#### ANALYSIS SUPPORTED SPH FLUID SIMULATION

A Thesis

Presented to the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment of the Requirements for the Degree Master of Science

By

Wei Cao

December 2013

#### ANALYSIS SUPPORTED SPH FLUID SIMULATION

Wei Cao

APPROVED:

Dr. Guoning Chen, Chairman Dept. of Computer Science

Dr. Zhigang Deng Dept. of Computer Science

Dr. Jingmei Qiu Dept. of Mathematics

Dean, College of Natural Sciences and Mathematics

#### ANALYSIS SUPPORTED SPH FLUID SIMULATION

An Abstract of a Thesis Presented to the Faculty of the Department of Computer Science University of Houston

> In Partial Fulfillment of the Requirements for the Degree Master of Science

> > By

Wei Cao

December 2013

## Abstract

Smoothed Particle Hydrodynamics (SPH) is a mesh-free method that has been widely used in several fields such as astrophysics, solids mechanics, and fluid dynamics. This computational fluid dynamics model has been extensively studied and is mature enough to enable detailed quantitative comparisons with laboratory experiments. Therefore, understanding and revealing the underneath behaviors of SPH fluid simulation becomes more meaningful when SPH is used to help us understand similar phenomena in the real world.

In the thesis, we use the Finite Time Lyapunov Exponent (FTLE) and a novel rotation metric as well as other analysis methods to analyze the SPH. First of all, we modify traditional FTLE by using Moving Least Squares to calculate the deformation matrix, and extend the usage from mesh-based to mesh-free data sets; we are the first to apply FTLE on free surface SPH fluid simulation. In addition, we are the first to apply rotation sum and gradient of rotation sum on particles based fluid simulation. We present a new way of using Moving Least Squares to calculate the gradient of rotation sum for mesh-free data sets. What's more, we are the first to apply asymmetric tensor field analysis on particle based fluid simulation. Furthermore, we utilize a number of visualization techniques on different analysis results. We present why choosing a proper visualization is crucial to reveal useful information, and we also demonstrate how to utilize transfer functions to decrease perturbations of data sets. Lastly, we compare different analysis results, such as FTLE versus gradient of rotation sum.

Our methods are also useful to enhance the rendering of SPH simulation results,

which reveals many small-scale detailed flow behaviors that would not be seen using existing rendering approaches. Our results are more realistic in terms of revealing the underneath behaviors of fluid simulation.

## Contents

1	Introduction			
	1.1	Background	1	
	1.2	Motivation	2	
	1.3	My Contribution	4	
	1.4	Related Work	5	
	1.5	The Structure of the Thesis	6	
<b>2</b>	Ove	erview of the System	7	
3	SPI	H Fluid Simulation	11	
	3.1	Particle-based Fluid Simulation	11	
	3.2	Weakly Compressible SPH Model	14	
	3.3	Boundary Conditions	16	
4	Dat	a Analysis	19	
	4.1	FTLE	19	
	4.2	Rotation Sum	22	
	4.3	Gradient of Rotation Sum	25	
	4.4	Magnitude of Velocity	28	
	4.5	Density	30	
	4.6	Asymmetric Tensor Field Analysis	31	

<b>5</b>	Vist	ualization and Fluid Rendering	32			
	5.1	RGB and HSV	32			
	5.2	Apply Transfer Function and HSV to RGB Mapping	35			
	5.3	Blue, White, and Red	36			
	5.4	Gray Color	38			
	5.5	Black, Blue, and White	39			
	5.6	Dark Blue and Light Blue	40			
6	$\operatorname{Res}$	ults and Conclusion	42			
	6.1	Results	42			
		6.1.1 Comparisons Between Different Visualization Results	42			
		6.1.2 Visualization Results of Complex Flow	47			
		6.1.3 Comparisons Between Two SPH Simulations	51			
	6.2	Conclusion	54			
Bi	Bibliography					

