Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Studienarbeit Nr. 2396

Glyph-Based 2D Flow Visualization

Hajun Jang

Studiengang:	Informatik
Prüfer:	Prof. Dr. Daniel Weiskopf
Betreuer:	Dipl.-Inf. Marcel Hlawatsch
begonnen am:	8. Januar 2013
beendet am:	10. Juli 2013
CR-Klassifikation:	I.3.3

Contents

Chapter 1

Introduction

Flow visualization is an important topic in scientific visualization. The goal of this work is to visualize 2-dimensional vector fields with Flow Radar Glyphs and field lines, mainly with pathlines.

In chapter 2 methods like sparklines and small multiples are introduced, which are simple forms of visualizing varying scalar data of an item. Definitions and coding scheme of general glyphs are explained.

In chapter 3 fundamental knowledge and techniques will be briefly explained. The idea of Flow Radar Glyphs (by Hlawatsch, see [HLNW11])is to visualize time-dependent data into a compact glyph. A simple scheme is illustrated in this chapter. Then, mathematics used in computing will be explained.

Chapter 4 shows the functionalities and usage of the application. More than one pathlines in a data domain depicted in original size can overlap many times, therefore it is hard to keep them visually separated. In this work, the idea to solve this problem is to implement downscaled pathlines. Another attempt to demonstrate the characteristics of pathlines, is to apply the Flow Radar Glyphs to tangents of pathlines(this will be referred to as 'tangential glyphs'). Downscaled pathlines and tangential glyphs are the 2 focuses of this work.

Chapter 5 tries to analyze these new visualization methods. This is done by analyzing them on a global and zoomed in view. A region of interest will be investigated closer to find visual informations that this visualization technique yields. Tangential glyphs will be shown in comparison to its original pathlines and its differences and similarities are examined. In the first data set, some pathlines run to the edge of domain, making the pathlines stop. This results in smaller tangential glyphs, making the evaluation of the method of tangential glyphs unfavorable. To examine tangential glyphs of pathlines with same length, A second data set is introduced. The second data set is examined in an Attempt to discover features that were not visible in the first data set, especially concerning the tangential glyphs.

In chapter 6, the results of this work comes to a conclusion about downscaled pathlines and tangential glyphs. Though the data sets are limited and a proper user evaluation is not executed, this conclusion seems to hold for this implementation and data sets shown.

Flow Radar Glyphs and field lines can be extended to increase visual perception and efficiency in field analysis. These ideas are proposed in Future Works.

Chapter 2

Related Work

Flow Radar glyphs and downscaled pathlines depict data in multiple lines. As an example of depicting data in lines, Sparklines and small multiples can be mentioned.

2.1 Sparklines

Sparklines 2.1 are line charts with typographic resolution, typically drawn without axis or coordinates. Thus, sparklines make the representation of the variation (typically over time) in measurements or data compact and simple. Whereas traditional charts aim to show all data available, and is separated from the text where it is referenced, sparklines are intended to be small, memorable, and located where they are discussed. Like the original sparkline 2.1 in Tufte's book[Tuf06], the most common data illustrated with sparklines are temperature or stock market price. Sparklines are meant to be used in line with text, placed at about the same height as the text before and after it.

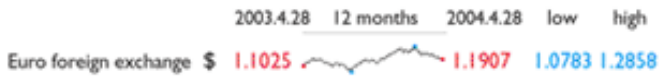

Figure 2.1: Sparkline applied to economic data, showing changes over time. This one shows overall trend as well as some local detail for colored spots. This sparkline depicts the euro exchange rate in reference to US dollar for every day, over a year. [Tuf06]

Sparklines and Flow Radar Glyphs have in common that they are a depiction of sequential numerical data in a 2-dimensional line.

2.2 Small Multiples

Multiple series of data can be plotted within the same axes. However, placing multiple series in the same space may produce overlapping curves that reduce legibility. An alternative approach is to use small multiples. Small multiples show each series of data in its own chart(see Fig.2.2). Grouping series of data into a category and aligning charts in a row or column is made apparently easy using small multiples. Small multiples can be constructed for many type of visualization: bar charts, pie charts, maps, glyphs, etc. This often produces a more readable visualization than trying to combine all the data into a single plot. Also, several Sparklines may be grouped together as elements of a small multiple. Flow Radar

Glyphs and downsized pathlines have their own axis and are depicted separately from each other, like small multiples. But in contrast to small multiples, Flow Radar Glyphs and downsized pathlines have explicit meaning in line starting point, that is where the data is been sampled.

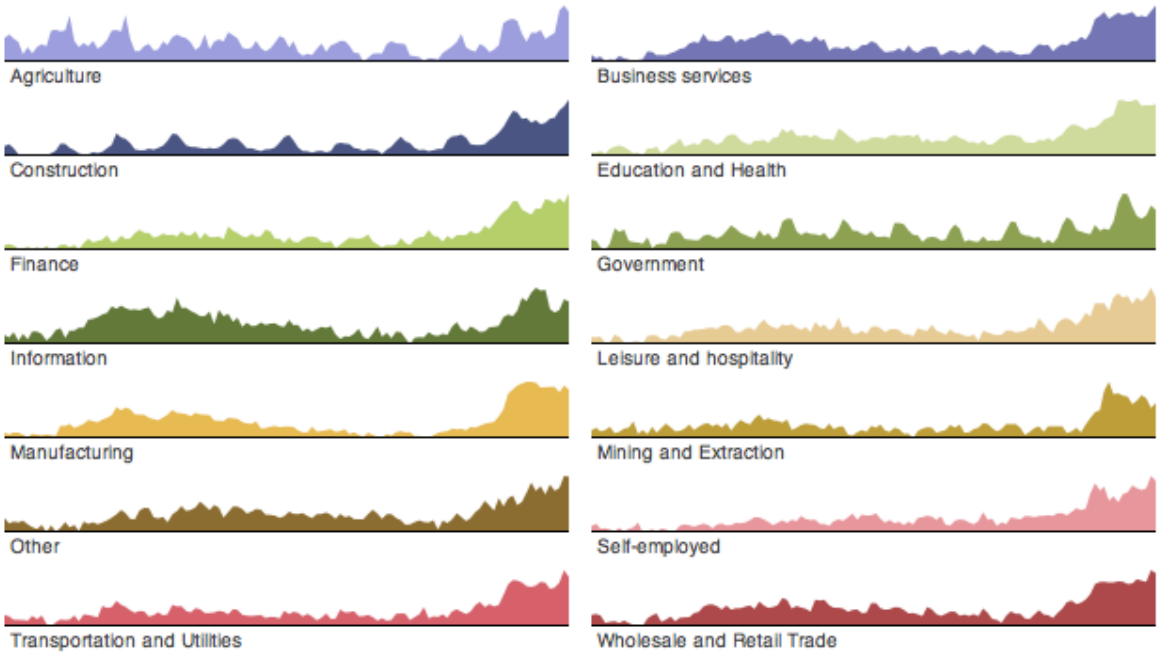

Figure 2.2: Small Multiples showing a series of data, each in its own chart. In this case, we see the number of unemployed workers in the US for the time period from 2000 to 2010, but normalized within each industry category[Hee]. It is possible to see both overall trends and seasonal patterns in each industry sector and accurately compare them.

2.3 Glyphs in Visualization

Glyph is a commonly used visual object in visualization. In data visualization using glyphs, data is depicted by a collection of glyphs. A glyph is a small independent visual object that depicts attributes of a data. Glyphs are discretely placed in the display space. In many visualizations, the spatial location of each glyph is predetermined by the underlying spatial structure encoded in the data, such as a vector field in scientific visualization or geographic information. Glyphs are a type of visual sign but differ from other types of signs such as icons and symbols. Numerical data can be encoded in Glyphs, which is not possible with icons or symbols.

The design of glyphs can make use of many different visual channels such as shape, color, texture, size, orientation, aspect ratio or curvature. (Borgo et al. [BKC+12] lists and categorizes possible visual channels (see Tab. 2.1). Also, MacEachren [Mac04] defined 'visual variables', a syntax for implementation of graphical representations. (see Fig.2.3) Multiple visual channels enable the encryption

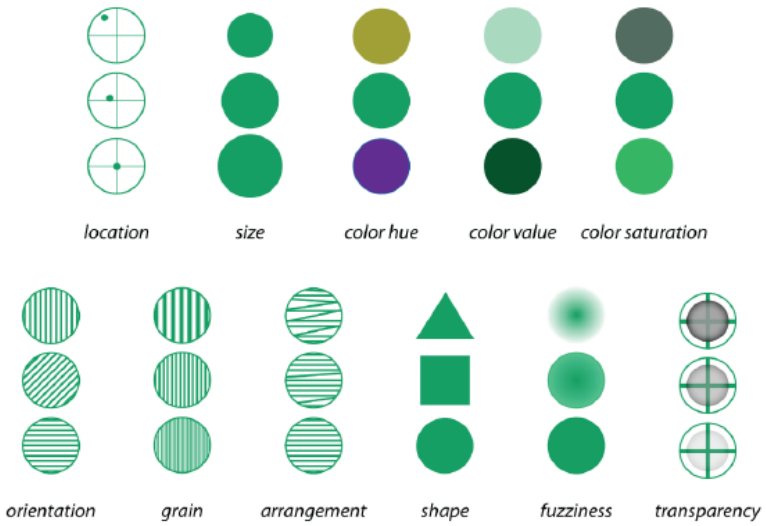

Figure 2.3: Information can be expressed through signs making use of an appropriate syntax. Displayed above are called primitive parameters, or fundamental visual variables according to MacEachren[Mac04].

of multi-dimensional data attributes. A specific design of a glyph is a visual coding scheme. Like all coding schemes, a well-designed glyph-based visualization can facilitate efficient information encoding and visual communication.

2.4 Flow Visualization

McLaughlin et al. [MLP+10] summarizes and categorizes exiting flow visualizing techniques. Techniques dealing with 2D integral curves and particle tracing, which is implemented in this work, is also listed and discussed.

Geometric Channels	Optical Channels	Topological and Relational Channels	Semantic Channels
size / length / width / depth / area / volume	intensity / brightness	spatial location	number
orientation / slope	color / hue / saturation	connection	text
angle	opacity / transparency	node / internal node / terminator	symbol / ideogram
shape	texture (partly geometric)	intersection / overlap	sign / icon / logo / glyph / pictogram
curvature	line styles (partly geometric)	depth ordering / partial occlusion	isotype
smoothness	focus / blur / fading	closure / containment	
	shading and lighting effects	distance / density	
	shadow		
	depth (implicit / explicit cues)		
	implicit motion / motion blur		
	explicit motion / animation / flicker		

Table 2.1: Visual channels that can be utilized when designing glyphs. By Borgo et al. [BKC+12]

Chapter 3

Fundamentals

In this chapter, theories and techniques used in later chapter are explained. The topics are principles and examples of related visualization techniques and theoretical basis of mathematics.

3.1 Overview of Flow Visualization Methods

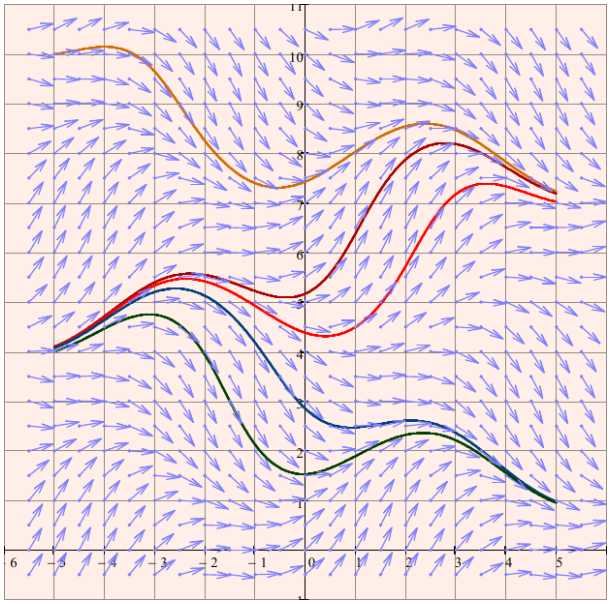

Figure 3.1: The traditional technique of arrow plots is a well-known example of direct flow visualization based on glyphs. Arrows scaled in accordance with the magnitude of the vector are drawn at discrete grid points. It shows the direction of the flow. The colored lines are streamlines. [Wer13]

Visualization method Fig. 3.1 is an arrow plot of a 2D steady vector field(or a Snapshot of a time-dependent field) with uniform grid. As similar techniques, hedgehogs(depiction of the flow by directed line segments scaled to vector magnitude) or streamlets(visualizing short streamlines) can be mentioned. Arrow plots can be directly applied to time-dependent vector fields by letting the glyph depict the velocity field of the time instance. The visualization has to be animated over the time range of the data. Another option is an user interaction to navigate over time. More complex glyphs(see [LW93]) can be

used to show Jacobian of the velocity field and many more properties of 3D vector fields like divergence, rotation, curvature, velocity, acceleration and shear.

