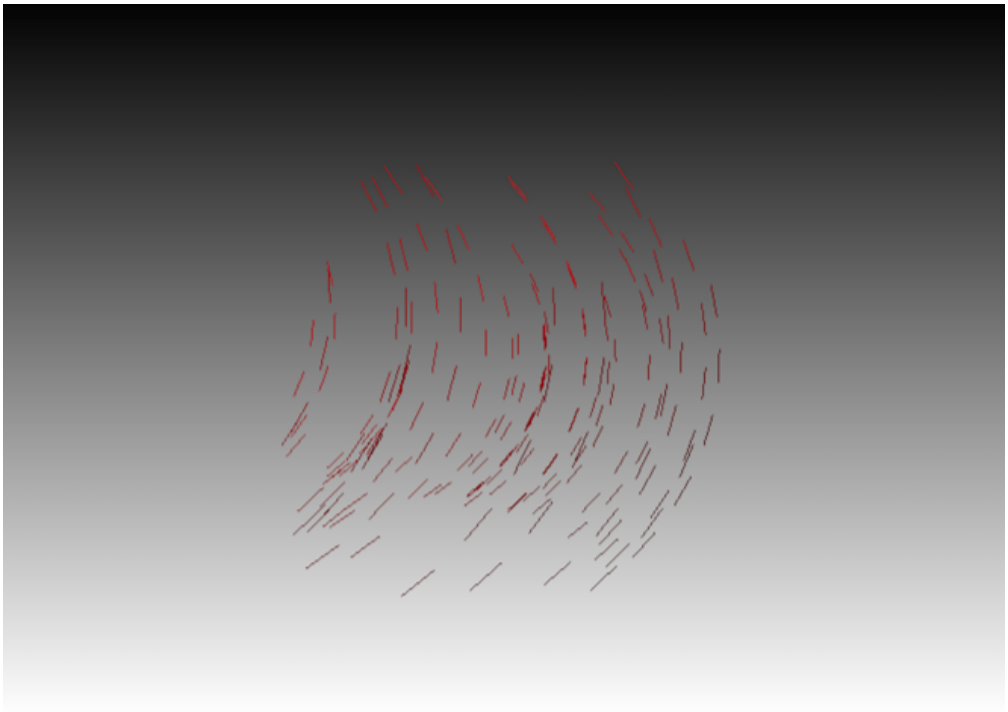

Figure 12: Our result when we use trilinear interpolation to create streamlines.

point and the next point is variable with time unless the average slope is not identical to the slope of the initial point.
Computation of the Euler method as follows:

$$y_{n+1} = y_n + hf(t_n, y_n) \tag{16}$$

where h is a step size and f is our function. You can observe the Euler function we used in this work in Section 8.2.

3.3.3 Runge-Kutta 4th Order method

In numerical analysis, the RungeKutta methods are a group of implicit and explicit iterative ways used in temporal discretization to approximate resolvent of ordinary differential equations. The most popular RK method is RK4 and it presentation a good equivalence between order of precision and cost of calculation. RK4 is the greatest order explicit Runge-Kutta method that needs the identical number of steps as the order of accuracy. The Runge-Kutta 4 usage a total of four experimental steps, K1-K4, to compute a slope based on a weighted average, K1, K2, and

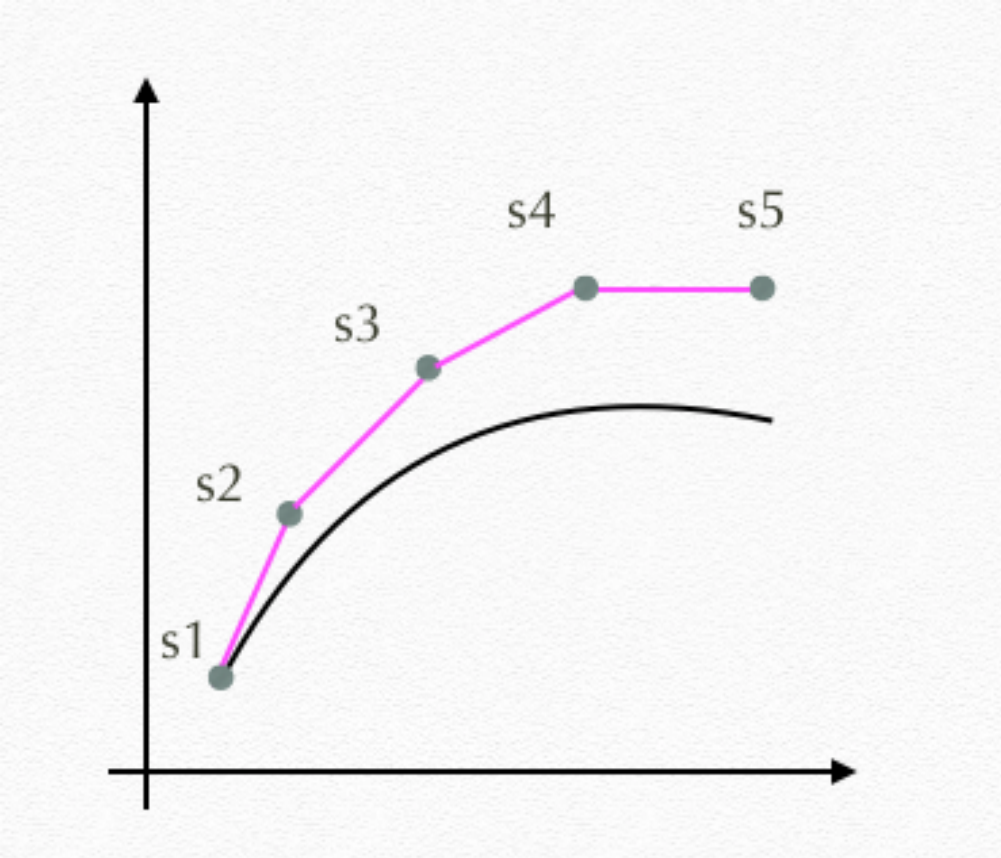

Figure 13: Euler method.

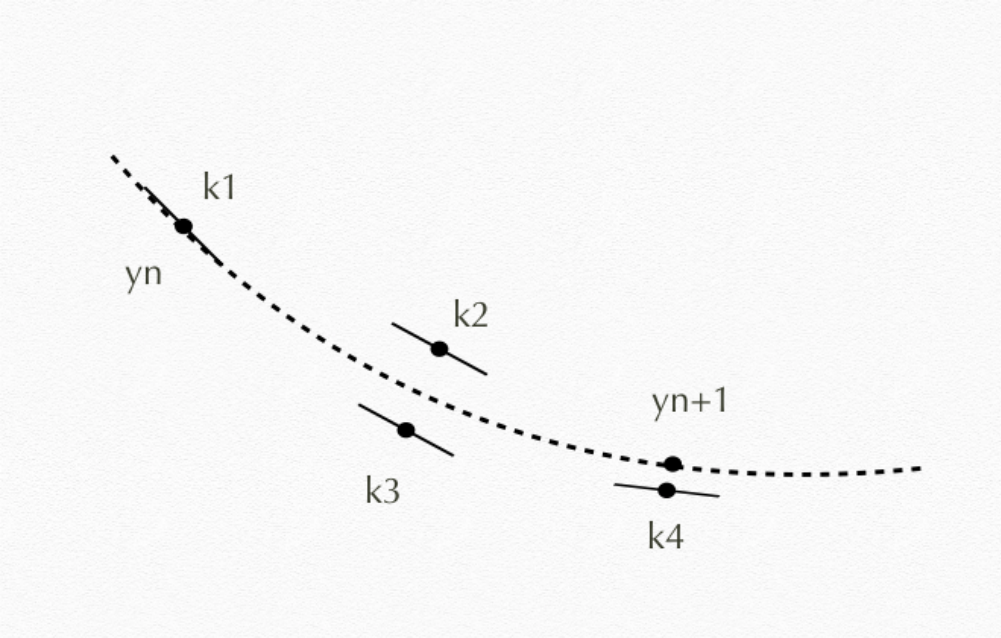

Figure 14: Runge-Kutta 4th Order method.

Yn+1 all represent the same quantities, K3 is a half time step, but determined using the K2 slope, K4 is the slope a full time step using the K3 as shows in Fig. 14. Computation of the Runge-Kutta 4th Order method as follows:

$$\begin{aligned} y_{n+1} &= y_n + h/6(k_1 + 2k_2 + 2k_3 + k4) \\ t_{n+1} &= t_n + h \end{aligned} \tag{17}$$

where h is a step size and $k_1,\ k_2,\ k_3,\ k_4$ as follows:

$$\begin{aligned} k_1 &= f(t_n, y_n) \\ k_2 &= f(t_n + h/2, y_n + h/2k_1) \\ k_3 &= f(t_n + h/2, y_n + h/2k_2) \\ k_4 &= f(t_n + h, y_n + hk_3) \end{aligned} \tag{18}$$

You can observe the Runge-Kutta 4th Order function we used in this work in Section 8.3.

Figure 15: The picture on the left shows our result with Euler method and the picture on the right shows our result with Runge-Kutta 4th Order method.

3.3.4 Compare Euler and Runge-Kutta 4th Order methods

Our result for Euler and Runge-Kutta 4th Order methods shows in Fig. 15. We put z coordinates to 0 and used circle line for ease of analysis.
Euler method is accurate if streamlines are lines and with a rotational flow field Euler method incorrectly diverges due to error. Its performance is poor. It was stable on very small time steps and rapidly gained energy (the numerical result diverged from the analytic one) as the time step incremented.
Runge-Kutta 4th order is precise on higher-order streamlines and higher time steps. It is stable numerical integration method that is easy to implement, popular for integration and best method depends on data and interpretation, but it is more expensive.