# List of Figures

2.1	SPH Solver	9
2.2	Analysis Framework	10
3.1	Smooth Kernel Functions	13
3.2	Partition the Space and Choose Neighbors	14
3.3	Comparison Between Tait and Gas Equation	16
3.4	Gaps Around Boundary	17
4.1	Traditional FTLE	20
4.2	FTLE Result for DoubleGyre	21
4.3	FTLE Result for DoubleGyre by using MLS	21
4.4	FTLE of Mesh-free Dateset	22
4.5	Different Rotation Behaviors	23
4.6	A Particle Advects with the Flow	24
4.7	Rotation Sum for DoubleGyre with Blue, White, and Red Color Scheme	24
4.8	Two Particles Advect with the Flow	26
4.9	Gradient of Rotation Sum Result for DoubleGyre by using MLS, and with Blue, White, and Red Color Scheme.	27
4.10	Limitation of FTLE Analysis	28
4.11	Magnitude of Velocity of SPH Fluid Simulation	29
4.12	Density of SPH Fluid Simulation	30

4.13	Asymmetric Tensor Field Analysis for DoubleGyre by Using MLS	31
5.1	RGB Model	34
5.2	The Color Changes Rapidly in Only One Step without Transfer Func- tion.	35
5.3	Rotation Sum Result for DoubleGyre by Using MLS, and with Blue, White, and Red Color.	37
5.4	Gradient of Rotation Sum Result for DoubleGyre with Gray Color. $% \mathcal{S}_{\mathcal{S}}$ .	38
5.5	Gradient of Rotation Sum Result for DoubleGyre by Using MLS, and with Blue, Black, and White Color.	40
5.6	FTLE Result for DoubleGyre by Using MLS, and with Dark Blue/Light Blue	41
6.1	Fluid Rendering Based on Density (Viscosity = $0.01$ )	43
6.2	Fluid Rendering Based on FTLE (Viscosity = $0.01$ )	44
6.3	Fluid Rendering Based on Rotation Sum (Viscosity = $0.01$ )	45
6.4	Fluid Rendering Based on Gradient of Rotation Sum (Viscosity = $0.01$ )	46
6.5	Fluid Rendering Based on Magnitude of Velocity (Viscosity = 0.01) $% \left( \left( {\rm Viscosity} \right) \right) = 0.01 \left( {\rm Viscosity} \right) \right)$	47
6.6	FTLE for Complex Fluid Behaviors (Viscosity = $0.01$ )	48
6.7	Rotation Sum for Complex Fluid Behaviors (Viscosity = 0.01) $\ . \ . \ .$	49
6.8	Gradient of Rotation Sum for Complex Fluid Behaviors (Viscosity $= 0.01$ )	50
6.9	FTLE for Two SPH Simulations (Left: Viscosity = $0.003$ Right: Viscosity = $0.01$ )	51
6.10	Rotation Sum for Two SPH Simulations (Left: Viscosity = $0.003$ Right: Viscosity = $0.01$ )	52
6.11	Gradient of Rotation Sum for Two SPH Simulations (Left: Viscosity $= 0.003$ Right: Viscosity $= 0.01$ )	53

### Chapter 1

## Introduction

#### 1.1 Background

In physics, fluid not only means liquids, but also includes gases, plasmas, and plastic solids. Fluids are ubiquitous in our lives, which is why fluid simulation is almost always a hot research topic in Computer Graphics, which tries to duplicate various fluid effects in digital worlds for movies, commercials, and games.

In addition to generating special effects, computational fluid dynamics (CFD) is widely used in many critical scientific and engineering applications, such as automotive and aircraft design, mechanical engineering, environment, oceanography, climate study, plasma physics, etc. There are many techniques for fluid simulation. The two most common methods are Eulerian grid-based method and smoothed particle hydrodynamics (SPH) method. There is also a newly invented Lattice Boltzmann method[15]. Modern theory of fluid dynamics stems from the well-known Navier-Stokes equations by Claude Navier and George Stokes in the 19th century. These equations describe the conservation of momentum. Together with two additional equations for mass and energy conservation, they enable us to numerically simulate the fluid flow. In general, all CFD methods are making use of the Navier-Stokes equations to describe the physics of fluids.

In CFD, two different points of view are usually employed to describe the physical governing equations-Navier-Stokes equations. Eulerian, the grid-based method, divides the fluid into grids and tracks quantities such as velocity and density at each vertex of each grid. Despite the accuracy of numerical calculation, grid-based method sacrifices its speed and suffers from mass loss compared to SPH method. In contrast, in SPH method, we simulate the fluid as a large number of discrete particles, and each particle carries various quantities, such as mass, velocity, density, etc. This method looks at the flow behavior from a Lagrangian point of view. It allows us to conserve the mass of the fluid and simulate free-surface flows, which explains why we have seen an exponential growth of the use of SPH method during the last decade. Although all CFD methods achieved some notable successes in the past, various limitations reside in different approaches, even when they are used collaboratively.