3.1.1 Issues to be Considered

The dimensionality of the data is an important issue. Many 2D visualization methods are not apt to be used in 3D problems without adaption, because of occlusion and recognizing position and direction of the visual element. The dimensionality also affects performance. A 3D Visualization technique puts a heavier load on computing power. This should be considered more carefully if interactive navigation through data domain is implemented.

Another factor is time-dependency. Time can be considered as another dimension depending on representation. A steady flow can be regarded as time-independent data. Steady flow is usually much less demanding concerning computation and memory, especially for visualization of field lines, since streamlines, pathlines, and streaklines are identical.

In a densely seeded data domain with uniform sampling, distracting patterns can arise(known as Moire pattern). to avoid this pattern, sampling positions can be slightly jittered by random variations(see [HLNW11]). In this paper, this perturbation of seed points is not implemented.

3.2 Flow Radar Glyphs for Time-Dependent Vector Fields

The primary representation of the vector field in this paper is performed with flow radar glyphs, coming from Hlawatsch et al. [HLNW11].

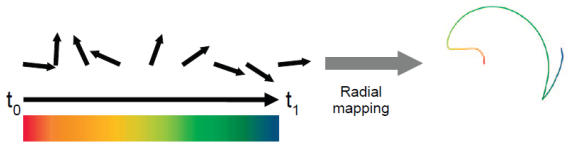

Figure 3.2: Arrows show the time-dependent vectors of one sampling point in the domain. Right is the resulting Flow Radar glyph. Every Flow Radar Glyph has its local coordinates, matching time in growing radius and direction of the vector in angle. Along with radius, color mapping is used to indicate the time. Image from [HLNW11].

An Option to normalize glyphs, that is to make the size independent of the magnitude of field vectors, is included in the implementation. Principle of Flow Radar Glyphs are explained in Fig.3.2. This is a contrived Flow Radar Glyph, not applied on actual data. Flow Radar Glyphs applied on data set is shown in Fig.5.1. The color mapping for Flow Radar Glyphs in Fig.5.1 is also used in field lines and tangential glyphs of field lines in the next chapter. If a vector field contains uncertainty, glyphs can be modified to show this uncertainty (see [HLNW11]).

3.3 Field Lines

Streamlines, pathlines and streaklines are field lines showing the fluid flow of a vector field.

3.3.1 Streamlines

Streamlines are a family of curves that are tangent to the velocity vectors of the flow at a time instance. These show the direction a fluid element will travel at the time instance. It is the most direct field line representation of the vector field.

3.3.2 Pathlines

Pathlines are trajectories that individual fluid particles follow. These can be thought of as 'recording' the path of a fluid element in the flow over a certain time period. The direction the path takes will be determined by the vector of the fluid at each moment in time.

3.3.3 Streaklines

Streaklines are the collection of positions of all the fluid particles that have passed continuously through a particular spatial point(that is the seed point for this streakline) in the past. Dye steadily injected into the fluid at a fixed point extends along a streakline. we can think of this dye as massless particles that is let go into the field successively at regular time steps(ideally infinitesimal).

3.3.4 Comparison of Field Lines

In an unsteady vector field, stramlines, pathlines and streaklines are different. This is demonstrated in Fig. 3.3 for a changing vector field according to time t_0, t_1 and t_2.

Streamlines provide a snapshot of flow field, whereas streaklines and pathlines depend on the time-history of the flow. Dye line is a different name for streakline. Streaklines can be interpreted as dye released gradually from a fixed location during time. In steady flow (when the velocity vector-field does not change with time), the streamlines, pathlines, and streaklines coincide.

By definition, different streamlines at the same instance in a flow do not intersect, because a fluid particle cannot have two different velocities at a position. But pathlines are allowed to intersect themselves or other pathlines.

3.3.5 Integration of Field Lines

Computing the field lines of the previous section is the problem of calculating the shape of an unknown curve which starts at a given point and satisfies a given differential equation. In this work, Runge Kutta 4(further referred to as RK4) is used as integration method. RK4 can be seen as a reasonable trade-off between computation time and accuracy.

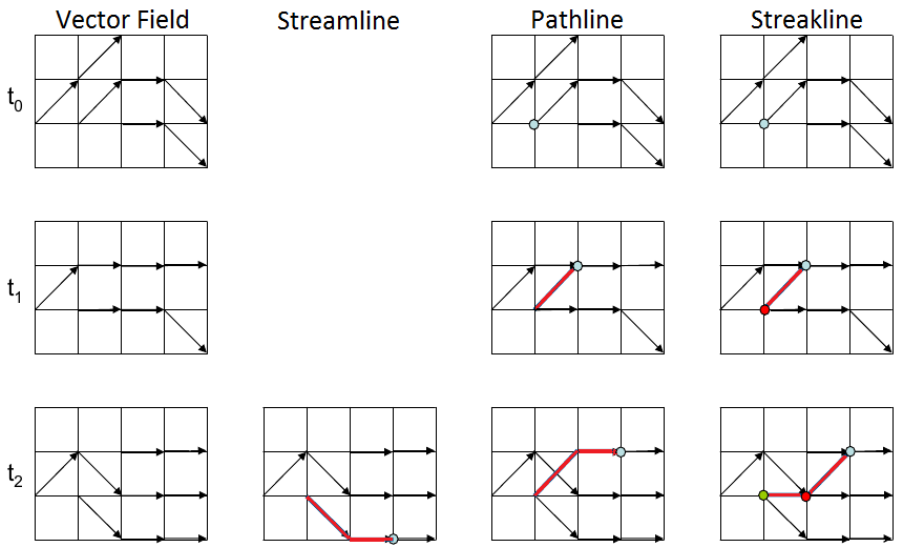

Figure 3.3: streamline, pathline and streakline in comparison. streamline in this illustration is drawn only for t_2, but can be recorded for all instances of time. pathline is the trajectory of a particle over the time range of interest, streakline likewise over a time range is a collection of particles that left the seed point in an earlier time. [Zil13]

The RK4 method can be described in Butcher table, where the time steps(relative to the entire time step, h) and weights on the increments are shown in a matrix.

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

vertical axis shows the time steps and horizontal axis the weight on the increments

$$y_{n+1} = y_n + \tfrac{1}{6}h\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{3.1}$$
$$t_{n+1} = t_n + h \tag{3.2}$$

for n = 0, 1, 2, 3, . . . , using

$k_1 = f(t_n, y_n)$,

$k_2 = f(t_n + \frac{1}{2}h, y_n + \frac{h}{2}k_1)$,

$k_3 = f(t_n + \frac{1}{2}h, y_n + \frac{h}{2}k_2)$,

$k_4 = f(t_n + h, y_n + hk_3)$.

Here y_{n+1} is the RK4 approximation of $y(t_{n+1})$, and the next value (y_{n+1}) is determined by the present value (y_n) plus the weighted average of four increments, where each increment is the product

of the size of the interval h, and an estimated slope specified by function f on the right-hand side of the differential equation. Formulas are from [Wik13]

k_1 is the increment based on the slope at the beginning of the interval, using $\dot{y}$

k_2 is the increment based on the slope at the midpoint of the interval, using $\dot{y} + \frac{1}{2}hk_1$

k_3 is again the increment based on the slope at the midpoint, but now using $\dot{y} + \frac{1}{2}hk_2$

k_4 is the increment based on the slope at the end of the interval, using $\dot{y} + hk_3$

In averaging the four increments, greater weight is given to the increments at the midpoint. The RK4 method is a fourth-order method, meaning that the error per step is on the order of $O(h^5)$, therefore the total accumulated error has order $O(h^4)$.

3.4 GPGPU and Compute Shader

GPGPU stands for 'General-Purpose computing on the GPU'. This approach of programming aims to use the computing power of Graphics Processing Units (further referred to as GPUs) instead of the CPU. Since GPUs are innately suitable for parallel programming with threads, GPGPU approach is often used in visualization of data.

The partitioning of a computation problem in threads is defined in number of workgroups, for GLSL compute shader. Fig. 3.4 shows how the numbers of workgroups can be defined. Depending on GPUs, some combinations of number of workgroups can be favorable (in terms of computation time) for GPUs and some other combinations can be not efficient than others. Then on the compute shader side, each instantiation of thread has its identification numbers of invocation in several built-in read-only variables.

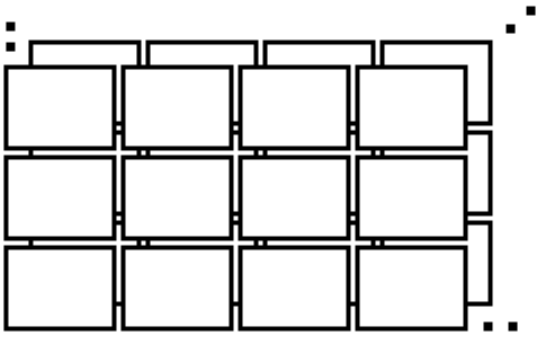

Figure 3.4: The 3D grid of work groups for GLSL compute shader. In this case, num_groups_x is 4, num_groups_y is 3 and num_groups_z is set to 2.

Chapter 4

Implementation

The application is implemented in C++, using OpenGL and GLSL compute shader. In the first place, the data sets were loaded into texture memory, in order to be accessed from the shader that can compute interpolations between seeding points using texture coordinates. The regular grid, on which the data is defined, makes the implementation in threads comfortable. One thread is responsible for the representation of entire data in one sampling point, may it be glyphs or field lines. All sampling points can be handled simultaneously and all threads write computed data in its implicitly assigned part of SSBO(Shader Storage Buffer Objects). A separate shader is employed for highlighting the field line pointed with mouse pointer. When the mouse pointer is moved and a new field line is pointed, only this shader has to be executed. For user interaction and instantaneous variable change and apply, AntTweakBar is used.

4.1 Flow Radar Glyphs

Given as the primary data set is the 'buoyancy' data set. This data set comes from simulations. (for more information, see[HLNW11].) It is sampled on a 41*41 2D domain, over 321 time steps. This can be seen as a 3D data set, which will be loaded to 3D texture and read from the shaders. In Fig.4.1 Flow Radar Glyphs seeded on a 99*99 grid are shown.

In the implementation, number of sampling points are not fixed. It can be interactively changed and the shader can compute the new vertices on the fly. This interaction is possible for many other variables for which it makes sense to have the functionality.

4.2 Visualization of Field Lines

4.2.1 Streamlines and Streaklines

Streamlines show the vector field at a time instant. To analyze field changes over time, multiple streamlines of different time instant must be compared. Fig.4.2 shows snapshots of streamlines, at regular interval. Fig.4.3 shows streaklines starting at time 0. Streamlines don't need to be downscaled at seeding points. They don't intersect.

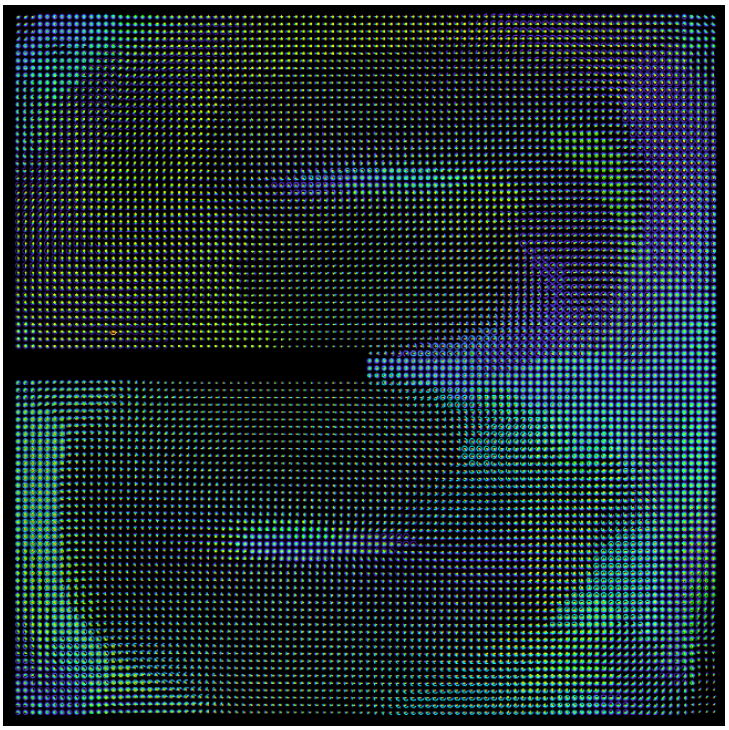

Figure 4.1: 99 by 99 Flow Radar Glyphs seeded on a regular grid. Patterns are visible when neighboring glyphs are of similar shape.