3.4 Sphere and Disc

In geometry, a regular icosahedron is one of the five Platonic solids and it is a convex polyhedron with 20 faces, 30 edges and 12 vertices and it has five equilateral triangular faces encountering at each vertex and the vertices are themselves the normal vectors as shows in Fig. 16. Then subdivide the triangles to form more new triangles and project them to sphere coordinate. Fig. 17 shows our result in level 1, level 2 and level 3 subdivisions. Our aim to create sphere is to apply it instead of sun and use the points on the sphere for our seed points.

We create a disc as the same method as sphere but the difference is, we apply this method in 3D for sphere but for disc use this method in 2D. Our target to use disc is for ease of search and

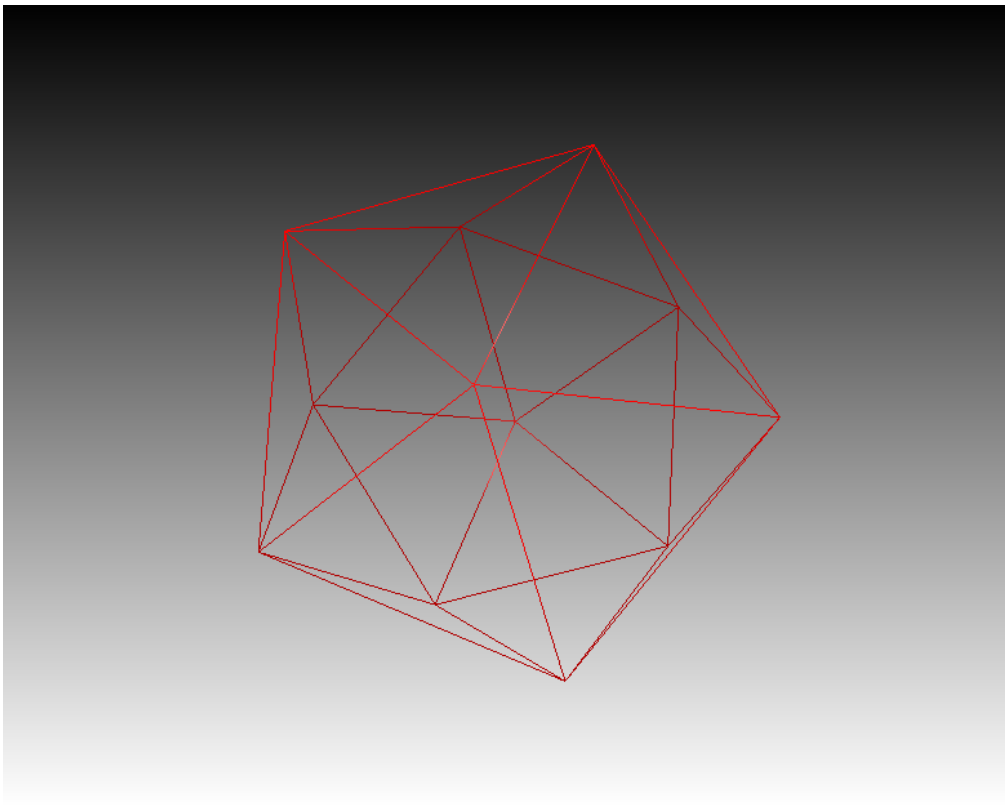

Figure 16: The initial icosahedron with 20 faces, 30 edges and 12 vertices.

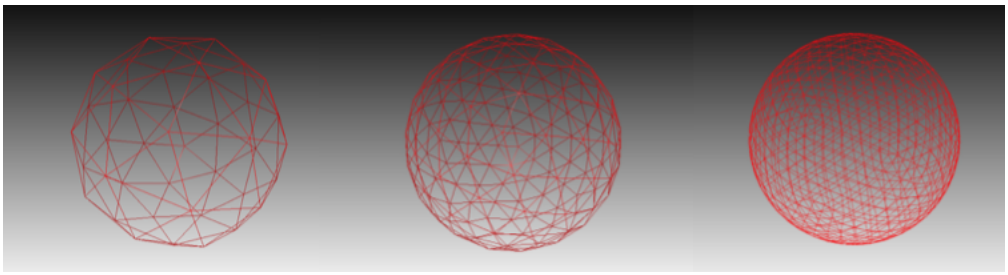

Figure 17: Icosahedron result for sphere in level 1, level 2 and level 3 (left to right) subdivisions.

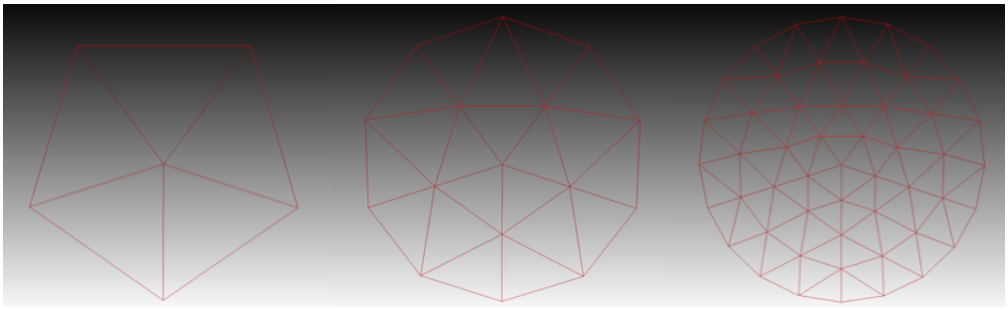

Figure 18: Icosahedron result for disk in level 0, level 1 and level 2 (left to right) subdivisions.

probing. Fig. 18 shows our result in level 0, level 1 and level 2 subdivisions.

4 METHODS

For the analysis of our data we provide filtering and rendering. One of the fundamental concepts that we use to find discontinuities in the behavior of the magnetic field is FTLE which will be described in the following. We start with a general introduction in 4.1 and then discuss two different approaches that we implemented for our setting. In section 4.2 first we explain FTLE Values for Selected Points. For sure we can use this method also for a dense representation of FTLE, however this leads to many redundant pathline computations which makes the analysis very slow, Therefor we use FTLE high resolution. Reaching a reliable accuracy this approach requires a minimum resolution.

4.1 Finite-Time Lyapunov Exponents

The description of the Finite-Time Lyapunov Exponents (FTLE) relies on notions from the theory of dynamical systems. The Lyapunov Exponents are determined to characterize the rate of seperation of infinitesimally close trajectories as time approaches infinity. The idea in the absence FTLE is to apply this concept in the context of finite-time flow fields and to describe asymptotically unstable and stable coherent structures in terms of the trajectories of intimately seeded particles. We bring up a time-dependent vector field V specified over a finite Euclidean domain R $\subset$ R^3 and a finit interim domain I $\subset$ R. The situation X of a particle starting at position at time t_0 after advection along the vector field may be formulated as a map X(t; t_0, X_0) satisfying X(t; t_0, X_0) = X_0 and d/dt(X) = v(t, X(t; t_0, X_0)). A linearization of the local variation of this map around the seed position X_0 is gained by its spatial gradient or Jacobian J_X (t; t_0, X_0) := ∇ X_0 X(t, t_0, X_0) at X_0. The gradient can be used to specify the maximal dispersion after time τ of particles in a vicinity of X_0 at time t_0 as a function of the direction dt_0 along which one moves away from X_0 by knowing the fact that:

$$dt = J_X(t, t_0, X_0)dt_0 \tag{19}$$

Maximizing the norm $\|$ dt $\|$ over all possible unit directions dt_0 corresponds to calculation the spectral norm of J_X (t, t_0 , X_0). So, maximizing the dispersion of particles around X_0 at t_0 is equivalent to evaluating:

$$\sigma_\tau(t_0, X_0) := \sqrt{\lambda_{max}(J_X(t, t_0, X_0)^T J_X(t, t_0, X_0))}. \tag{20}$$

Where λ_{max} is the maximum eigenvalue. To take the average exponential separation rate λ (t, t_0, X_0), the logarithm is applied and the result is normalized by advection time τ to obtain:

$$\lambda(t, t_0, X_0) := 1/|\tau| log\sqrt{\lambda_{max}(J_X(t, t_0, X_0)^T J_X(t, t_0, X_0))}. \tag{21}$$

This rate is then called the Finite Time Lyapunov Exponent and can be evaluated for both forward and backward advection. Large values of λ for forward advection indicate to unstable mainfolds while large FTLE values for backward advection identify stable mainfolds [8]. You can observe the FTLE function we used in this work in Section 8.4.
In the following we explain how we use neighbor points for each seed particles in FTLE Values for Selected Points and FTLE high resolution.

4.1.1 FTLE Values for Selected Points

In FTLE Values for Selected Points we add four neighbor points for each seed particles which these four points are perpendicular with each other and apply FTLE method for them as show in Fig. 19

In this figure, p is our seed particle and p1, p2, p3, p4 are our four neighbor points and d1 is distance of seed particle from each four neighbor points and d2 is distance to next point in the streamline. One issue which important is for computing FTLE If we apply matrix 2×2 then we get two eigenvalues and we loss some of information because we don't use z coordinates but if we apply matrix 2×3 we don't loss our information and we get three eigenvalues.

4.1.2 FTLE High Resolution

In FTLE high resolution because we use Icosahedral method to create sphere in the begining we have 12 vertices or seed points and each seed particles have five neighbor points but when we subdivide sphere that 12 seed points have five neighbor points and another addition seed particles have six neighbor points around. So we should do two differents method for each of these as shows in Fig. 20.