#### 1.2 Motivation

In movies and games, no matter which methods is chosen for the fluid simulation, as long as human observers are satisfied with the visual representation of the fluid effects, the simulation results are sufficient. Whereas in physics, engineering, and mathematics, we need more rigorous error metrics to ensure that the simulation also complies with the laws in physics.

The computational fluid dynamics model has been extensively studied in Computer Graphics. The majority of the existing work focuses on two aspects of the simulations, performance and accuracy, measured by human observance. For example, the improvement of the performance of the fluid simulation is achieved by utilizing the power of GPU[9], or by splitting the fluid into different levels [17]. On the other hand, to make the fluid simulation more realistic, some research has been focused on using new formula to calculate density force [1], or introducing new algorithm to reconstruct the fluid surface 20. However, very few has investigated how to reveal the underneath behaviors of the fluid by analyzing the simulation data. For example, humans can observe vortex and undercurrent in the fluid by naked eyes, but can neither discover the micro details of exteriors nor the interior dynamics of those vortex and undercurrent by visual inspections. Currently, SPH method enables us to conduct numerical experiments rather than expensive real experiments, and in some occasions, impractical experiments. Understanding those behaviors helps us to generate more realistic fluid simulation and understand similar phenomena in the real world. Locating the places where these important behaviors, or in other words features, occur will help develop an efficient SPH simulator that assigns more samples to the places where the simulation needs to be computed with higher accuracy in order to capture the critical behaviors of the flow there. All these motivate this work.

#### **1.3** My Contribution

This thesis makes the following contributions to the SPH simulation community. Particularly, a number of analyses have been utilized to help understand the detailed flow behaviors encoded in the SPH simulation data, which in turns helps create informative visualization of the simulation results.

- FTLE (Finite-time Lyapunov exponent): we modify traditional FTLE by using Moving Least Squares to calculate the deformation matrix, and extend the usage from mesh-based to mesh-free data sets; we are the first to apply FTLE on free surface SPH fluid simulation.
- Rotation sum and gradient of rotation sum: we are the first to apply rotation sum and gradient of rotation sum on particles based fluid simulation. We also present a new way of using Moving Least Squares to calculate the gradient of rotation sum for mesh-free data sets.
- Asymmetric tensor field: we are the first to apply asymmetric tensor field analysis on particle based fluid simulation.
- We utilize a number of visualization techniques on different analysis results. We present why choosing a proper visualization is crucial to reveal useful information, and we also demonstrate how to utilize transfer functions to decrease perturbations of data sets. We also compare different analysis results, such as FTLE versus gradient of rotation sum.

#### 1.4 Related Work

Smoothed Particle Hydrodynamics(SPH) was first invented to solve astrophysical related problems. The earliest attempt to numerically calculates the incompressible fluid with free surface was introduced by Harlow and Welch (1965)[8]. Then, Foster and Metaxas (1996)[6] presented how to solve the Navier-Stokes equations in 3D. Muller, Charypar and Gross (2003) presented a simple way to utilize SPH method for achieving a real-time fluid simulation[11]. They introduced several kernel functions to interpolate different kinds of quantities involved in fluid simulation. Since then, the SPH method has been experiencing an exponential growth during the last decade. Becker and Teschner (2007) proposed a new transfer function to convert the density of a particle to the density force on it, to create weakly compressible fluid simulation for free surface flows[1].

At the same time, some researchers explore new ways to conduct different kinds of fluid simulation, for example, Clavet, Beaudoin and Poulin (2005) presented a new SPH based method for Viscoelastic Fluid Simulation[3]. Because SPH is so powerful, Muller *et al.* (2005) worked further to simulate the interaction between different kinds of particle based fluid simulations[12].

Since SPH is a particle-based method, it is necessary to reconstruct the fluid surface. However, extracted smooth and temporally coherent fluid surface over time is challenging. Most existing work avoids this step by using a large number of particles in the SPH simulation. So far, extracting fluid surface is still an open research topic. Recently, Yu and Turk (2013) introduced a new way to reconstruct fluid surface for SPH simulation by using some Anisotropic Kernels[20]. In order to create more realistic fluid simulation, people usually create as many particle as possible, and thus, how to improve the SPH simulation performance is also a hot research topic. Yan *et al.* (2009) invented a way to dynamically change the kernel size of SPH fluid simulation, and then increase the speed of calculation[19]. On the other hand, Solenthaler and Gross (2011) simply created two different scales for particle simulation, in which active areas are simulated by smaller particles[17]. Krog and Elster (2010) also utilized the power of Graphical Processing Units (GPUs) to significantly improve the performance of the simulation[9].