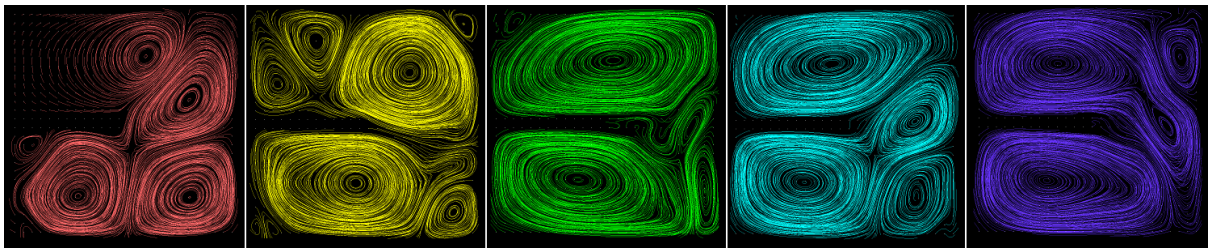

Figure 4.2: Streamlines of data set 'buoyancy' at time t= 0.1, 0.3, 0.5, 0.7, 0.9 (time runs from 0.0 to 1.0). Same color coding as Flow Radar Glyphs(see Fig.3.2) is used here.

4.2.2 Downscaling Field Lines at each Seeding Points

Visualizing Pathlines, we can quickly see that depicting more than one pathline(see Fig. 4.4) is hardly viable. Following a pathline is obstructed through other pathlines interfering each other.

We can try to choose one pathline and highlight this pathline. The seeding points on regular grid are the starting points for each pathline. If the mouse pointer points to a seeding point(to which the distance is the smallest), the pathline starting at the pointed seeding point can be marked with a different color as in Fig.4.5. Favorably, a color that is not present in the color coding for other pathlines would

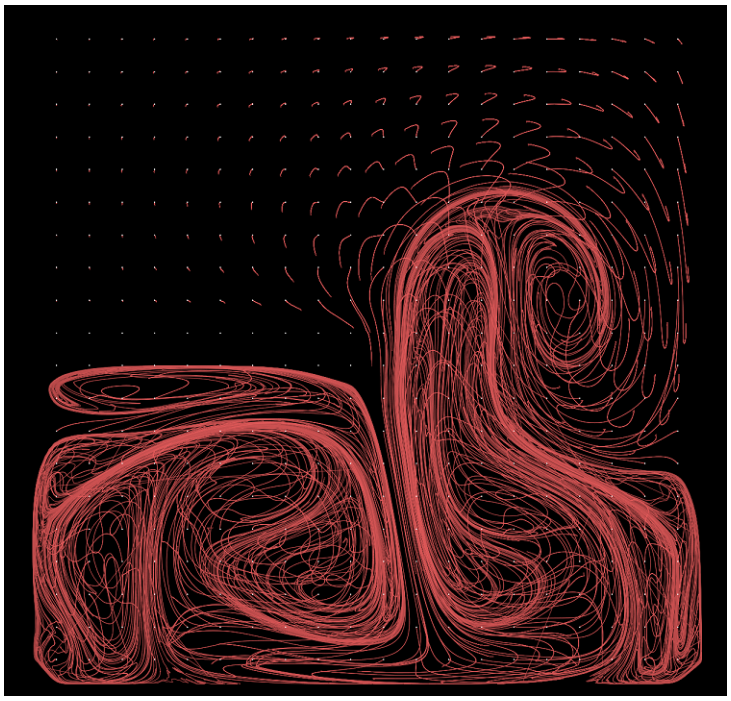

Figure 4.3: Streaklines of data set 'buoyancy'. Problem with partly straight lines is a common issue for visualizing streaklines. This happens where the points on streaklines change abruptly between time steps.

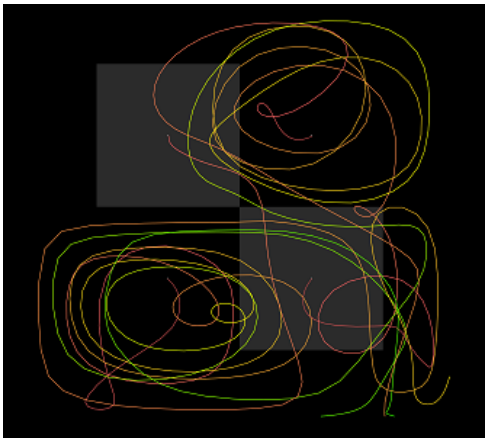

Figure 4.4: 2x2 pathlines depicted in original size. Tracking a single pathline is not easy.

enhance visibility of the highlighted pathline. The width of line is increased to emphasize this line. In this method of highlighting pointed pathline with a uniform color, the color coding of the highlighted pathline is lost. Furthermore, examining more than one pathline at the same time is still not possible.

One of the focuses in this paper is to solve this problem of visibility of pathlines. The solution

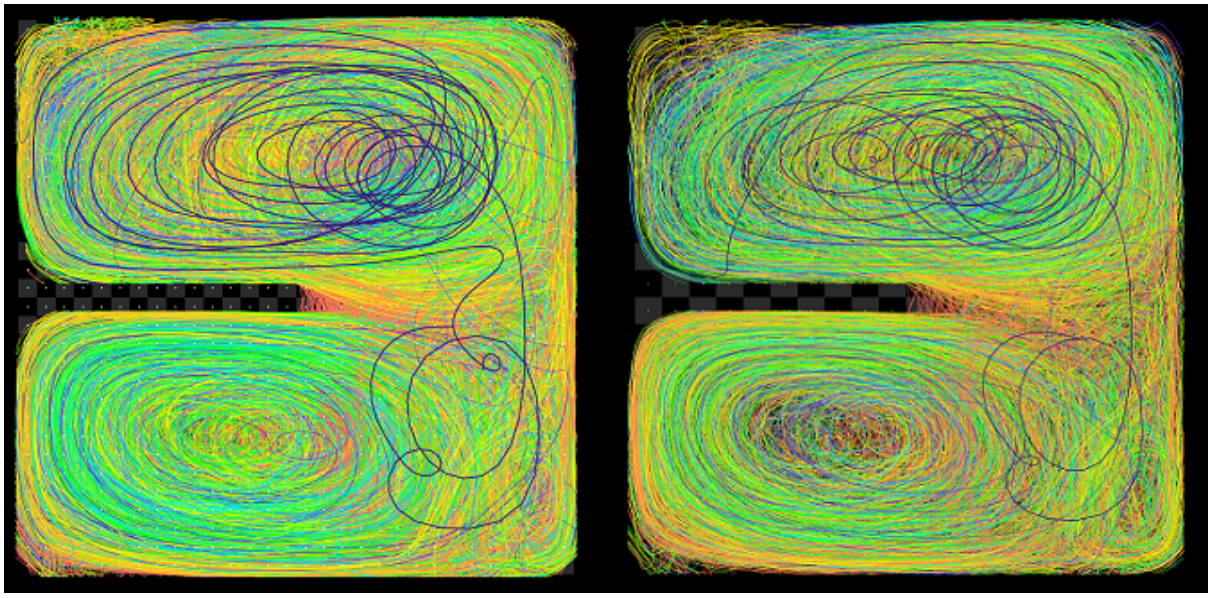

Figure 4.5: Depiction of all pathlines in original size made by 33x33 seeding points(left) and 21x21 seeding points(right). the highlighted pathline with mouse-over has one uniform color and has a line thickness twice as thick as the others. All pathlines drawn on the domain, It is nearly impossible to keep track of one line. one can merely recognize rough curling regions and distinguish between densely or sparsely visited areas. on the 21x21 seeded field, the background is not completely filled with pathlines, therefore the highlighted pathline vaguer than in the left.

implemented in this paper is to downscale each pathline in its seeding point (see Fig. 4.6). since the seeding points are arranged on a regular grid, every grid element is a downscaled representation of the whole vector field. Thus the starting points of each pathline are not in the center of the grid element, but locally displaced to give a exact representation of the entire vector field with only one pathline present. Since pathlines can reach the edge of the domain, neighboring grid elements can have parts of pathlines touching the raster or interfere each other.

Downscaled pathlines show global patterns. Neighboring pathlines are likely to be similar, suggesting that physically near staring points often lead to similar course of particles over the same time range (see Fig.4.7). In Fig.4.7 some grid elements are dense, meaning the pathlines started from these points run over the entire domain. Then there are reddish sparse areas meaning the pathlines are present only for the earlier time range.

4.3 Tangential Glyphs

Flow Radar Glyphs can be applied to vector field properties that change over time. In this section, implementation of Flow Radar Glyphs that visualize tangentials along Pathlines will be discussed. In this implementation, the magnitude of vector does not play a role. A constant factor independent of

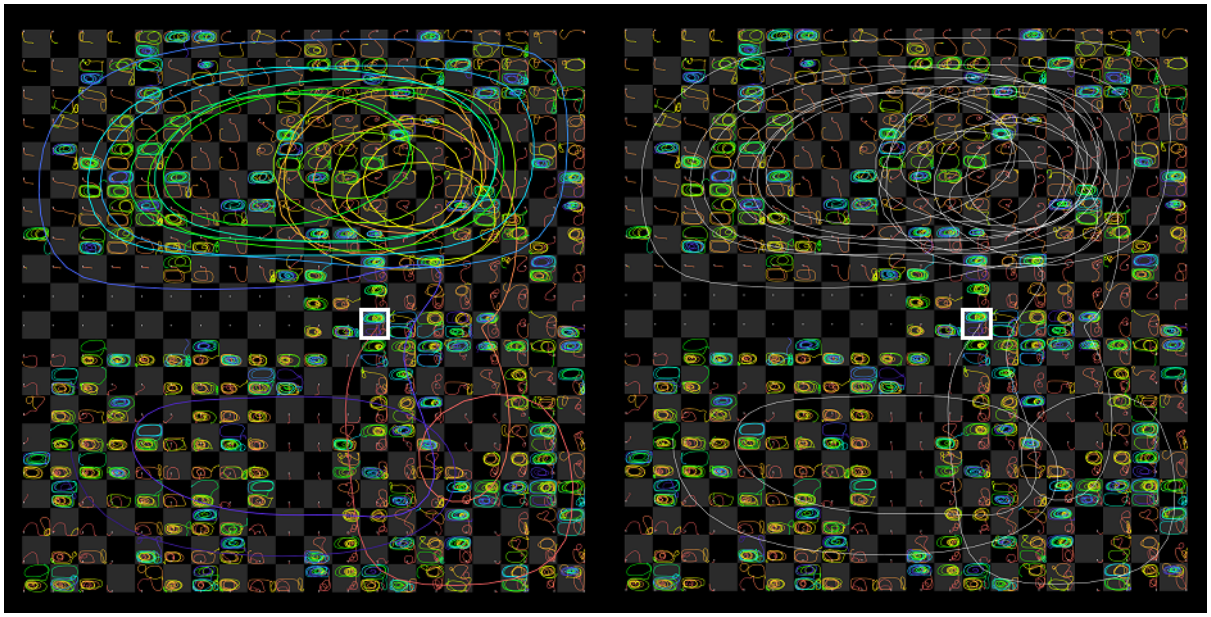

Figure 4.6: Downscaled pathlines inside each local representation of the domain. Pathlines can be displayed in its original size on the global domain, using mouse-over function. The highlighted pathline can be displayed using the same color coding as the downscaled lines(left) or using a color that is not present in the color coding(right).

vector magnitude is added to the radius of Flow Radar Glyphs with increasing time(which means that normalizing glyph size is not needed). If the pathline runs for the entire time range, all tangential glyphs will have the same size. We will see that this is not the case for some data sets. Fig. 4.8 shows how to interpret tangential glyphs. These two tangential glyphs show a comparison of a glyph with a monotonous change of direction to a glyph with a different behavior.

With these properties of tangential glyphs of pathlines, we will examine a global view and a close-up of an area of interest in Fig. 4.9.

4.4 Grids

A last point that deserves mention in implementation is the marking of the grid. Grids may not obtrude or interfere with the actual visualization elements. Instead, they must be helpful keeping the elements separate and distinct. Bartram et al.[BSC07] suggests putting a transparent gray grid whereby the grid may or may not be layered on the visualization. In this implementation, two methods of marking grids are used, which can be applied alone or together at the same time. Fig.4.10 shows checkerboard grids only.