The right panel of this figure (20) shows what we did FTLE for five neighbor points. If we get our first point p1 then in the opposite direction we don't have any point which is perpendicular

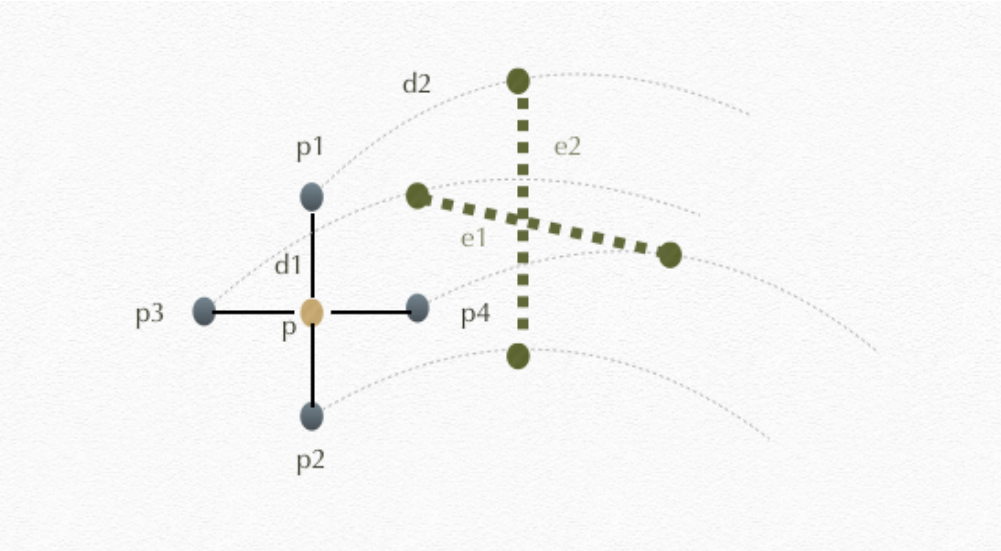

Figure 19: Add four neighbor points for FTLE Values for Selected Points. P is our seed particle and p1, p2, p3, p4 are our four neighbor points. d1 is distance of seed particle from each four neighbor points and d2 is distance to next point in the streamline.

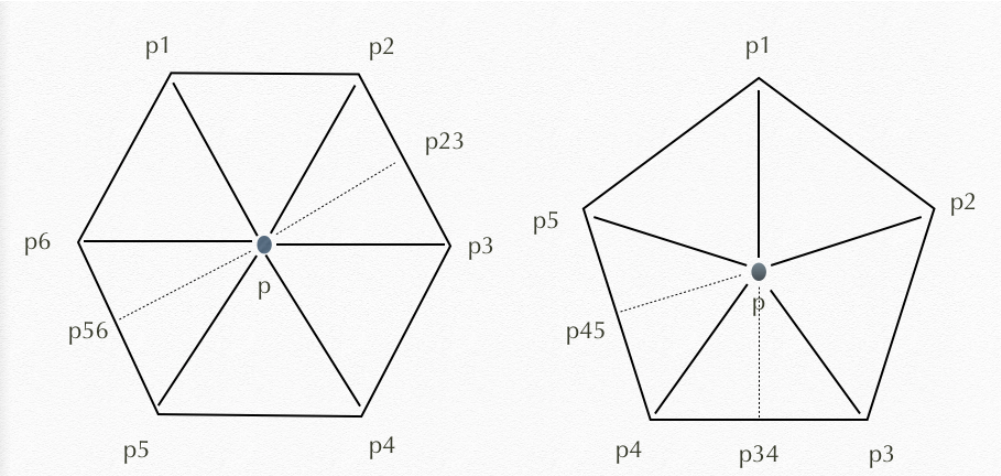

Figure 20: Two various methods for five (right) and six (left) neighbor points for FTLE high resolution.

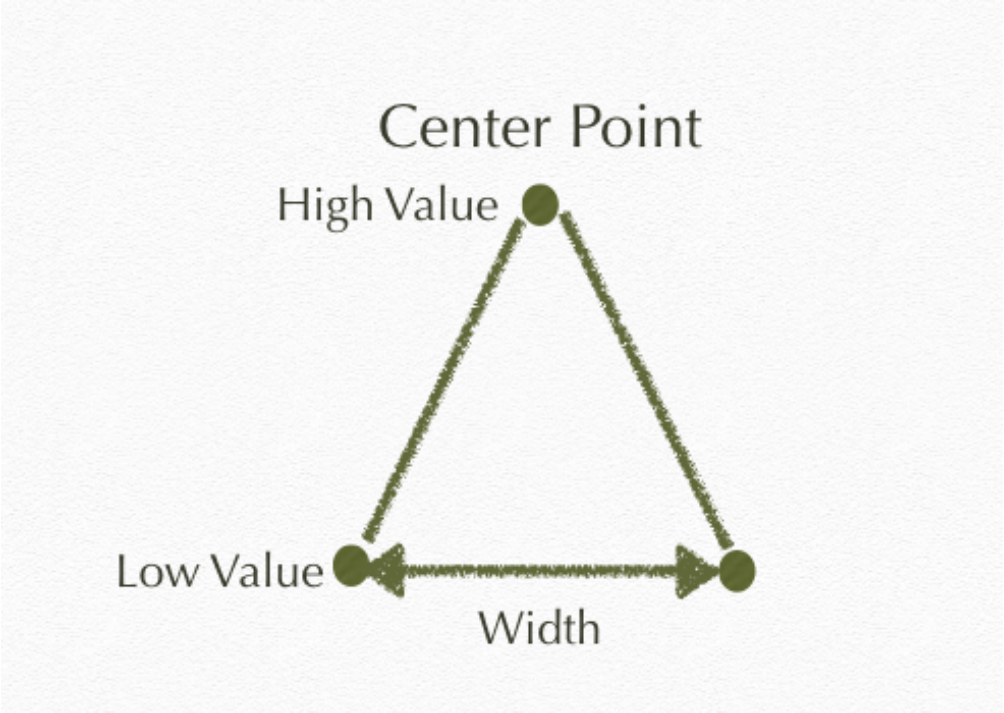

Figure 21: The RGB channels must be fully saturated while the alpha channel change linearly pursuant to four parameters: centerpoint, width, lowvalue, highvalue.

with p1 so we should find the mid point of two points in the opposite direction which is p34 and then subtraction p1 and p34. Do the same for p2 and find mid p45 in the opposite direction and then subtraction p2 and p45. Then create a matrix from these results and finally do the FTLE method.

Left picture in Fig. 20 shows what we did FTLE for six neighbor points. If we get our first point p1 then we have another point in the opposite direction which is p4 and subtraction p1 and p4. For p2 and p6 get p3 and p5 respectively and find the mid point p23 and p56 respectively and subtraction p23 and p56. Then create a matrix from these results and finally do the FTLE method. The important issue is when we find number of neighbor points for each seed points, we should put them in the plane (2D) and sort them from the angles(vector from seed point to first neighbor point (p1) to vector from seed point to another neighbor points).

Figure 22: Color look-up table.

4.2 Transfer Function

A Transfer function specifies how the scalar values are mapped to dimness and colour in the image. The mapping can explicit as a function F_{rgba}(i) = (r, g, b, a). A very easy version is linear interpolation between start and end values. To describe the transfer function we will add several segments for each color channel. The RGB channels must be fully saturated while the alpha channel change linearly pursuant to four parameters: centerpoint, width, lowvalue, highvalue as shows in Fig. 21. This represent a linear ramp that starts at the ”lowvalue” for an index ”centerpoint - width / 2” and continue to the ”highvalue” for an index ”centerpoint”. Then the value reduction linearly until it reaches the index ”centerpoint + width / 2”. The segment should lie in the range [0 255] whenever the low and high value should be in the range [0 1]. The ramp must only affect the determined color channels RGBW where W is white. Eventually, the result sent to the GPU as a texture and it is used as a look up table for volume samples. Then a certain density in the volume mapped to a RGBA value. Fig. 22 shows our colour look up table in this work.

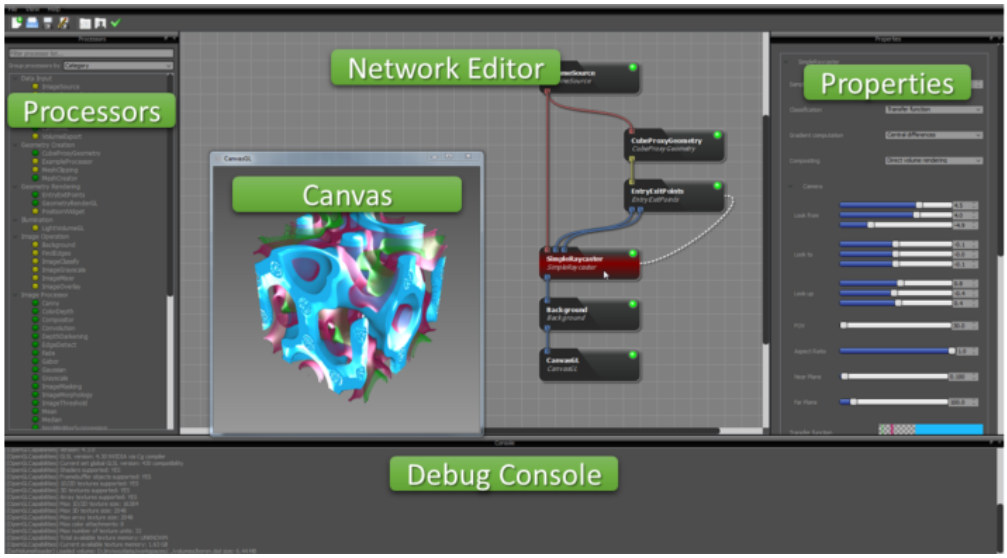

Figure 23: Inviwo contains of Processors, Network Editor, Canvas, Properties and Debug Console [24].

5 IMPLEMENTATION IN INVIWO

For our implementation we apply Inviwo which is a software framework for rapid visualization prototyping and it is written in c++ and exploits modern graphics hardware. In the following we explained overview about Inviwo, our processors and properties we apply.