Fig.4.11 shows the second method of marking the grid, which is rasters. In many cases the checker-

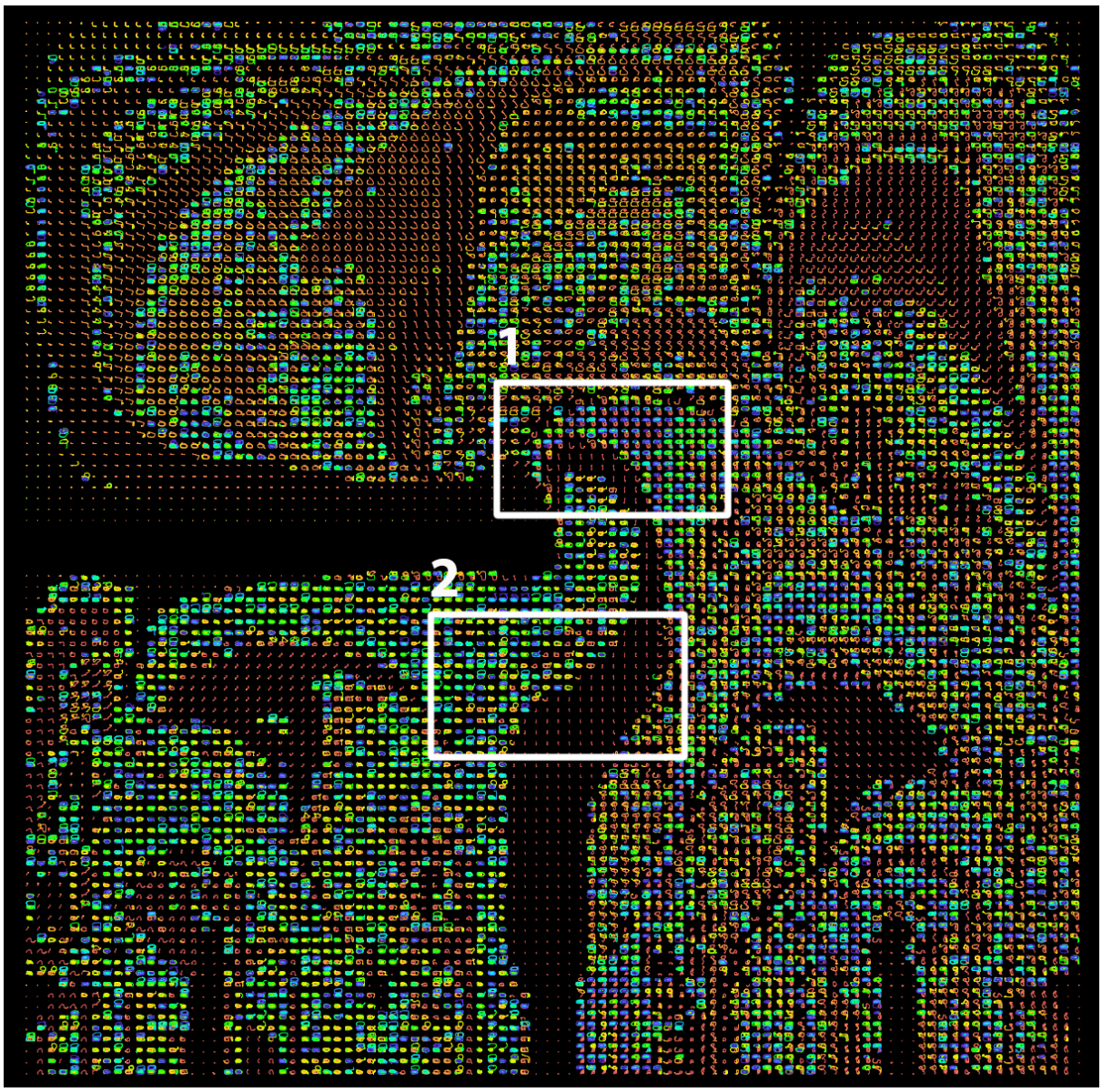

Figure 4.7: Entire domain with 99x99 downscaled pathlines. Areas 1 and 2 are zoomed-in in Fig.5.2 and Fig.5.3.

board grid seems to be better for perception. Especially in zoomed-out view, rasters disturb the recognition of patterns .

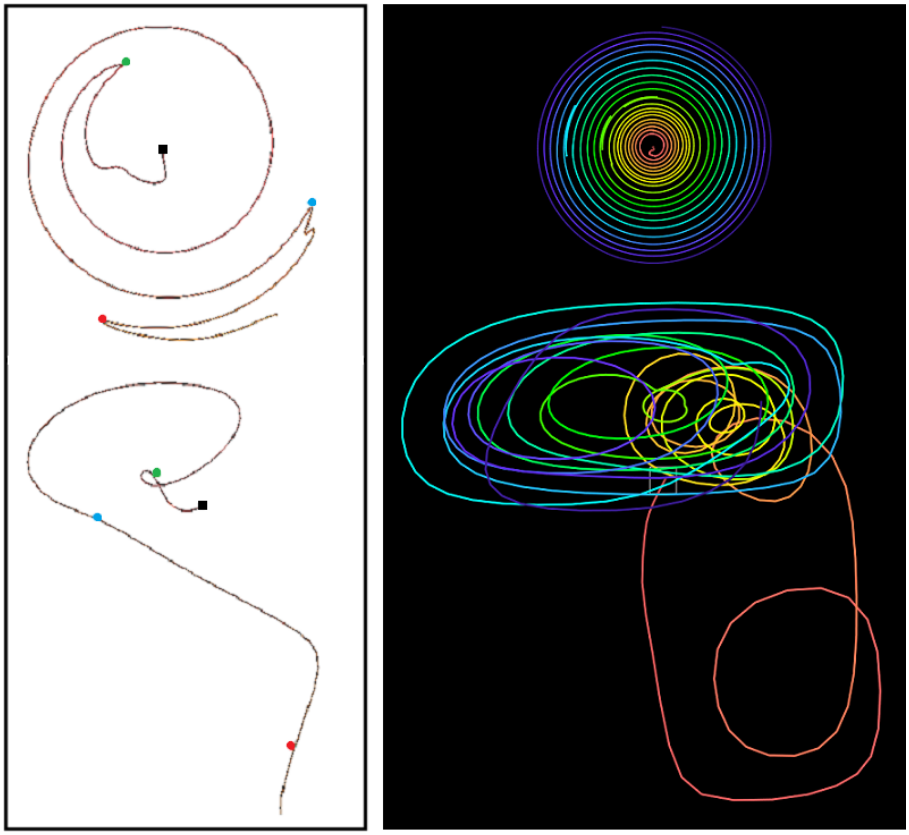

Figure 4.8: Left image shows a pathline and corresponding tangential glyph. respectively at green, blue, red point the sign of the curvature of pathline changes from - to + or + to - and passes 0. The Flow Radar Glyph resulting from tangential vectors of pathline shows sharp folds where this occurs.
Right image shows another pathline and its tangential glyph. In contrast to the pathline in the left, this pathline almost curls only in one direction throughout the path. Therefore, there is no remarkable change in the tangential glyph.

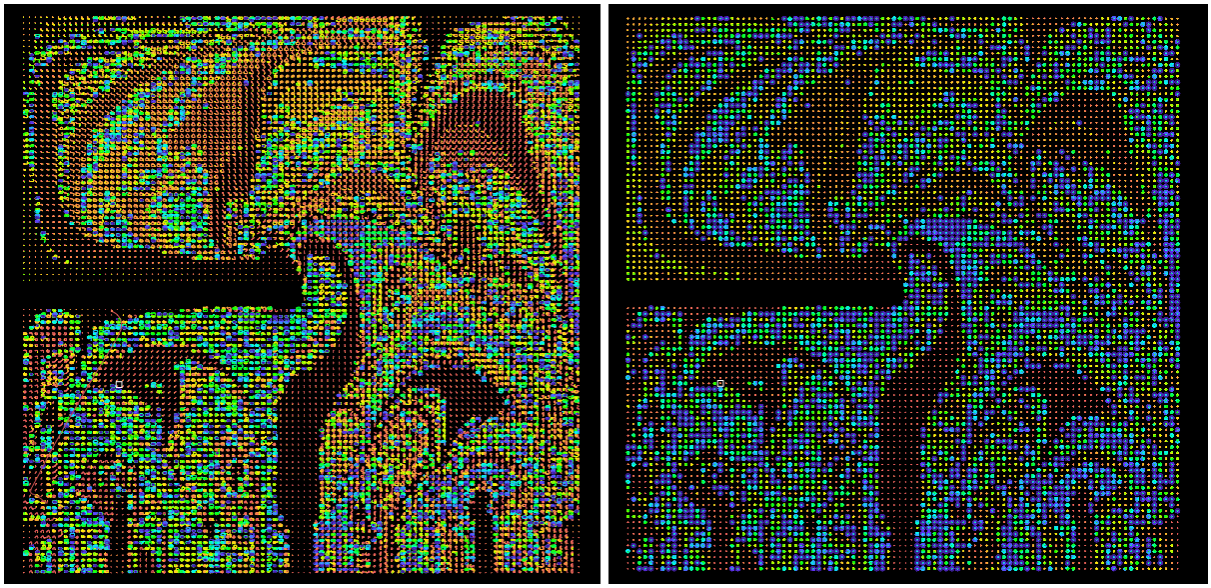

Figure 4.9: Pathlines(left) and tangential glyphs(right) with 99x99 seeding grid compared. In the tangential glyphs, blue color is more dominating, because in a later time range the arcs on the tangential glyphs occupy more pixel.

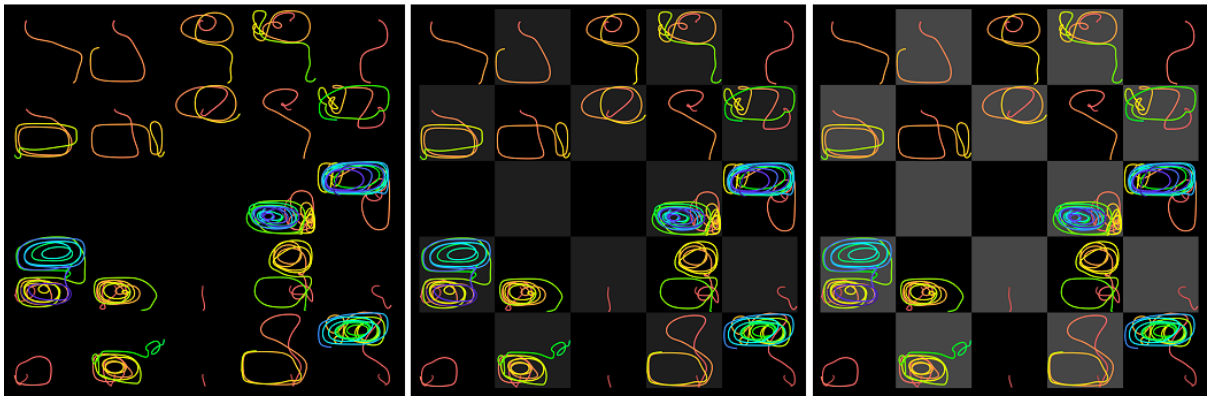

Figure 4.10: Downscaled pathlines in checkerboard raster with varying brightness. from left to right, RGB values are set to 0, 30 and 70 in 0 to 255 range. Left, no orientation with respect to grid is possible. Position of seeding point and position of pathline with respect to domain is hard to read, especially when the entire pathline is only in the lower part or upper part of the domain. In the center image, problems with the left image are mended. Although this gray value might be too weak especially when zoomed out. The right image has a higher gray value, making the pathlines appear vague. In the application and snapshots in this paper, the default value is set to 43, which is between the center and the right image.

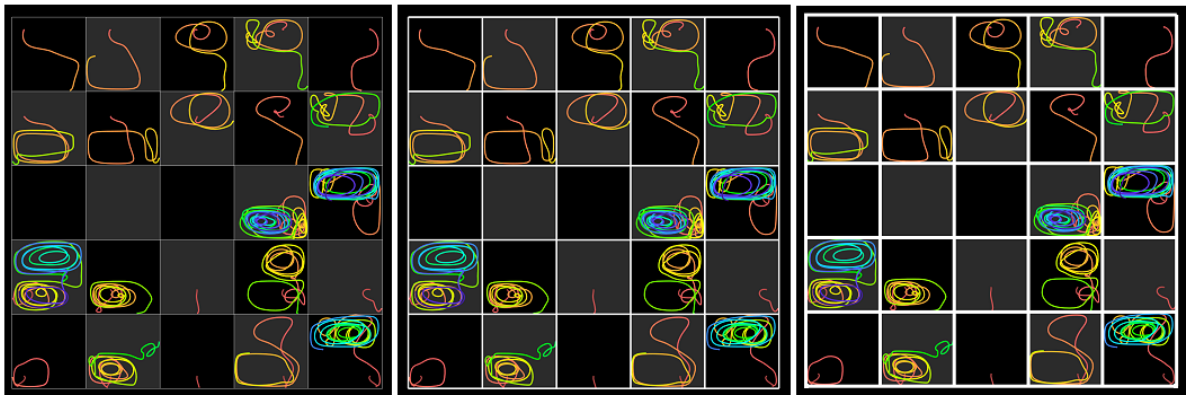

Figure 4.11: Rasters with pixel-width 1, 5 and 10. In the implementation, default value is set to 1. Snapshots in this paper are mostly took without rasters though. Raster at the boarder of downscaled pathlines can be opaque and do not intersect with visualization.

Chapter 5

Results

In this Chapter, zoom-in view of Flow Radar Glyphs are shown (Fig. 5.1 and a second data set with different properties then the first data set will be introduced, to examine what their field lines and tangential glyphs tell about the vector field.

5.1 Flow Radar Glyphs

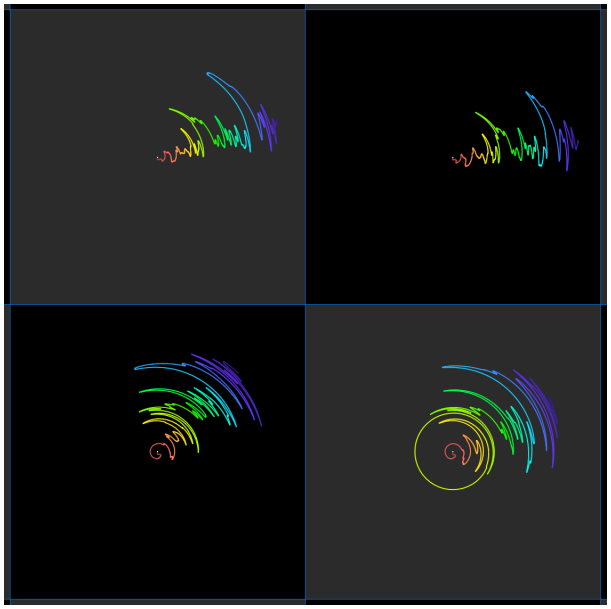

Figure 5.1: 4 adjacent Flow Radar Glyphs. 2 Glyphs on top are similar, which means that the vector field over the whole time range does not change much between these two seeding positions. Bottom 2 Glyphs are distinctly different from each other. Comparing those 2 at the bottom, we can see that having different vectors for just a short time range can make a noticeable difference between two glyphs.

5.2 Pathlines

2 interest areas from Fig.4.7 will be examined here. In the first area(Fig. 5.2), 2 regions with similar pathlines are shown. In region 2, there are pathlines that stop at the edge of the domain, (magnitude of the vector field at the edge of the domain is 0) where the particle cannot change its position and neither come back to the domain. This 'early stop' of pathlines comes from the data and is not an anomaly of visualization.

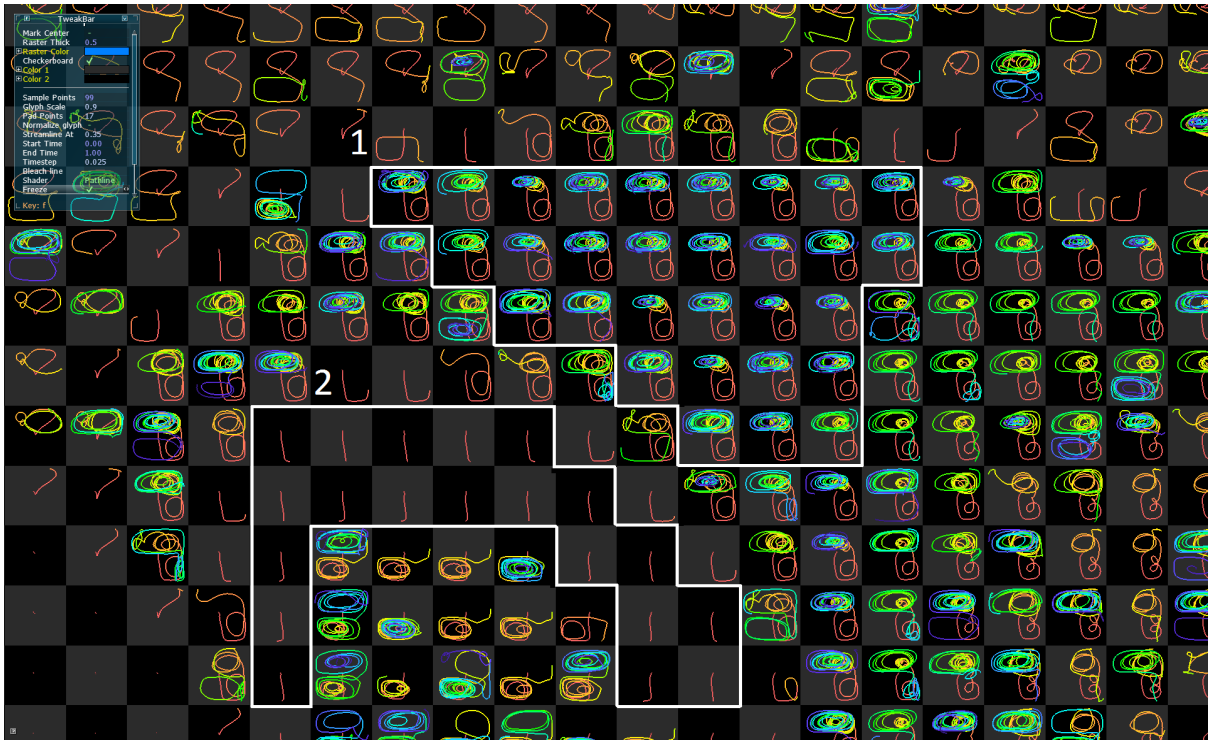

Figure 5.2: zoomed-in view on area 1 from Fig.4.7. Region 1 shows very similar pathlines. The pathlines go in to the lower right part of the domain(red part), curling one or two cycles. After that comes back to the upper half of the domain, and keep curling over the entire remaining time range(yellow, green and blue part of time range). Region 2 has similar pathlines that are short and running only for a short time range, showing a sharp contrast to neighboring pathlines. The pathlines in region 2 meet the edge of the domain. Particle positions read in neighboring regions at the time instant at which the pathlines in region 2 meet the edge of the domain are very similar to region 2. Thus we can deduce that the difference of the vector field on the spot is very subtle at the time instant.

In the second area(Fig. 5.3, in region 1, every neighboring pathlines are different from each other. In region 2, pathlines end up at the bottom of the domain, similar to region 2 in Fig.5.2. A closer look at the global view (Fig. 4.7) shows that they are connected by similar pathlines (note the streak of short red pathlines running from bottom center to the center of domain).

Figure 5.3: zoomed-in view on area 2 from Fig.4.7. Region 1 shows a region where all pathlines are different from each other. Note the big difference between adjacent pathlines. They do have in common that they mostly stay in the lower half, but still some pathlines have a short stay in the upper half and a few of them curl in the upper half until the end of time range(4 pathlines in the top row). Region 2 marks the region of pathlines ending up at the edge of the domain.

5.3 Pathlines and Tangential Flow Radar Glyphs

The pathlines and tangential glyphs for the data set 'buoyancy' are compared in this section. Instead of comparing each pathline to its tangential glyph, the patterns made by the pathlines will be compared to the patterns made by the tangential glyphs. entire domain shown in Fig. 5.4 will be zoomed-in in Fig. 5.5.

In Fig. 5.5 it will be shown that there are some features clearly visible in visualization with pathlines and vague in visualization with tangential glyphs.

5.3.1 Pathlines and Tangential Glyphs within Short Time Range

Pathlines examined so far ran over entire time range. Pathlines can be fit to start and end at desired time. Fig.5.6 shows Pathlines in the upper area of the domain running for 8% of the entire time.

Looking at downscaled pathlines with short time range in fine seedings, we can separate regions

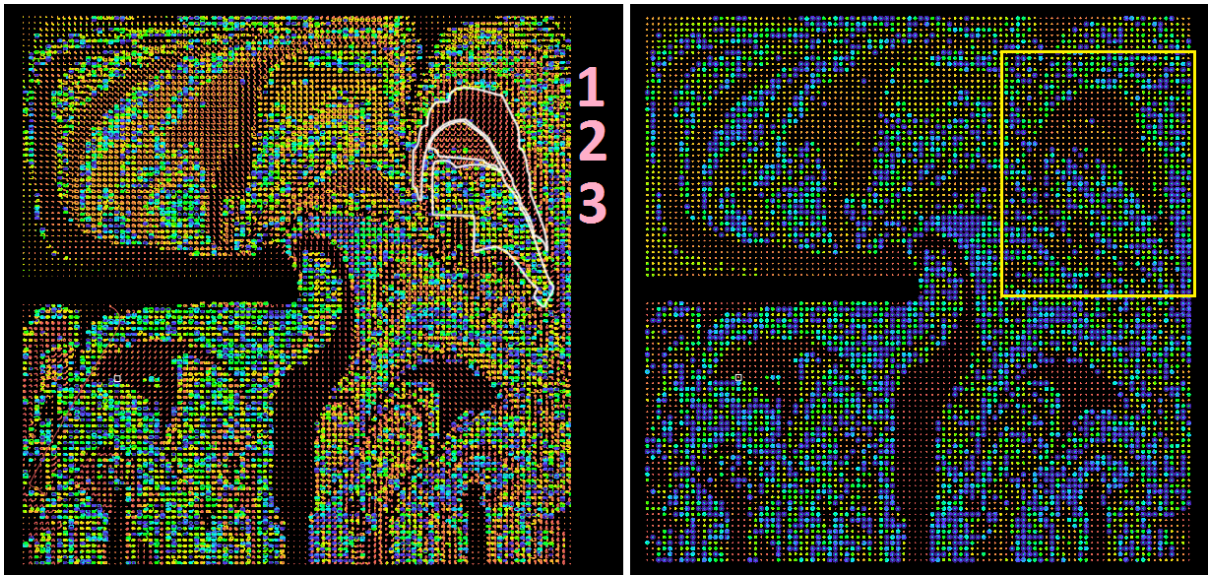

Figure 5.4: On left image, regions with similar pathlines are marked with pink numbers 1, 2 and 3, which will be referred to in Fig. 5.5. Including these regions, yellow area marked in the right image right will be examined in Fig.5.5.

of pathlines with similar course. Navigating through time steps of the pathlines, we can see at what time and where this separation occurs. A separation line between two regions can be detected, also intermediate pathlines between 2 separated regions can be seen. Both cases are shown in 5.7.

Setting the seeding points closer, we can observe one region changing into another. The vector field is interpolated continuously, spotting transitions is not obvious in many cases though. Note that the vector at the transition area is very slow, which can be interpreted from region 2, 3 and 4 in Fig. 5.7.

Tangential glyphs to these pathlines are shown in Fig. 5.8. The division into regions with similar pathlines cannot be made with tangential glyphs. There are very different tangential glyphs for similar pathlines and on the other side, similar tangential glyphs for distinctly different pathlines. Differences between pathlines and tangential glyphs for a short time range can be displayed emphasized or exaggerated.

Transition zones of pathlines in Fig.5.7 are marked in Fig.5.9. As explained above, some transitions are very straightly noticeable and change abruptly, but some others are ambiguous or change gradually with multiple-step transition areas. A denser seeding can make a transition between transitions visible and reducing seeding points can reduce the number of transitions.

Next image (Fig. 5.10, a part of Fig. 5.9) shows at which spot the pathline is moved into which direction.

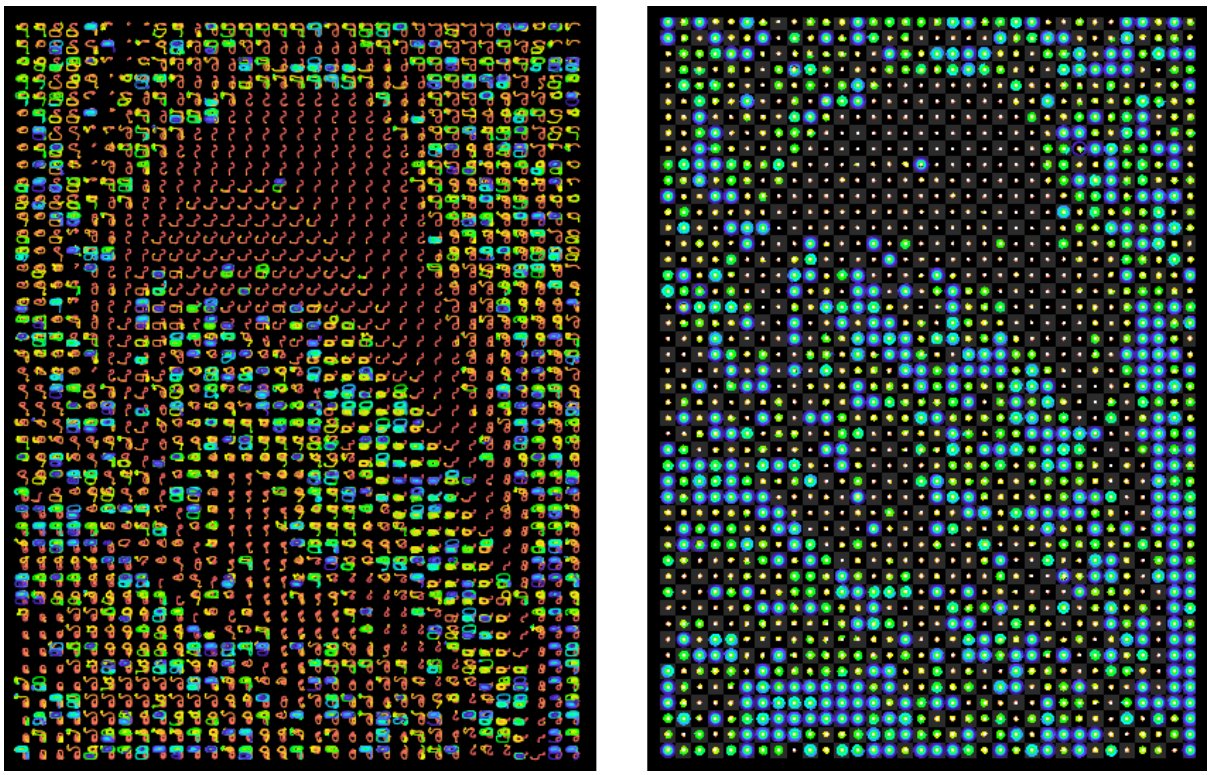

Figure 5.5: Area of interest in Fig.5.4. Left image shows the pathlines, right image shows tangential glyphs. On left, area 1 of Fig.5.4 and and area 2 of Fig.5.4 are clearly distinguishable. This visual separation is not visible in tangential glyphs. On right, the boundary of area 3 of Fig.5.4 is not so clear like on left image. In this view, visual information is rather getting lost with tangential glyphs.