5.1 Inviwo

Inviwo is an extensible C++ framework for simple prototyping of intractive applications. It has an inbuilt network editor for the designing of data flow networks. The data flow in such a network runs of top to bottom and the nodes are indicate processors and each processors have input and output. Properties define the state of a processor and ports are used to exchange data among processors and properties as shows in Fig. 23.

5.2 Processors

Processors are the primitive objects the user interacts with inside the network editor. We create and add some processors to our network editor as shows in Fig. 24. These processors as follows:

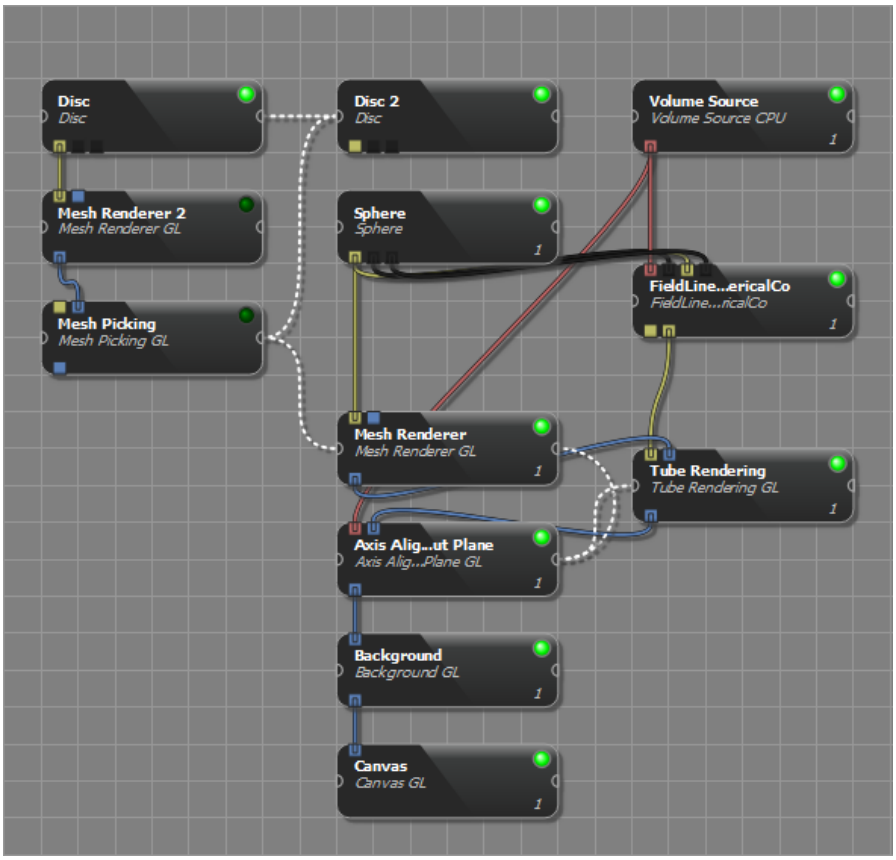

Figure 24: Our Network Editor.

Sphere and Disc Processors: We use Icosahedra method in 3D for Sphere processor and 2D for Disc processor. These processors don't have Input but they have three output, mesh, seedpoints and faces.

FieldLineSphericalCo Processors: This is our main processor and this processor has four input, volume, mesh, seedpoints and faces and two output, mesh and line. It gets input volume from volume source which has a magnetic field of the Sun and input mesh, seed points and faces from sphere or disc. This processor creates streamlines from each seed points with two integration methods, Euler and Runge Kutta fourth order and do the FTLE Values for Selected Points, FTLE High Resolution and dot between vector (first point and second point of each streamlines) and normal of seed points. It converts spherical coordinate to cartesian coordinat and visa versa and gives the lines color from index, FTLE, length and dot as well.

5.3 Properties:

Each processors have different properties. Sphere and Disc processors have subdivide and subdivide back, radius and center properties as shows in Fig. 25.
FieldLineSphericalCo processor has different properties as shown in Fig. 26.
Neighbor Distance: This property changes the distance from each seedpoint from their neighbors.
NextPoint Distance: This property changes the distance from first point to next point in the streamlines.
Trim Distance: This property changes the distance of cutting streamlines.
Power Number: This property changes the power of maximum lambda in FTLE method.
Max Lambda: This property changes the maximum lambda in FTLE method.
Cartesian/Spherical Coordinate: This property changes our coordinates.
Step Number: This property changes the number of steps in streamlines.
Step Size: This property changes the step size in streamlines.
Step Direction: This property changes the step direction in streamlines and it has three options, Forward, Backward and Bi Directional.
Integration: This property changes the method of integration in streamlines and has two options, Euler method or Runge-Kutta 4th Order method.
Vertex Color: This property gives the vertex or seed points color and has three options, FTLE

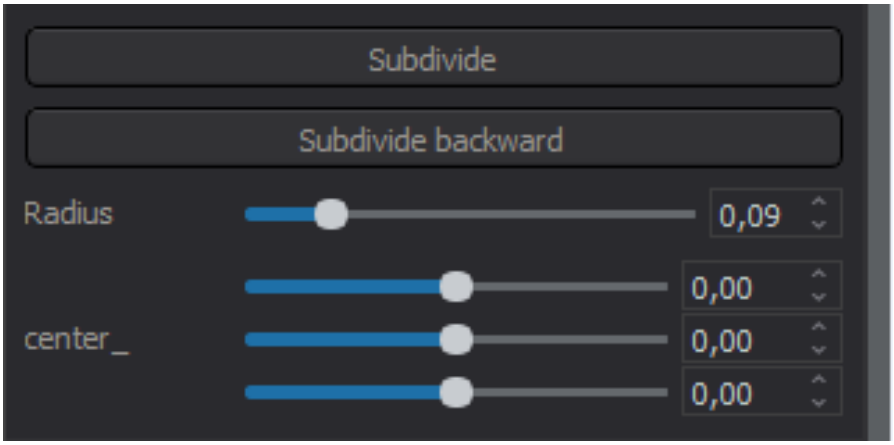

Figure 25: Sphere and Disc processors properties.

LR, FTLE HR and DOT which finds the dot between normal seed points and vector.
Line Color: This property changes the lines color and has four options, INDEX, FTLE , LENGTH and DOT.

After adding these processors in the Network Editor as shown in Fig. 24, our final result shall be displayed in Canvas.

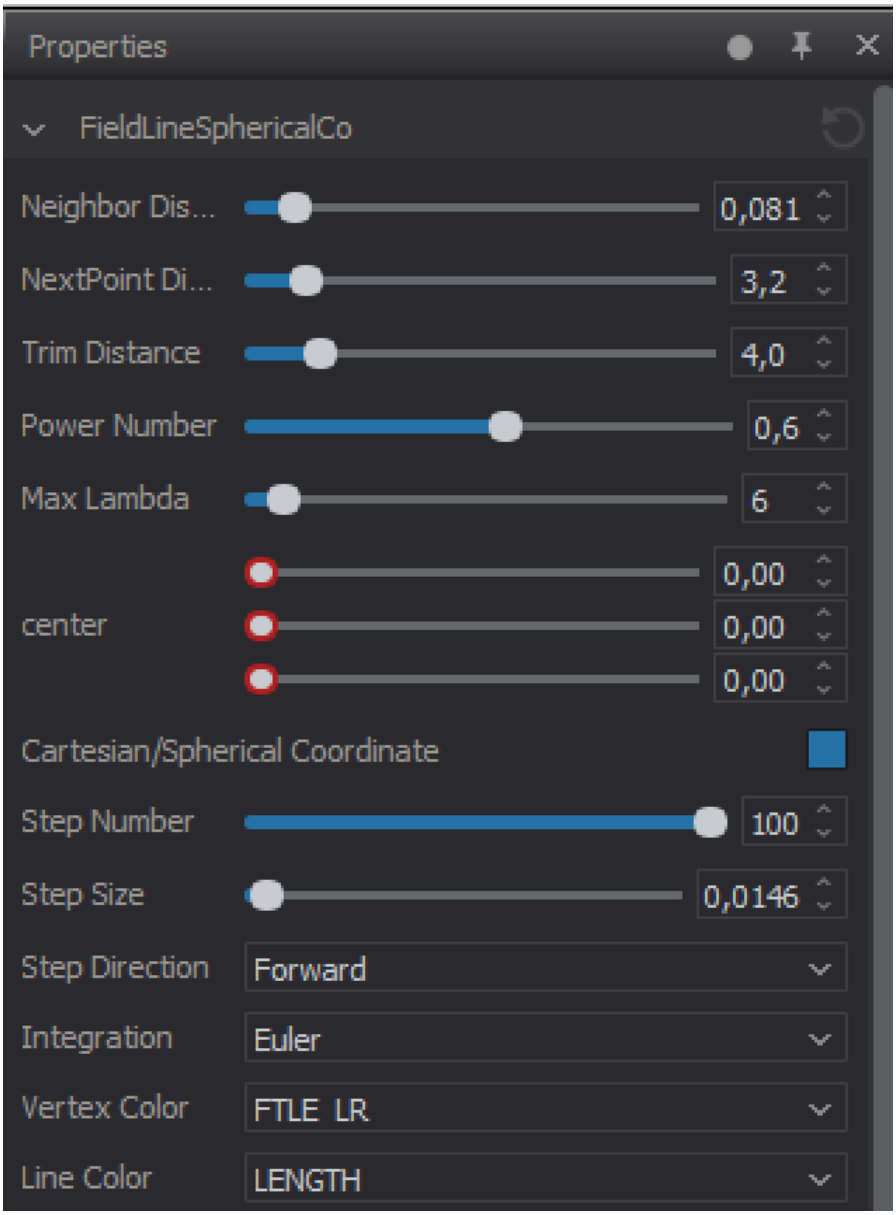

Figure 26: FieldLineSphericalCo processor properties.