5.4 Another Data Set -'velo'

As shown in section 5.2, there are pathlines ending early at the edge of domain, making the tangential glyphs small. For this Reason the technique with tangential glyphs may not be properly evaluated with the first data set. For a better evaluation, representation of a data set for which the tangential glyphs do not leave the domain is needed. In this section, a second data set is introduced. Tangential glyphs of the second data set, 'velo' has the desired property. Moreover, vectors are normalized in this data set, which means same magnitude of velocity. In the first data set, the vectors were not normalized, but the tangential glyphs were normalized, so the magnitude of the velocity vector had no influence on the tangential glyph. The data is sampled on a 161x161 grid, each 1.25cm apart. It is measured 241 times for different time instants with 0.125 seconds interval.

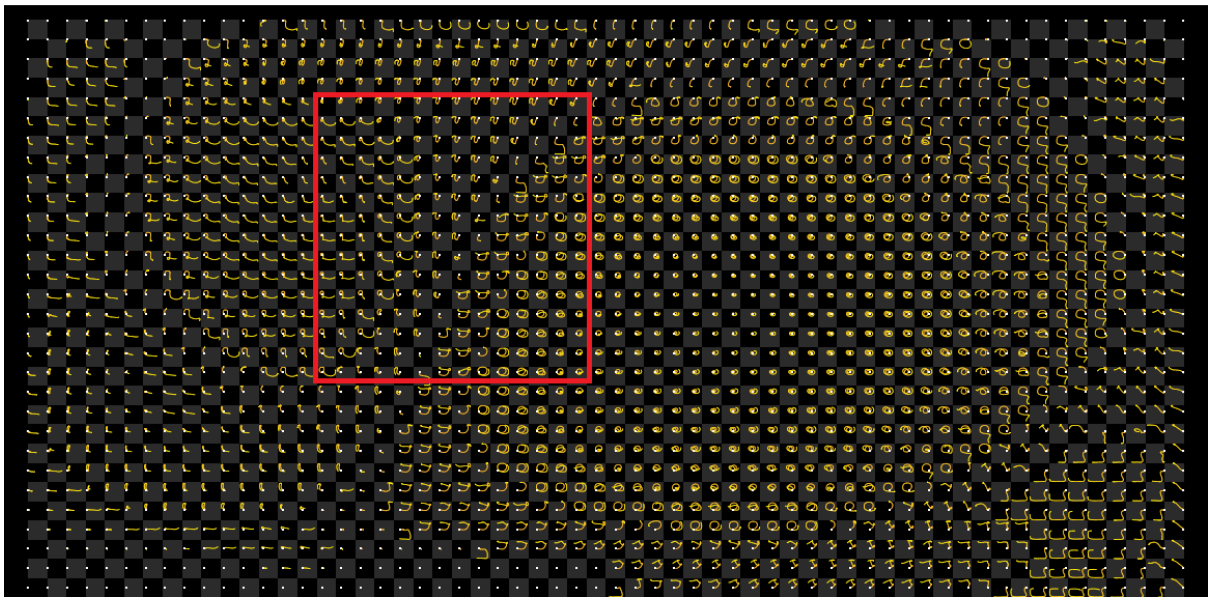

Figure 5.6: Pathlines from time 0.2 to 0.28 (entire data has time range 0.0 to 1.0). Lines are shorter, color coding is varying only a little. This image is showing the same upper area of the domain. Area of interest which will be examined in a close-up view in Fig. 5.7 and Fig. 5.8(tangential glyphs) is marked red.

5.4.1 Pathlines

Fig.5.11 shows 3 representative pathlines over full time range. At first Sight, the domain can be subdivided into 4 regions. In Fig. 5.11, only one subregion is examined, while the behavior of the other 3 subregions are very similar to this subregion.

Summing up the 3 images in Fig.5.11, the curling region seems to move horizontally. Streaklines(Fig.5.12) show that the structure of the field is divided into 4 subregions.

5.4.2 Tangential Flow Radar Glyphs

Before depicting the entire domain with tangential glyphs, tangential glyphs for single pathlines in Fig.5.11 center image and Fig.5.11 right image will be compared in Fig.5.13.

Left image and right image of Fig. 5.13 shows two kinds of different tangential glyphs in this data set. One kind is those that do not change the curling direction, the other kind is those that do. On a global view (see Fig.5.14), these two kinds of tangential glyphs can not be fully distinguished, but the dense region in the center of each 4 curls is visible. In a close-up(see Fig 5.15), for every glyph it is determinable to which kind it belongs.

tangential glyphs can also be depicted only for an arbitrary time range. In Fig. 5.16 tangential glyphs

Number of Vertices	Computation Time
Flow Radar Glyphs, 53484057	0.05s
Flow Radar Glyphs, 9173217	0.25s
Pathlines, 6292242	0.08s
Tangential Glyphs, 106968114	1.295s

Table 5.1: Time that compute shader needs for computing the vertices.

Number of Vertices	Anti-aliasing On	Anti-aliasing Off
Flow Radar Glyphs, 53484057	20.557	10.067
Flow Radar Glyphs, 9173217	75	29.17
Pathlines, 6292242	75	75
Tangential Glyphs, 106968114	20.557	6.67

Table 5.2: OpenGL rendering speed in frames per second.

for a tenth of the entire time range is depicted. For dense glyphs like in this data set, short time range helps recognize features like speed of curl better.

5.5 Performance

Tests to measure duration of compute shader were run on a Nvidia Titan 6GB GPU (see Tab.5.1). Two measures for Flow Radar Glyphs are taken with different numbers of seeding points(41x41 and 99x99, pathlines and tangential glyphs are measured with 99x99 seeding points). The numbers of vertices for pathlines depend on the size of time step used in the integration. Tangetial glyphs more vertices than pathlines, since refining points are needed for making arcs out of straight lines. Same number of refining points per an original point as in the Flow Radar Glyphs are used here. Computing time is roughly proportional to number of vertices, though not exactly. This depends on different computation methods for different glyphs and field lines.

Shader computation is only initiated when necessary(for example when the size of the buffer changes and therefore vertices have to be computed). On the other hand while the user navigates through the 2D-plane, OpenGL renders a new picture without invoking compute shader. This rendering time is measured in frames per second in Tab.5.2. It shows that antialiasing in OpenGL requires a longer rendering time, which means less frames per second.

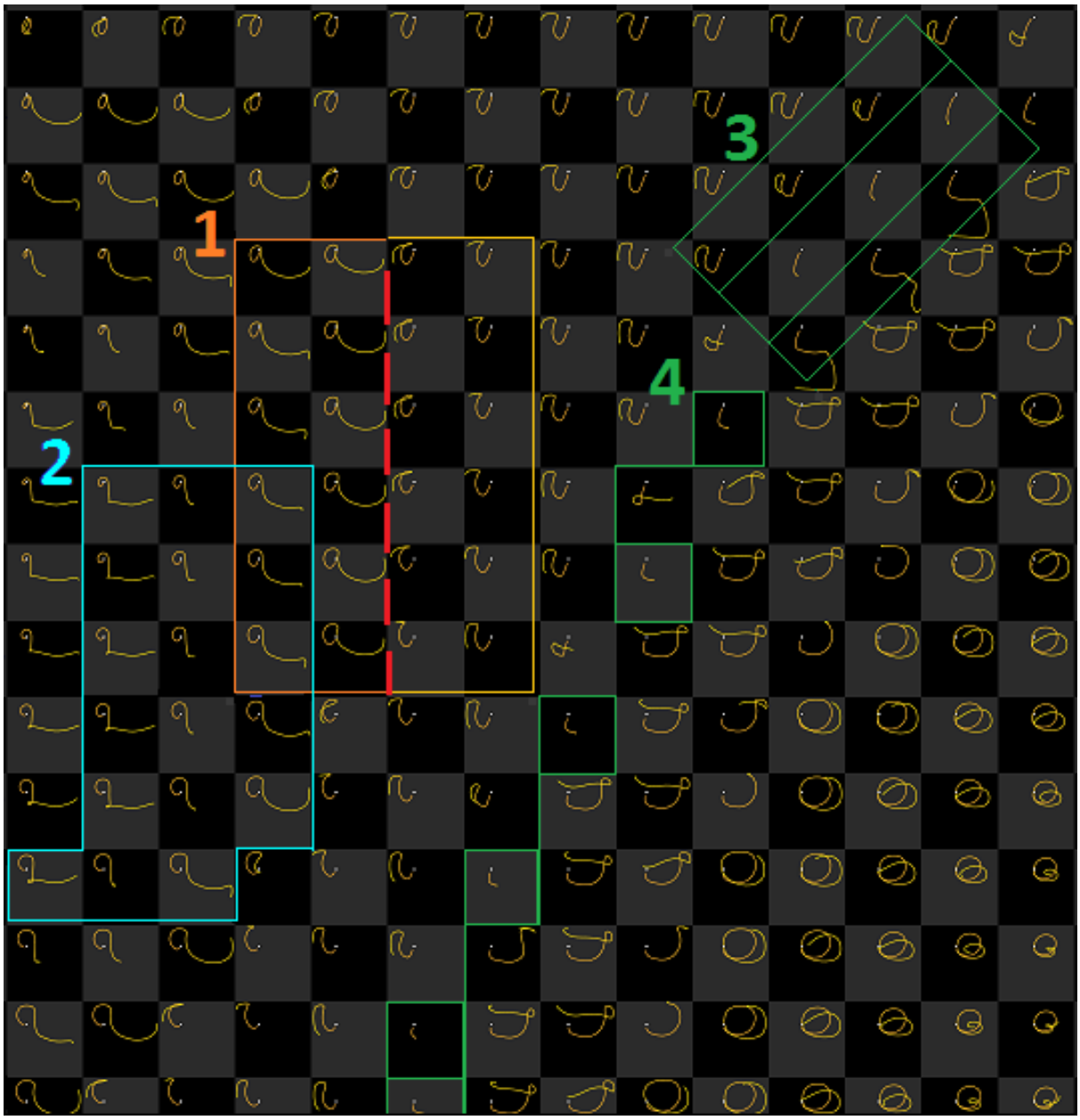

Figure 5.7: Pathlines of marked region in Fig. 5.6 zoomed in. In region 1, there is a clear separation into two subregions. there is an abrupt direction change in the middle of time range. Red dotted line indicates this separation. With a finer sampling, pathlines may appear which show the transition. In the orange part, particles are engaged in an area with fast flow after separation, whereby in the yellow, the vectors are much slower. In region 2, the pathlines in the left column seem to be repeating itself in the right column. This pattern is displaced one step in the bottom row. Center column shows a little transition. In the left column there is a little fold, abruptly changing direction. Direction changes are mostly slow parts of pathlines, which may be related to transition in the center column. while left column has a fold, right column has a smooth direction change of pathlines. Region 3 shows a diagonal separation strip. upper left 3 particles go in the left direction and the lower right ones to the right. Single pathlines in squares (number 4) are also transition areas dividing left and right. They rather belong to the right as they start moving to the right. Straight green lines between pathlines suggest separations without intermediate pathlines.

Figure 5.8: Tangential glyphs of Pathlines in Fig.5.7. Same seeding areas are marked. Left of region 1 is visually subdivided into two groups of glyphs(into first column of glyphs and second column of glyphs), caused by the difference in pathlines in the latter part of time range(See Fig.5.7, the last part of pathlines are in the first column headed upwards and headed downwards in the second column.) In area 3, the most top-right glyph is notably different from 2 others in the same diagonal column. In area 4, the lowermost glyph is different from others. This difference is hardly noticeable in pathlines(in Fig. 5.7). Furthermore, glyph A and B is similar, while in Fig. 5.7, these two pathlines are distinctly different.

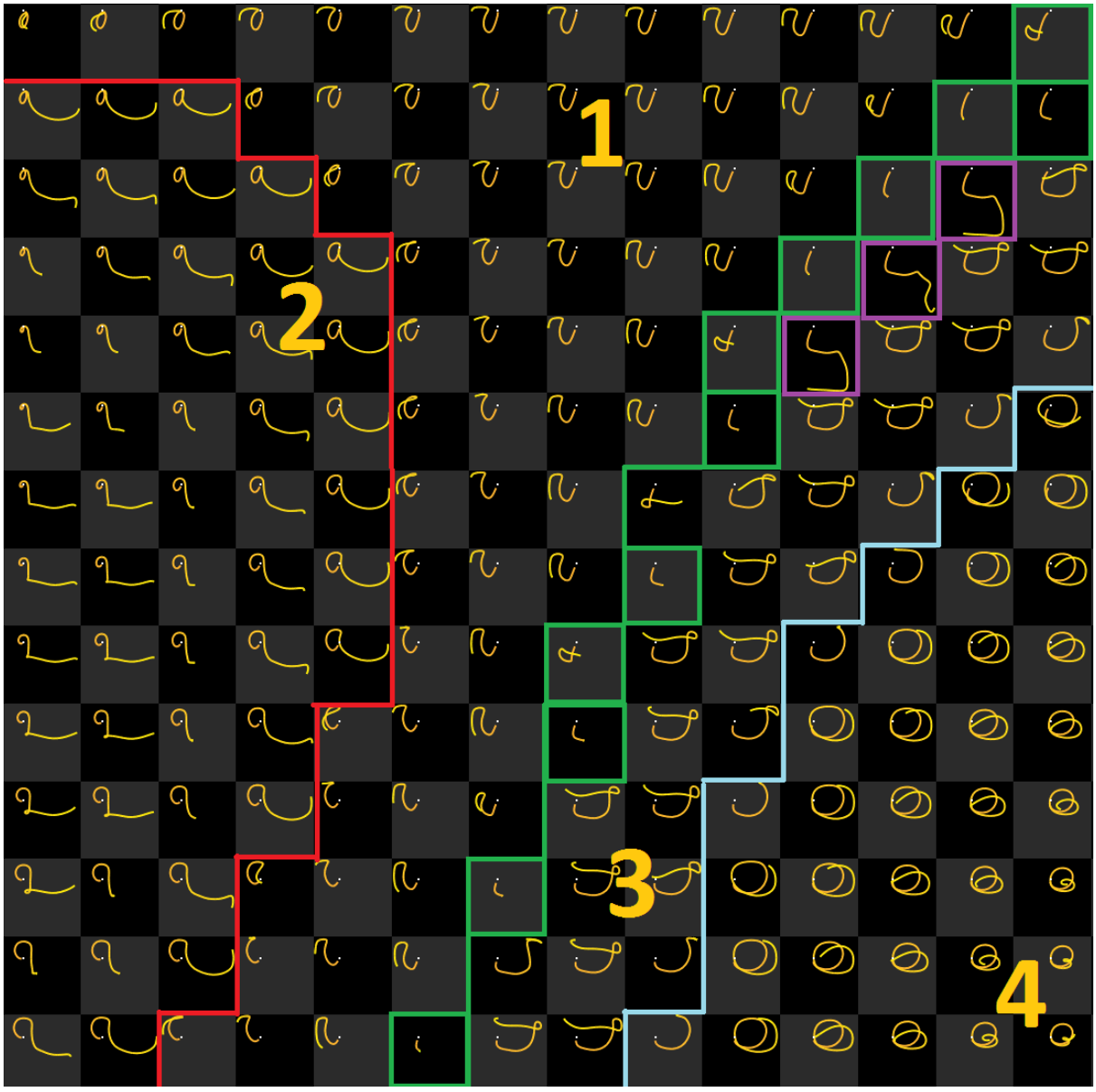

Figure 5.9: Transition zones of pathlines in Fig.5.7. Transition between 1 and 2 is sharp and narrow, while the transition engaging areas 1, 3 and 4 is broader and more gradual. Notice the outliers like the purple pathlines are different from all other neighboring pathlines.

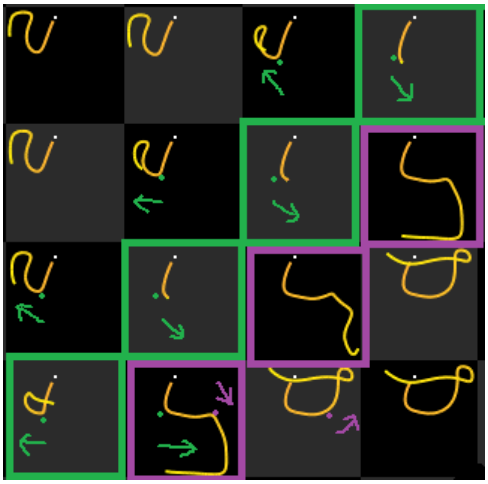

Figure 5.10: Green and purple points show at which position of domain the separation of vector occurs (two neighboring green or purple points point the same spot on the domain). Green points show vectors splitting roughly to the right and left direction. Purple points show vectors splitting downwards and upwards. Pathlines compared pairwise are at green and purple spots at the same instant of time.

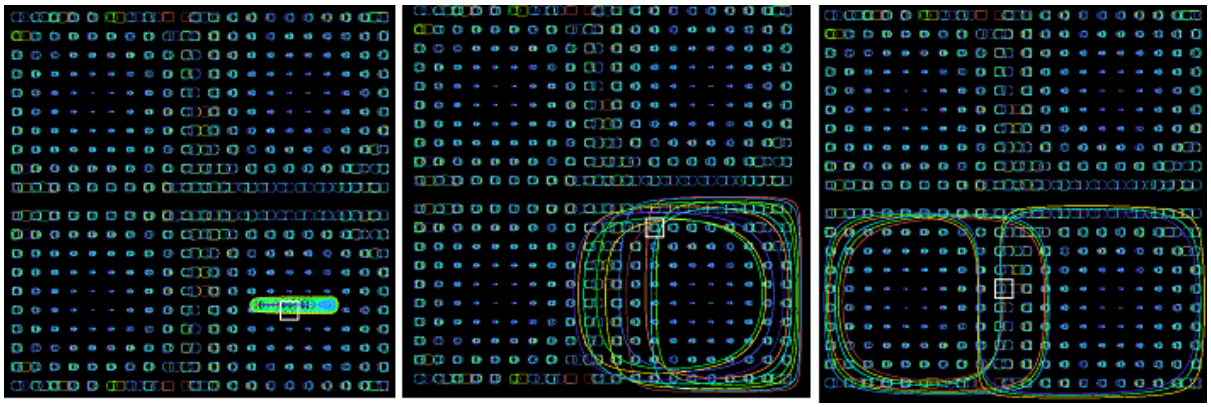

Figure 5.11: 3 representative Pathlines of lower right region. In the left image, the particle at the center do not leave the center area where it started, but the center of the region moves horizontally. In the center image, This pathline curls in its own roughly rectangular subregion. In the right image, pathline of a particle started at the boundary of two regions is shown. This particle goes over to the neighboring curling region many times. It partly alternates between the two neighboring regions, drawing lying 8's. This shows that the separation of regions move horizontally.

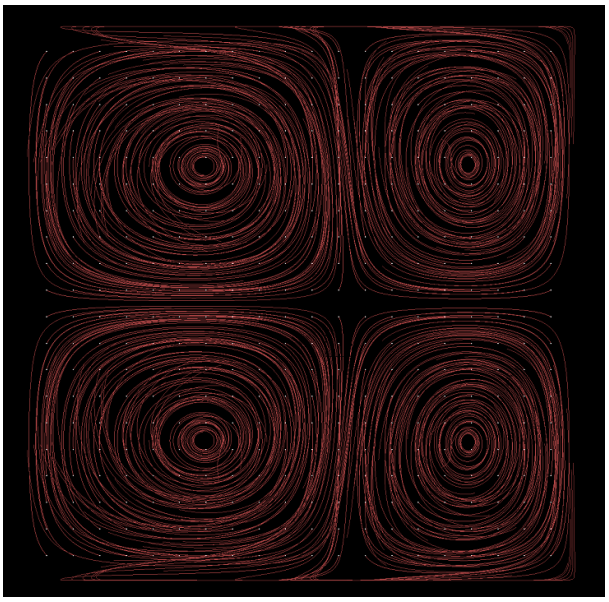

Figure 5.12: Streakline of data set 'velo', with a short time range starting from 0. Streaklines show that this vector field is made up of 4 similar curling regions.

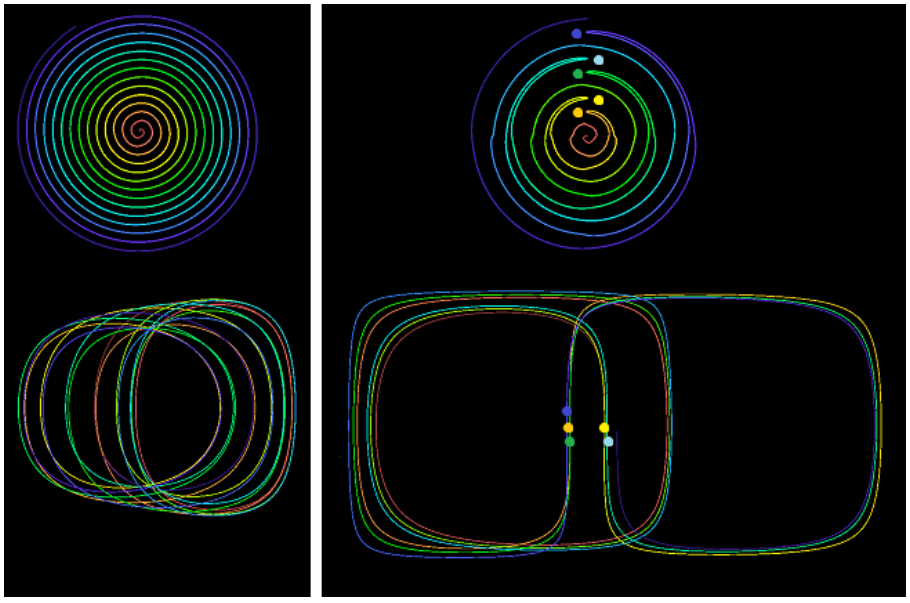

Figure 5.13: Left image shows the pathline of center image of Fig.5.11 and its tangential glyph. Curling center of circle moves, this information is neglected in the glyph. Any pathline swirling in one direction of curve results in a glyph very similar to this.
Right image shows pathline of right image of Fig.5.11 and its tangential glyph. Points on the glyph on which the pathline changes the curling direction are marked in its color coding. Change of direction is depicted very distinctly in its tangential glyph. Number of changes can be count with tangential glyph. This pathline changes its curling direction only when heading straight upwards.

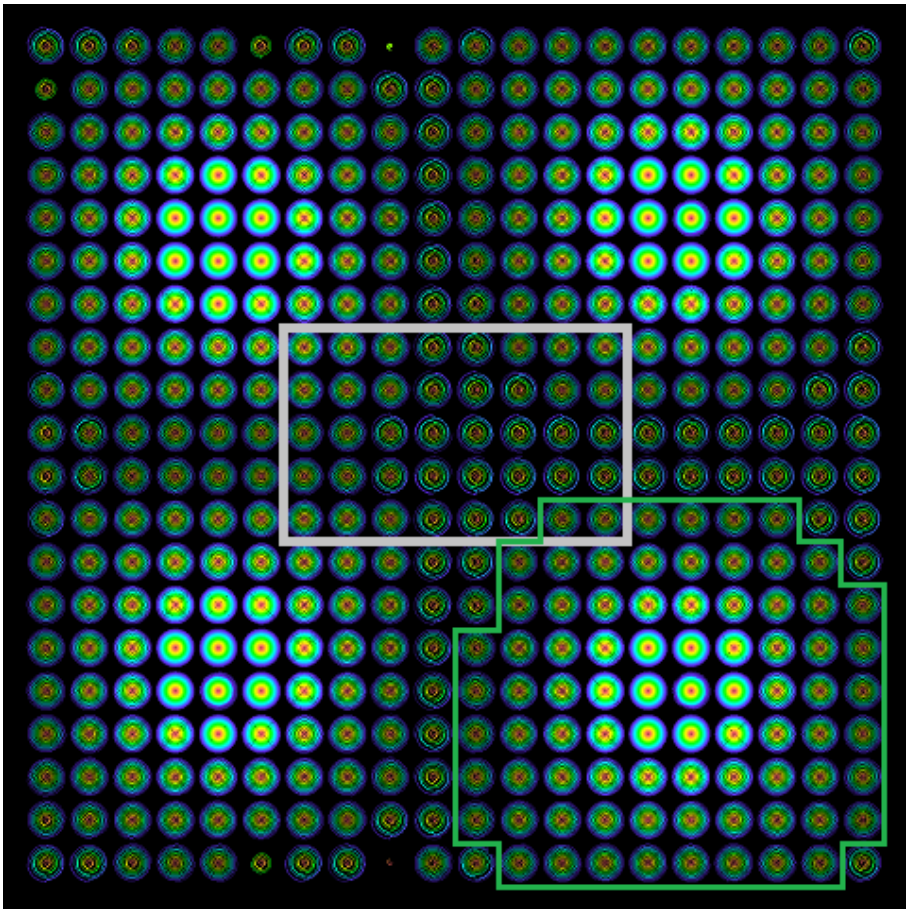

Figure 5.14: Tangential glyphs of data set 'velo'. Note the central regions of curls are depicted very densely, oppositely to the same regions of pathline representation in the left of Fig.5.11), which are sparsely filled. Gray area will be zoomed-in in Fig.5.15. In the region with green boundary are glyphs of pathlines which do not have a change in curling direction and look like the glyph in the left image of Fig.5.13. The domain has 4 such regions, since the domain is subdivided into 4 curling regions.