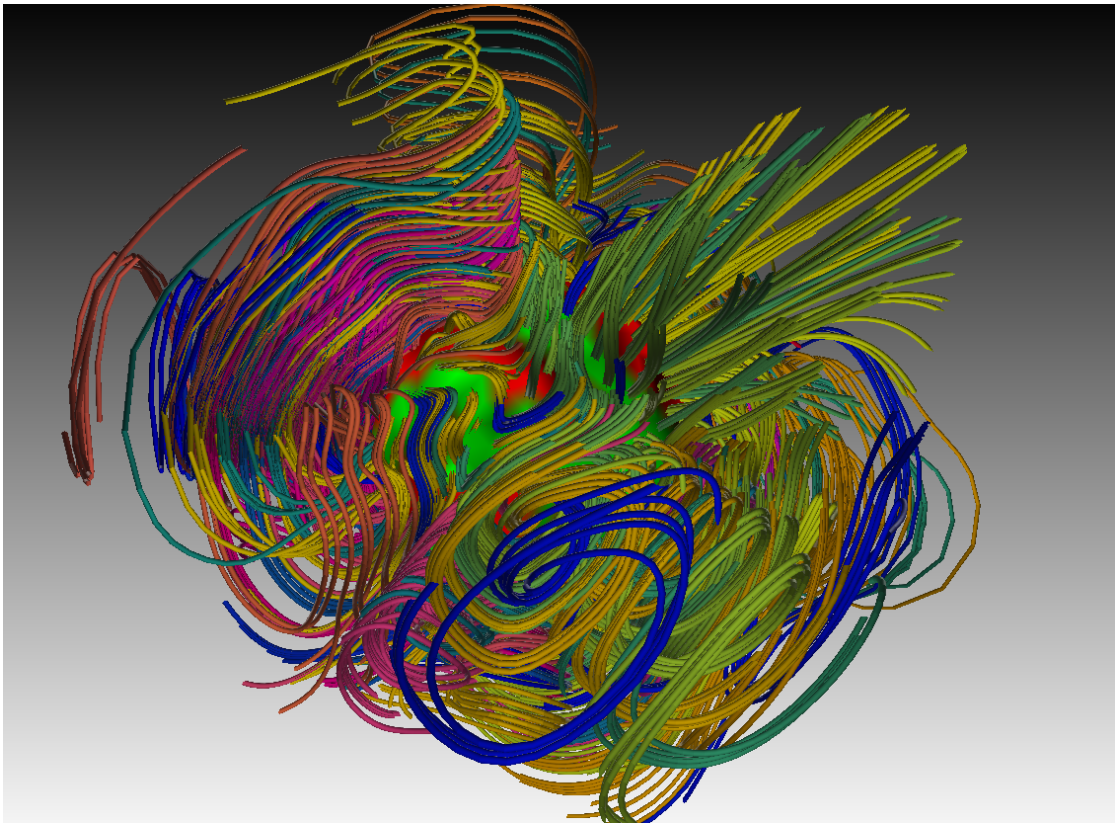

Figure 27: FTLE Values for Selected Points result. Index apply for streamlines color.

6 RESULTS

In this section we represent our final results for each method.

6.1 FTLE Values for Selected Points and FTLE High Resolution

Our result for FTLE values for selected points is shown in Fig. 27 and FTLE high resolution shown in Fig. 28. As you can see if the velocity is more than zero or equal to zero the color for vertices is green and if less than zero the color is red. For velocities more or equal to zero if the streamlines are more divergent the vertex color becomes more bright green. If there is less divergence or convergence the vertex color becomes more dark green and it is the same if velocity is less than zero but instead of green we use red.

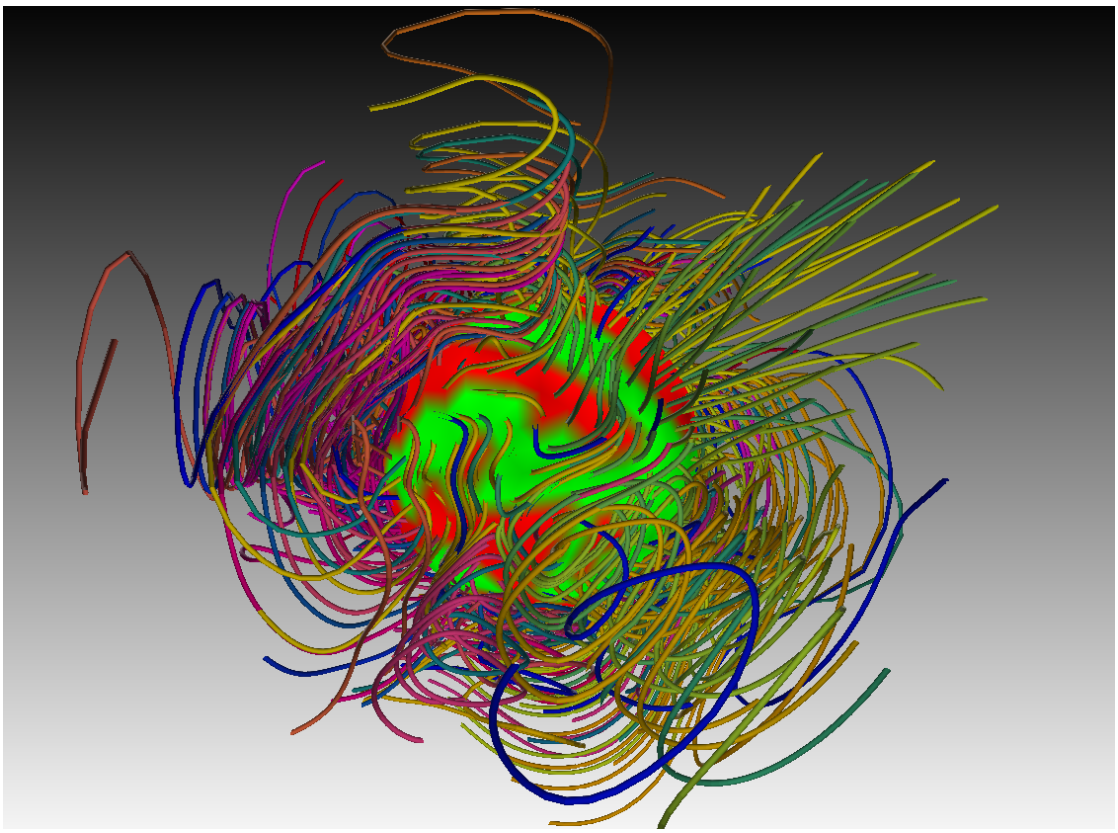

Figure 28: FTLE High Resolution result. Index applied for streamlines color.

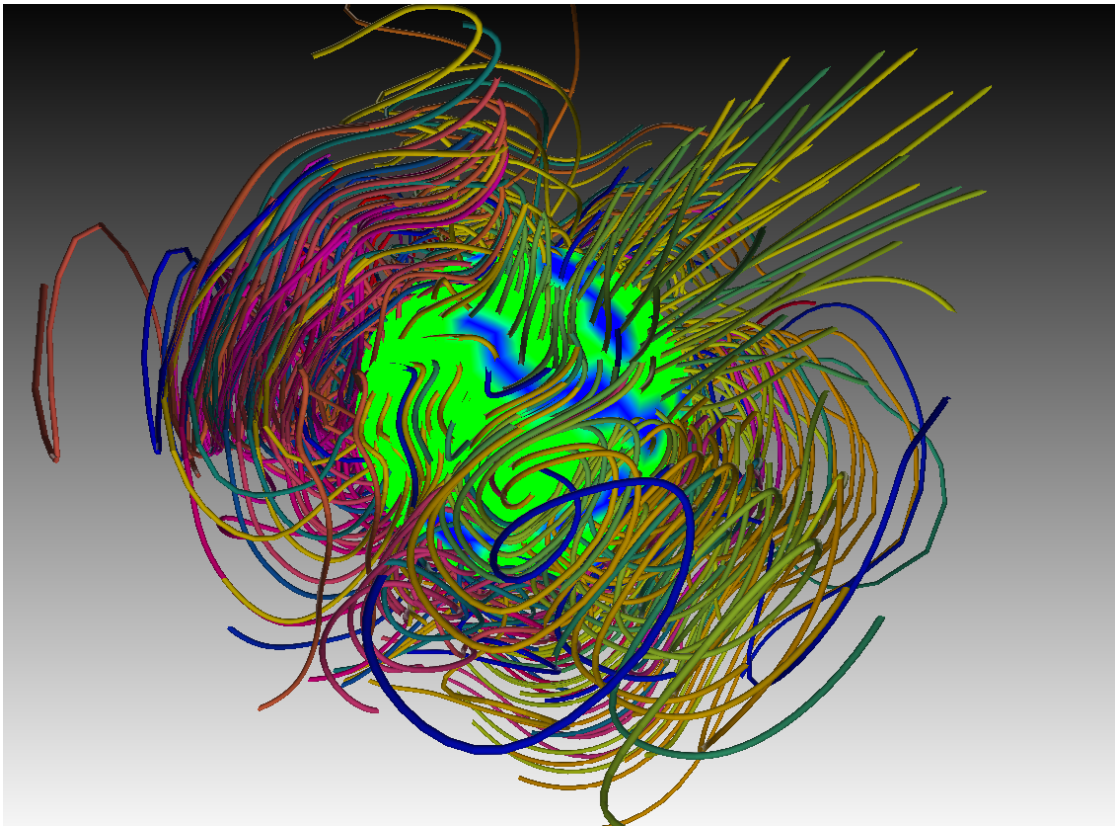

Figure 29: Dot between normal and vector. Index applied for streamlines color.

6.2 Dot between normal and vector

Our result for Dot between normal and vector is shown in Fig. 29. The dot is between seed points normal and vector (subtract between first and second points of streamlines). If dot is more than zero the color of the vertices become green and if less than zero the vertices color become blue.

6.3 Color Lines

For color lines we have four options so we have four results as well. Fig. 30 shows our result for color lines from index which get the streamlines number and from that number get their color low to high from the look up table.
Fig. 31 shows our result for color lines from FTLE. We store the colors from FTLE methods and get them to the streamlines.

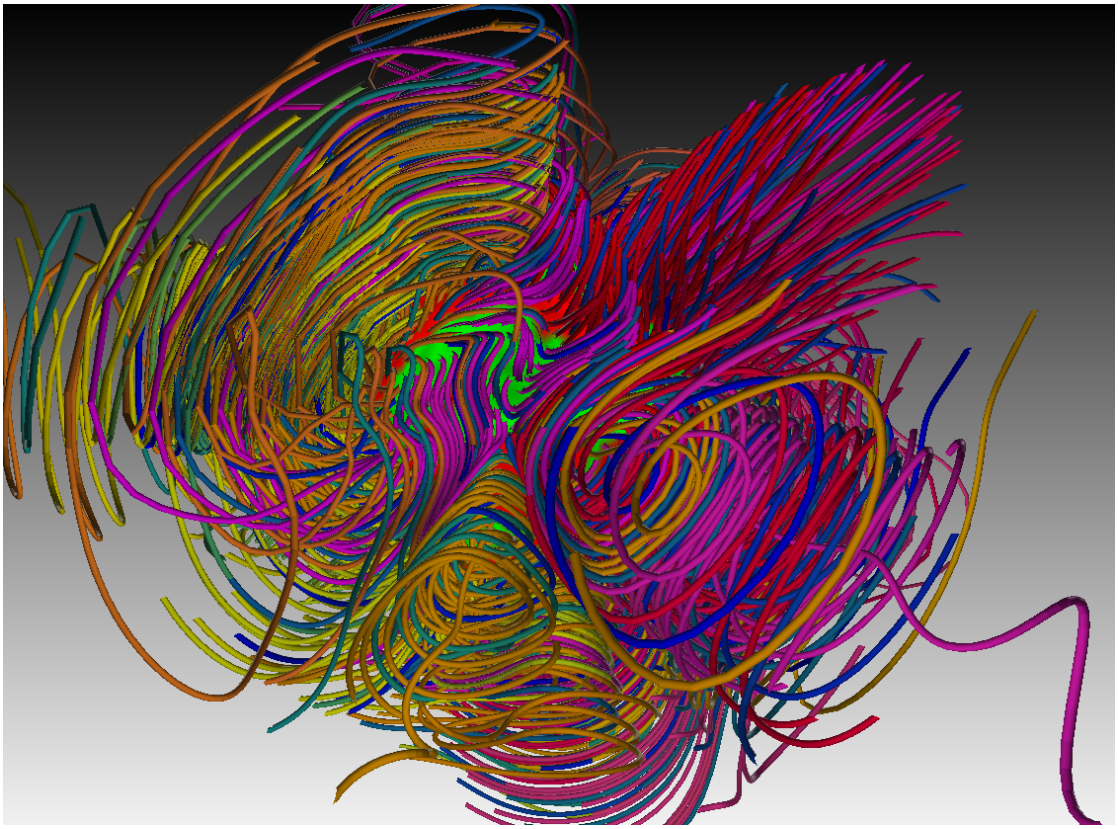

Figure 30: Color Lines from Index.

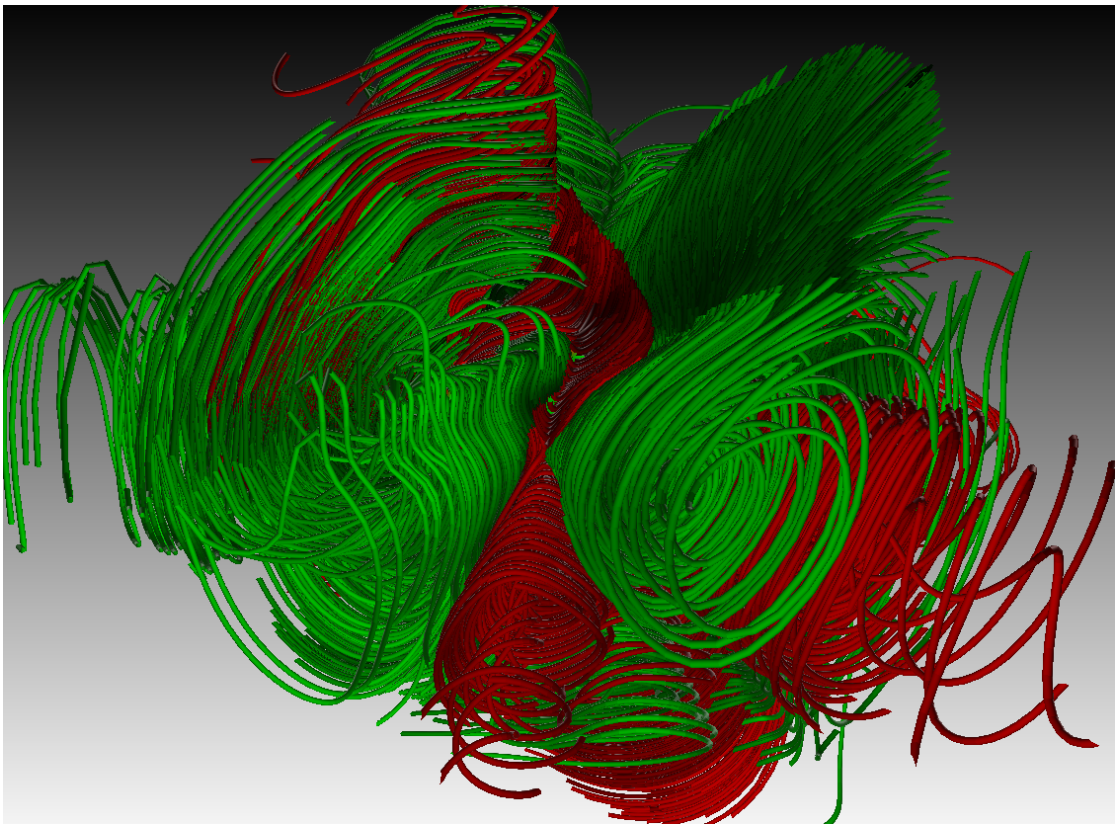

Figure 31: Color Lines from FTLE.

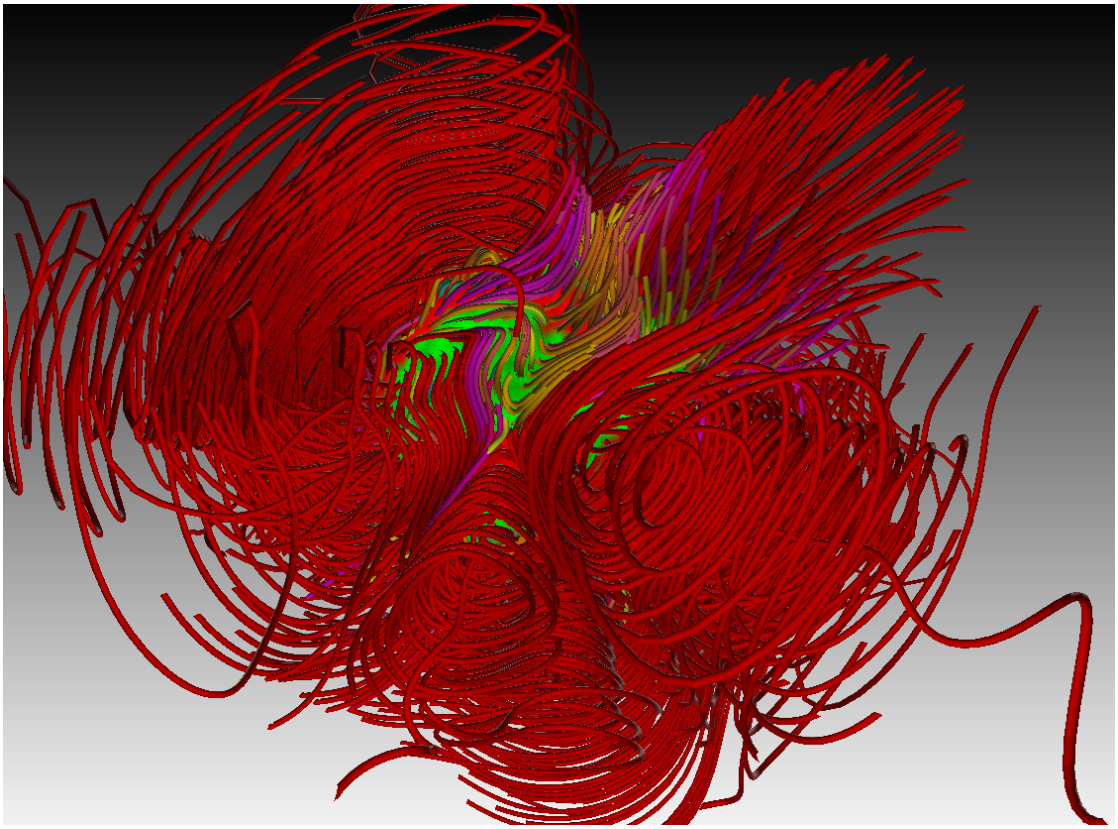

Figure 32: Color Lines from Length.