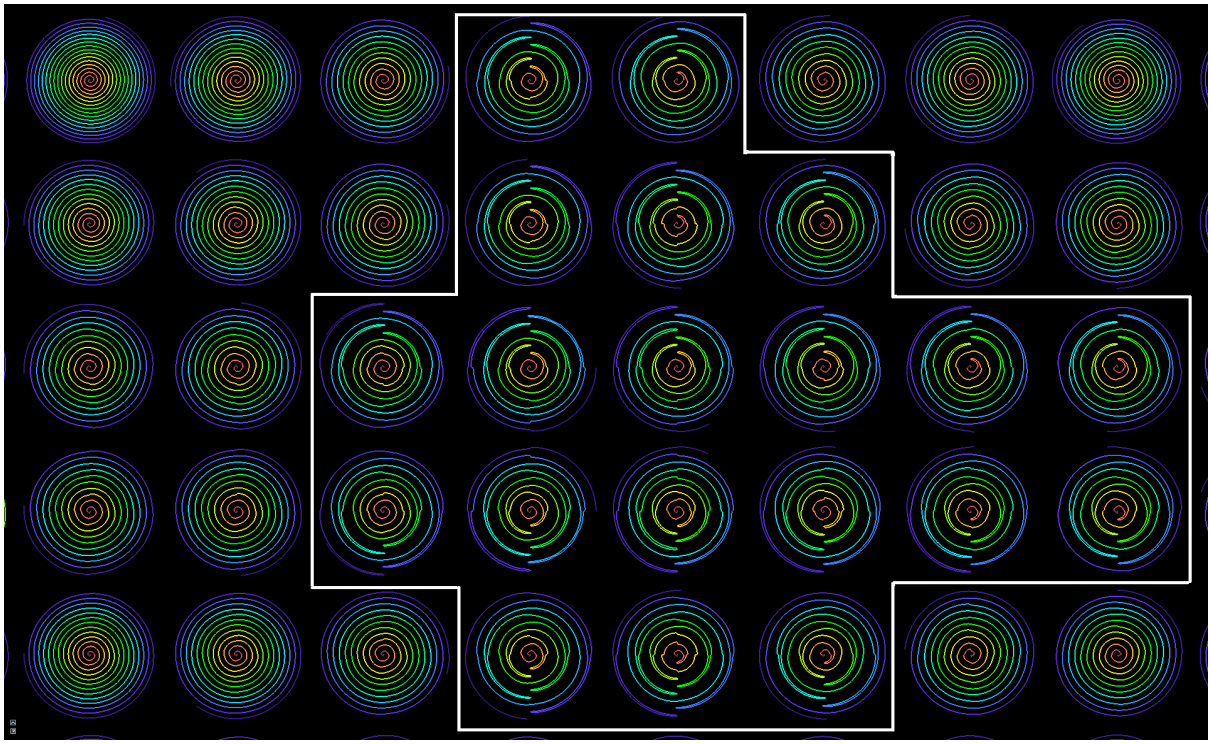

Figure 5.15: Close-up of gray area in Fig.5.14. Again in gray area are pathlines with change in curling direction. Above the red line, the change occurs when particles are headed upwards, below the red line when they are headed downwards.

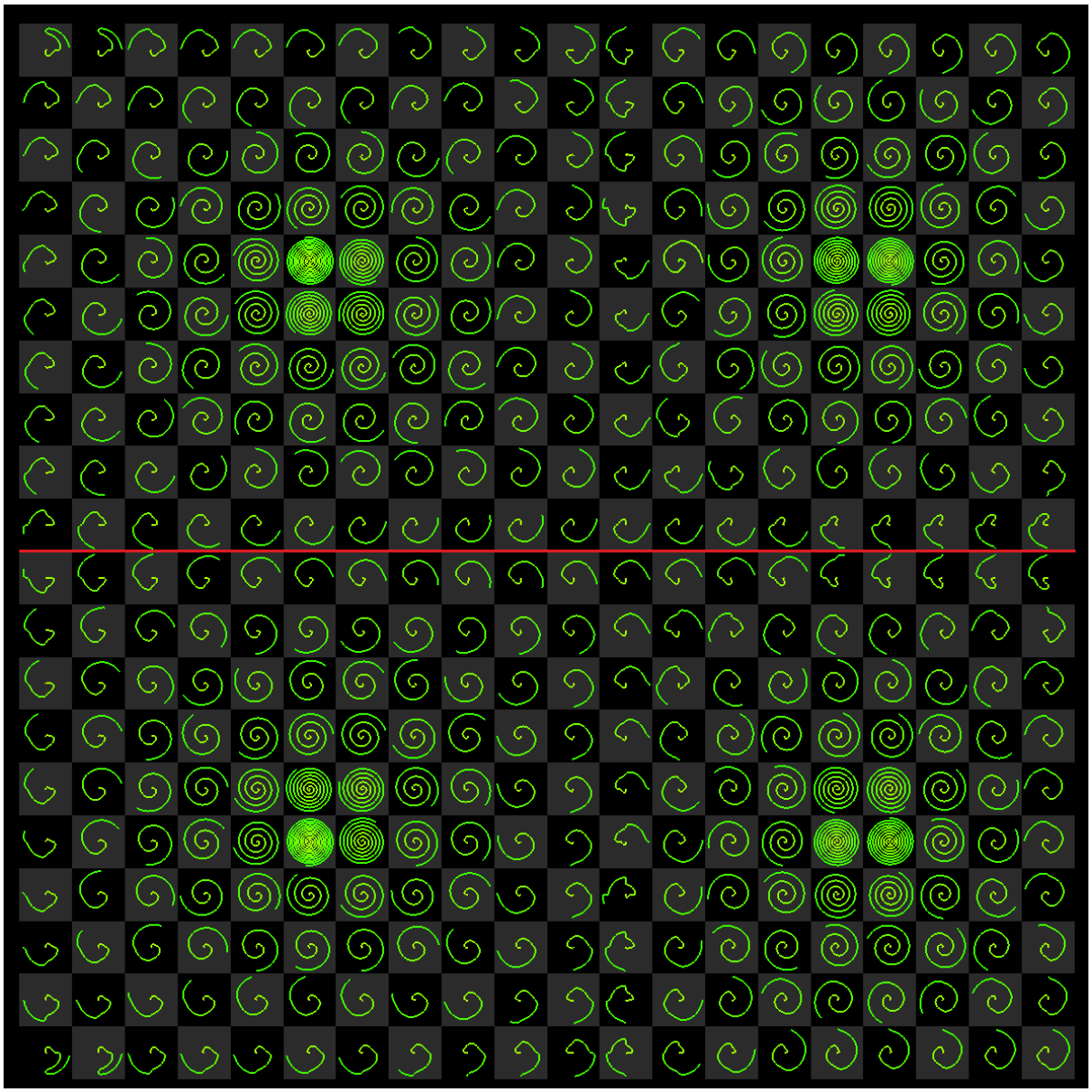

Figure 5.16: Tangential glyphs for a tenth of the entire time range. The vector field is extremely curling, so the tangential glyphs get very dense soon. A short time range can be better at showing the actual directions of the pathlines. The highlight of the center of curl gets even clearer compared to Fig.5.14. Tangential glyphs adjacent to the red line are mirrored to each other. This shows that the upper half and the lower half is identical over time. This representation shows clearly how many times the pathlines curl and the differences in curling speed.

Chapter 6

Conclusion

Through downscaling pathlines, we can view many neighboring pathlines simultaneously and compare them. Through this possiblity, separation of vector field at a time instant into different directions could be detected. By means of these separations, transition zones could be defined.

Visualizing tangents along pathlines as Flow Radar glyphs turned out to be not the most effective visualization for pathlines. Tangential glyphs are very sensible to curls in vector field and any direction changes in the pathline. Curls make the glyph dense very fast(as seen with the second data set), zooming out just a bit makes the glyph hardly suitable for interpretation. This hampers viewing many glyphs at once, thereby making it hard to compare multiple glyphs or glyphs of a region. Susceptibility to direction changes makes it hard to distinguish between meaningful lasting direction change and short jerky movements or shiverings. Therefore, Insignificant change in direction of points on pathline is exaggerated. For the data sets examined in this paper tangential glyphs tend to be similar and it is hard to characterize each glyph. Dominant curls in both data sets might be responsible for this similarity. Moreover, the amplitude of the vector is neglected in tangential glyphs. All in all, tangential glyphs were not an intuitive tool for interpreting pathlines in this study.

Of the grid marking methods introduced in 4.4, gray checkerboard patterns seemed to be the more effective. It does not collide with start point markings, also did not disrupt visualization when zoomed out.

Chapter 7

Future Work

7.1 Clustering

Neighboring Flow Radar Glyphs are often similar. Clustering similar glyphs or can enhance recognition of patterns in the flow. The clustered glyph could be represented with a correspondingly bigger size according to the size of the cluster, and its position is to be set with an adequate function. This makes it possible to stay in the global view and still be able to examine the glyphs.

For a clustering criteria, similarity functions for glyphs must be defined. One simple intuitive similarity function is to sum up the differences in vector angles. Telea[TVW99] defines an elliptic similarity function using hierarchical clustering. In hierarchical clustering, the pair of glyphs with the highest similarity is chosen to unite in every step. The clustering can be continued until a single Glyph represents the entire domain. Multi-pass invocation of shaders can be implemented here. Optionally, the clustering stops if the similarity of the most similar pair of cluster is less than a predefined value.

Colors can be granted to clusters for a better visual separation of cluster. Furthermore, the seeding points in the cluster can be given a color gradation according to the similarity to the representative glyph of the cluster. In Telea [TVW99], each cluster has its color, but a gradation inside a cluster is not implemented.

Pathlines could be clustered, too. A simple similarity function between two glyphs could be summing up the distance of two points on each pathline in every time step.

7.2 Field Lines

For a more exact computing of field lines, adaptive step size can be used in integration. Timesteps could be lessened according to the difference between current vector and the next one.

Like pathlines, streaklines could also be minimized at its seeding point.

On a zoomed-out view, glyphs and lines do not need to be as precise as when zoomed-in. Application could support an interactive change of field line density while zooming in and out. Adaptivity with zooming does not necessarily enhance performance when invoking compute shader often. In this case, simply reducing the number of points drawn when zoomed-out could help.

Bibliography

[BKC+12] BORGO, Rita; KEHRER, Johannes; CHUNG, David H.; MAGUIRE, Eamonn; LARAMEE, Robert S.; HAUSER, Helwig; WARD, Matthew ; CHEN, Min: Glyph-based Visualization: Foundations, Design Guidelines, Techniques and Applications. In: *Eurographics 2013-State of the Art Reports* The Eurographics Association, 2012, S. 39–63

[BSC07] BARTRAM, Lyn; STONE, Maureen ; CONSULTING, StoneSoup: Whisper, don't scream: Characterizing subtle grids. In: *IEEE Visualization 2007, Sacramento CA* (2007)

[Hee] HEER, Jeffrey: *A Tour Through the Visualization Zoo*. `http://hci.stanford.edu/jheer/files/zoo/`. – [Online; accessed 13-June-2013]

[HLNW11] HLAWATSCH, Marcel; LEUBE, Philipp; NOWAK, Wolfgang ; WEISKOPF, Daniel: Flow radar glyphs-static visualization of unsteady flow with uncertainty. In: *Visualization and Computer Graphics, IEEE Transactions on* 17 (2011), Nr. 12, S. 1949–1958

[LW93] LEEUW, Willem C.; WIJK, Jarke J.: A probe for local flow field visualization. In: *Visualization, 1993. Visualization'93, Proceedings., IEEE Conference on* IEEE, 1993, S. 39–45

[Mac04] MACEACHREN, Alan M.: *How maps work: representation, visualization, and design*. The Guilford Press, 2004

[MLP+10] MCLOUGHLIN, Tony; LARAMEE, Robert S.; PEIKERT, Ronald; POST, Frits H. ; CHEN, Min: Over Two Decades of Integration-Based, Geometric Flow Visualization. In: *Computer Graphics Forum* Bd. 29 Wiley Online Library, 2010, S. 1807–1829

[Tuf06] TUFTE, Edward R.: *Beautiful Evidence*. Graphics Press Cheshire, CT, 2006

[TVW99] TELEA, Alexandru; VAN WIJK, Jarke J.: Simplified representation of vector fields. In: *Visualization'99. Proceedings* IEEE, 1999, S. 35–507

[Wer13] WERNER, E: *Vector Field Plots in 2D for Mathcad 15 and Prime 2*. `http://communities.ptc.com/docs/DOC-3503`. Version: 2013. – [Online; accessed 04-July-2013]

[Wik13] WIKIPEDIA: *Runge-Kutta methods — Wikipedia, The Free Encyclopedia*. `http://en.wikipedia.org/wiki/Runge-kutta`. Version: 2013. – [Online; accessed 18-June-2013]

[Zil13] ZILKEN, Herwig: *Visualisierung von Vektorfeldern*. http://www.fz-juelich.de/SharedDocs/Personen/IAS/JSC/EN/staff/zilken_h.html. Version: 2013. – [Online; accessed 04-July-2013]

List of Figures

Erklärung

Hiermit versichere ich, diese Arbeit selbständig verfasst und nur die angegebenen Quellen benutzt zu haben.

_____

(Hajun Jang)