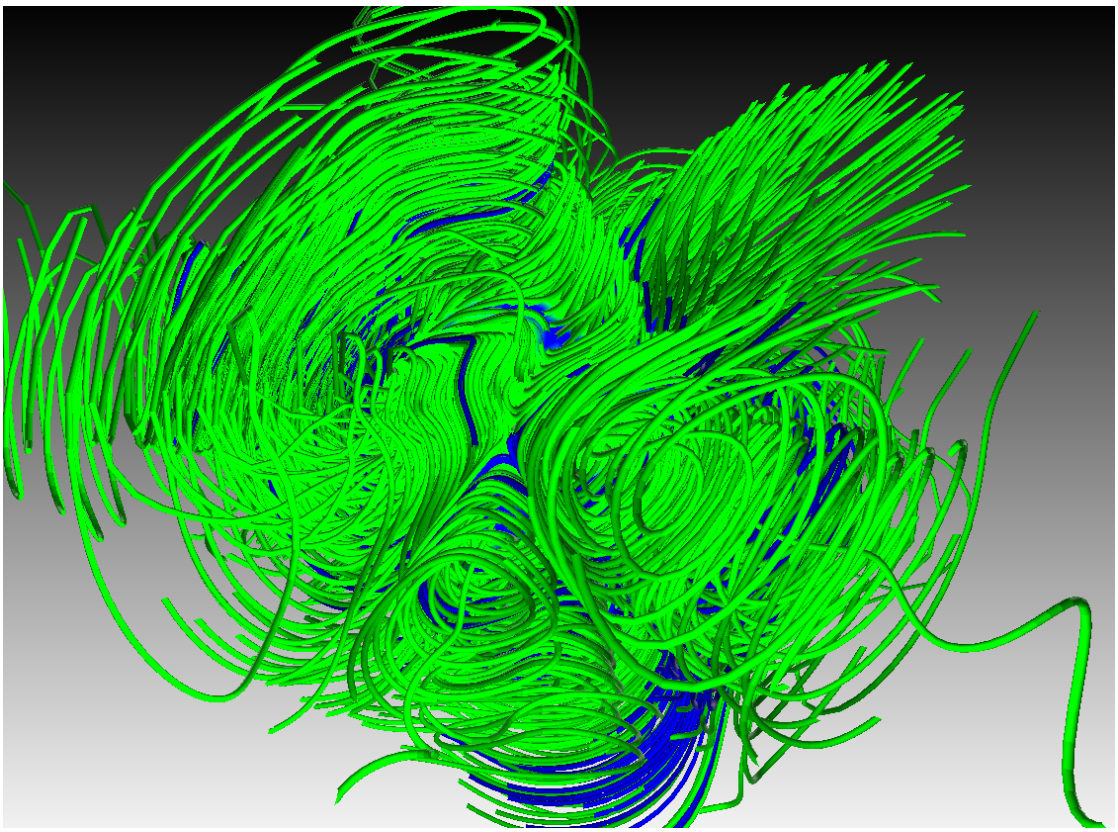

Figure 33: Color Lines from Dot.

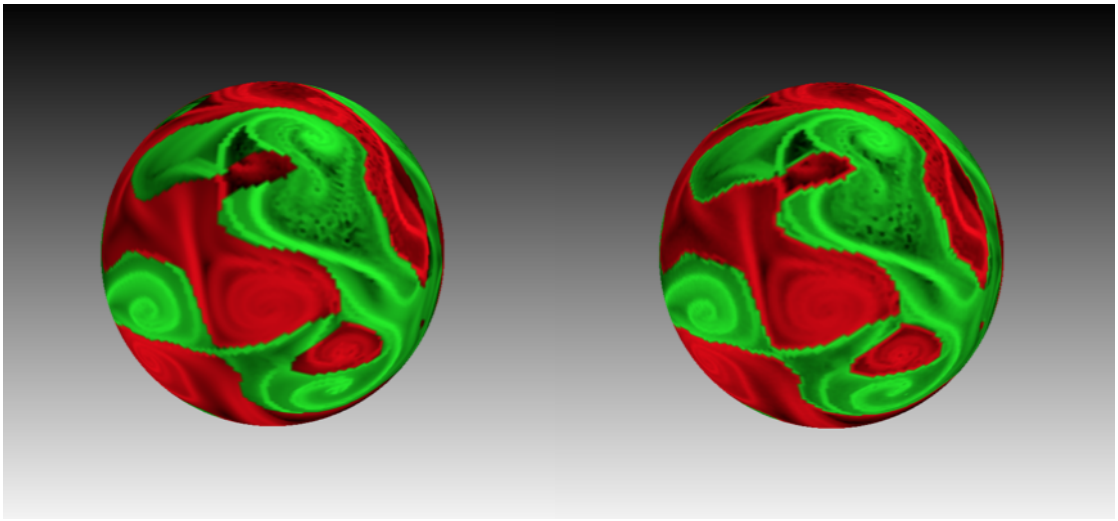

Figure 34: Comparison between FTLE Values for Selected Points(left) and FTLE high resulotion(right).

Fig. 32 shows our result for color lines from Length and Fig. 33 shows our result for color lines from Dot.

6.4 Comparison between FTLE Values for Selected Points and FTLE High Resulotion

Our comparison between FTLE Values for Selected Points and FTLE high resulotion is displayed in Fig. 34. As you can see both of them are almost similar and it is difficult to say which one is more accurate than the other one. For FTLE Values for Selected Points we add four neighbouring points for each seed point but in FTLE high resolution we don't add neighbouring points and instead we use neighbouring seed points for each seed points so computation time for FTLE high resolution becomes faster than FTLE Values for Selected Points. Table.1 compare computation times for FTLE Values for Selected Points and FTLE high resolution in fifth subdivision and sixth subdivision. Another important issue is in FTLE high resolution we have brighter strip around the boundary between the negative velocity and the positive velocity but in FTLE Values for Selected Points that boundary is darker. This problem maybe disappeared if we subdivide more. This is one of our future works, wherein we shall aim to find the problem.

Table 1: Compare computation times for FTLE Values for Selected Points(FTLE VSP) and FTLE high resolution(FTLE HR)

Number of subdivision	Computation time FTLE VSP	Computation time FTLE HR
Fifth	9570.96ms	5639.56ms
Sixth	46160.6ms	38088.8ms

7 CONCLUSION AND FUTURE WORK

Our aim in this work was focused on satisfaction of the scientist and easing the investigation to understanding and analyzing the data and helping them to easily excavate the divergence between these magnetic fields. We visualized the data with streamlines and applied the sphere instead of the Sun. Then measurement of divergence neighboring streamlines of each seed points was conducted using the FTLE method. We first added four neighbouring points for each seed points (FTLE Values for Selected Points) and then we applied neighbouring seed points for each seed point (FTLE high resolution) and finally compared these two methods. We found that, as we explained in the result section,computation time for FTLE high resolution is faster than FTLE Values for Selected Points but it is difficult to say which one is more accurate than the other one. There are some avenues for future work:

1. As we explained in the result section, we can work to find the brighter strip problem in FTLE High Resolution.

2.Another improvements, we can further combine this method with other methods that are not based on Lagrangian structures to combine the beneficial effects of each one.

3.In this work, we have not discussed the influence of integration length on the FTLE calculations.

4.While the application of this method seems straightforward in theory, the computational cost becomes prohibitive and to help solve this issue we need fast algorithms.

5.We can apply CPU instead of GPU to become faster when we subdivide for high resolution.

8 APPENDIX

In this section we will present our functions we used for transformation spherical coordinate to cartesian coordinate and Vice versa , Euler method, Runge-Kutta 4th Order method and FTLE method.

8.1 Transformation function

In this transformation function theta is between -90 to 90 degrees and phi between 0 to 360 degrees.

function CartesianToSpherical {
input: Cartesianposition
output: vector3

r = (sqrt((Cartesianposition.x * Cartesianposition.x) + (Cartesianposition.y * Cartesianposition.y) + (Cartesianposition.z * Cartesianposition.z)));
theta = (arccos(Cartesianposition.z / r)) - (PI / 2) ;
phi = (arctan(Cartesianposition.y / Cartesianposition.x));
return sphericalCoordinate(r, theta, phi); }

function SphericalToCartesian {
input: sphericalposition
output: vector3

theta = (sphericalposition.y) + (PI / 2);
phi = (sphericalposition.z);
x = sin(theta) * cos(phi);
y = sin(theta) * sin(phi);
z = cos(theta);
r = sphericalposition.x;
return cartesianCoordinate(x, y, z) * r; }

8.2 Euler function

functin euler {

input: vetor3 p, vectorfield, stepsize, dir,transformationmatrix, velocity
output: vector3

K1 = **trilinear Interpolation**(vectorfield, p);
K1 = K1 * stepsize;
vel = transformationmatrix * K1;
if (velocity != nullptr) {
velocity[0] = vel.x;
velocity[1] = vel.y;
velocity[2] = vel.z;
}
return p + vel * staticcastfloat(dir);
}

8.3 Runge kutta 4th order function

functin rungekutta4 {

input: vector3 p, vectorfield, stepsize,dir, transformationmatrix, velocity
output: vector3

h = stepsize / 2.f;
K1 = **trilinear Interpolation**(vectorfield, p);
K1 = K1 * stepsize;
newpos = p + h * (transformationmatrix * K1);
if (newpos.x $<$ 0.0) newpos.x = .0;
if (newpos.y $<$ .0) newpos.y = .0;
if (newpos.z $<$ .0) newpos.y = (newpos.y + 1);

```
if (newpos.x  > 1.)  newpos.x = 1.;
if (newpos.y  > 1.)  newpos.y = 1.;
if (newpos.z  > 1.)  newpos.y = (newpos.y - 1);
K2 = trilinear Interpolation(vectorfield, newpos);
K2 = K2 * stepsize;
newpos = p + h * (transformationmatrix * K2);
if (newpos.x  < .0)  newpos.x = .0;
if (newpos.y  < .0)  newpos.y = .0;
if (newpos.z  < .0)  newpos.y = (newpos.y + 1);
if (newpos.x  > 1.)  newpos.x = 1.;
if (newpos.y  > 1.)  newpos.y = 1.;
if (newpos.z  > 1.)  newpos.y = (newpos.y - 1);
K3 = trilinear Interpolation(vectorfield, newpos);
K3 = K3 * stepsize;
newpos = p + stepsize * (transformationmatrix * K3);
if (newpos.x  < .0)  newpos.x = .0;
if (newpos.y  < .0)  newpos.y = .0;
if (newpos.z  < .0)  newpos.y = (newpos.y + 1);
if (newpos.x  > 1.)  newpos.x = 1.;
if (newpos.y  >1.)  newpos.y = 1.;
if (newpos.z  >1.)  newpos.y = (newpos.y - 1);
K4 = trilinear Interpolation(vectorfield, newpos);
K4 = K4 * stepsize;
vel = (transformationmatrix * K1) * (1.f / 6.f) + (transformationmatrix * K2) * (1.f / 3.f) + (trans-
formationmatrix * K3) * (1.f / 3.f) + (transformationmatrix * K4) * (1.f / 6.f);
if (velocity != nullptr) {
velocity[0] = vel.x;
velocity[1] = vel.y;
velocity[2] = vel.z;
}
return p + vel * staticcastfloat(dir);
}
```

8.4 FTLE function

function FTLE {

input: meshsphere
output: No output

for (it = lines.begin(); it != lines.end(); it += 5){
e1 = (**function sphericalToCartesian**((it + 1)/− >GetPoint(Nextpointdistance.get())) - **function sphericalToCartesian**((it + 2) /− >GetPoint(Nextpointdistance.get()))/(2 * Neighborpointdistance.get());
e2 = (**sphericalToCartesian**((it + 3) − > GetPoint(Nextpointdistance.get())) - **sphericalToCartesian**((it + 4) − > GetPoint(Neighborpointdistance.get())) / (2 * Nextpointdistance.get());
mat2x3 A = mat2x3(e1, e2);
mat3x2 AT = transpose(A);
e = util::glm2eigen(A);
EigenSolver(Eigen::MatrixXf) es(e.transpose()*e);

lambda1 = es.eigenvalues()[0];
lambda2 = es.eigenvalues()[1];
lambda3 = es.eigenvalues()[2];

lambda1 = abs(lambda1);
lambda2 = abs(lambda2);
lambda3 = abs(lambda3);

if (lambda1.real() >0){
lambda1 = log(lambda1); }
if (lambda2.real() > 0){
lambda2 = log(lambda2);
}
if (lambda3.real() > 0)
lambda3 = log(lambda3);

```
}
Maxlocallambda = max(lambda1.real(), lambda2.real());
Maxlocallambda = max(lambda3.real(), Maxlocallambda);
if (Maxlocallambda < minlbda){
minlbda = Maxlocallambda;
}
if (Maxlocallambda > maxlbda){
maxlbda = Maxlocallambda;
Maxlocallambda = abs(Maxlocallambda);
}
colorLR = vec4(pow(float(Maxlocallambda / maxlambda.get()), Power), pow(float(Maxlocallambda
/ maxlambda.get()), Power), pow(float(Maxlocallambda / maxlambda.get()), Power), 1);

meshsphere − > setVertexColor(IndexSeedPoint, colorLR);
}
}
```

References

[1] G. Machado, F. Sadlo, T. Muller, T. Ertl, "Escape Maps", IEEE Trans Vis Comput Graph, 20(12):2604-13, 2014.

[2] G. Haller, "Distinguished material surfaces and coherent structures in three-dimensional fluid flows," Physica D, 149:248-277, 2001.

[3] F.Lekien, C. Coulliette, A. J. Mariano, E.H.Ryan, L. K. Shay, G. Haller, and J. Marsden, "Pollution release tied to invariant mainfolds: A case study for the coast of florida," Phys. D, 210(1), 2005.

[4] G. Haller. "Lagrangian coherent structures from approximate velocity data," Physics of Fluids, 14(6):1851-1861, 2002.

[5] S. C. Shadden, F. Lekien, J. D. Paduan, F. P. Chavez, and J. E. Marsden. "The correlation between surface drifters and coherent structures based on high-frequency radar data in monterey bay," Deep Sea Research Part II: Topical Studies in Oceanography, 56(3-5):161172, 2009.

[6] M. Weldon, T. Peacock, G. B. Jacobs, M. Helu, and G. Haller. "Experimental and numerical investigation of the kinematic theory of unsteady separation," Journal of Fluid Mechanics, 611:111, 2008.

[7] S. C. Shadden, F. Lekien, and J. E. Marsden. "Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows," Phys. D, 212(7):271304, 2005.

[8] C. Garth, F. Gerhardt, X. Tricoche, and H. Hagen. "Efficient Computation and Visualization of Coherent Structures in Fluid Flow Applications," IEEE Transactions on Visualization and Computer Graphics, 13:1464 1471, 2007.

[9] C. Garth, G.-S. Li, X. Tricoche, C.D. Hansen, and H. Hagen "Visualization of Coherent Structures in Transient 2D Flows," Proc. Topology-Based Methods in Visualization II, Math. + Visualization, pp. 1-14, 2009.

[10] Natural Magnetism
http://www.naturalmagnetism.com/earths-magnetic-field.html

[11] SDO | Solar Dynamics Observatory
http://sdo.gsfc.nasa.gov/mission/spaceweather.php

[12] S.Barakat, C.Garth, X.Tricoche,"Interactive Computation and Rendering of Finite-Time Lyapunov Exponent Fields", IEEE Trans Vis Comput Graph,18(8):1368-80, 2012.

[13] S. Shadden, F. Lekien, and J. Marsden, "Lagrangian Coherent Structures in n-Dimensional Systems," J. Math. Physics, vol. 48, p. 065404, 2007.

[14] S. Shadden, J. Dabiri, and J. Marsden, "Lagrangian Analysis of Fluid Transport in Empirical Vortex Ring Flows," Physics of Fluids, vol. 18, p. 047105, 2006.

[15] G. Haller, "Lagrangian Structures and the Rate of Strain in a Partition of Two-Dimensional Turbulence," Physics of Fluids, vol. 13, no. 11, pp. 3365-3385, 2001.

[16] M. Green, C. Rowley, and G. Haller, "Detection of Lagrangian Coherent Structures in 3D Turbulence," J. Fluid Mechanics, vol. 572, pp. 111-120, 2007.

[17] K. Matkovic, D. Gracanin, Z. Konyha, and H. Hauser, Color Lines View: An Approach to Visualization of Families of Function Graphs, Proc. Information Visualization 07, pp. 59-64, 2007.

[18] F. Sadlo and R. Peikert, "Efficient Visualization of Lagrangian Coherent Structures by Filtered AMR Ridge Extraction," IEEE Trans. Visualization and Computer Graphics, vol. 13, no. 6, pp. 1456- 1463, 2007.

[19] F. Sadlo and A. Rigazzi, "Time-Dependent Visualization of Lagrangian Coherent Structures by Grid Advection," Proc. Topological Methods in Data Analysis and Visualization (TopoInVis 09), pp. 151-165, 2009.

[20] S.L. Brunton and C.W. Rowley, "Fast Computation of Finite-Time Lyapunov Exponent Fields for Unsteady Flows," Chaos: An Interdisciplinary J. Nonlinear Science, vol. 20, no. 1, p. 017503, 2010.

[21] M. Hlawatsch, F. Sadlo, and D. Weiskopf, "Hierarchical Line Integration," IEEE Trans. Visualization and Computer Graphics, vol. 17, no. 8, pp. 1148-1163, 2011.

[22] K. Burger, P. Kondratieva, J. Kr uger, and R. Westermann. "Importance- Driven Particle Techniques for Flow Visualization," In Proceedings of IEEE VGTC Pacific Visualization Symposium 2008, pages 7178, 2008.

[23] K.H. Hhne, B.Pflesser, A.Pommert, M.Riemer, R.Schubert, T.Schiemann, U.Tiede, U.Schumacher, "A Realistic Model of the Inner organs from the Visible Human Data", Germany, 2000

[24] Inviwo
http://www.inviwo.org/overview/

[25] Community Coordinated Modeling Center
http://ccmc.gsfc.nasa.gov/models/modelinfo.php?model=ENLIL

[26] D. Odstrcil, "Modeling 3-D Solar Wind Structure," Advances in Space Research, vol. 32, no. 4, pp. 497-506, 2003.

[27] D. Mackay and A. Yeates, "The Suns global photospheric and coronal magnetic fields: Observations and models", Living Reviews in Solar Physics, 9(6), 2012.

[28] R. Englund, T. Ropinski, and I. Hotz ,Coherence Maps for Blood Flow Exploration, Eurographics Workshop on Visual Computing for Biology and Medicine, 2016.

[29] Schmidt, H. U., Mitt. Astron. Ges., Sept, 89, 1965.

[30] Schmidt, H. U., in W. N. Ness (ed.), Solar Flares, AAS-NASA Symp., NASA SP-50, p.107, 1964.

[31] Rust, D. M.Thesis, University of Colorado, 1966 .

[32] Rust, D. M., AIAA Observations and Predictions of Solar Activity Conf., Huntsville, AL), 1970.

[33] Rust, D. M., and Roy, J.-R., in R. Howard (ed.), Solar Magnetic Fields, IAU Symp. 43, Reidel, Dordrecht, Holland, p. 569, 1971.

[34] Semel, M., Ann. Astrophys. 30, 513, 1967.