However, there is little work on the analysis of the behaviors of SPH fluid, let alone to utilize those analysis results to improve the performance and rendering. In the thesis, we propose a new approach to calculate the Finite Time Lyapunov Exponent (FTLE) along with other analysis methods to analyze the SPH fluid simulation. Then, we validate our analysis results by comparing them with those generated with conventional methods for the mesh-based setting. Finally, we show our fluid rendering simulation based on various analysis results.

#### 1.5 The Structure of the Thesis

The rest of the thesis is organized as follows: The overview of the whole system is provided in Chapter 2. Chapter 3 introduces the classical SPH fluid simulation method and some improvements. Different analysis methods are detailed in Chapter 4. Chapter 5 describes a number of visualization techniques. Chapter 6 provides the results, discussions and conclusions.

## Chapter 2

## **Overview of the System**

The whole system includes two components:

- 1. A SPH solver that is used to generate particle based vector field data sets. The detailed tasks of this component are as follows.
  - (a) In order to generate different kinds of fluid simulation, we adjust the parameters for SPH, such as gas constant, rest density, viscosity, step size, and gravity;
  - (b) While running the SPH simulation, the position and velocity of each particle is saved over time.
- 2. Analysis framework that first parses the data sets, and then applies different methods to analyze the data sets. Lastly, we utilize various visualization techniques to display the results. The detailed tasks of this component are as follows.

- (a) Load data sets generated by SPH solver, and parse the data;
- (b) Calculate the FTLE of each particle, and use CUDA to improve the performance;
- (c) Calculate the Rotation Sum and Gradient of Rotation Sum on runtime;
- (d) Calculate the neighbors of each particle by using triangulation;
- (e) Apply transfer functions to analysis results;
- (f) Combine the results of triangulation and transferred analysis results, and select different visualization techniques to visualize the results;
- (g) Output the results.

Below are the work flows of these two components:



Figure 2.1: SPH Solver



Figure 2.2: Analysis Framework

### Chapter 3

## **SPH Fluid Simulation**

#### 3.1 Particle-based Fluid Simulation

Smoothed-particle hydrodynamics (SPH) is a computational method used for simulating fluid flows. It is an interpolation method for particle systems.

In SPH method, we simulate the fluid as a large number of discrete particles, and each particle carries various quantities, such as mass, velocity, density, etc. It allows us to conserve the mass of the fluid and simulate free-surface flows. These particles have a spatial distance, over which their properties are calculated by a kernel function. This means that the physical quantity of any particle can be obtained by summing the relevant properties of all the particles that lie within the range of the kernel. The equation for any quantity A at any point r is given by the equation:

$$A_{s}\left(r\right) = \sum_{j} m_{j} \frac{A_{j}}{\rho_{j}} W(r-r_{j},h)$$

The function W(r, h) is called a kernel function with smoothing kernel size h. In practice, we need to normalize the kernel function such that:

$$\int W(r)dr = 1$$

As we discussed above, each particle in SPH method carries various quantities, which generate various forces naturally. For example, pressure force is generated because of the pressure differences between different partitions of the fluid. Viscosity force is generated because of the relative velocity between different particles, and fluid particles tend to attract each other when they tend to separate from each other. For simplicity, we only consider three kinds of forces: pressure force, viscosity force and external force. According to modern theory of fluid dynamics, the distance between two particles is closer, their mutual influence should be larger. Thus, we need to choose appropriate kernel functions to calculate different quantities. In the thesis, we use the kernel functions proposed in [11].

The  $W_{\text{poly6}}(r, h)$  is used to calculate the pressure of each particle:

$$W_{\text{poly6}}(r,h) = \begin{cases} \frac{315}{64\pi h^9} (h^2 - r^2)^3 & 0 \le r \le h \\ 0 & otherwise \end{cases}$$

The  $W_{\text{spiky}}(r, h)$  is used to calculate the pressure force of each particle:

$$W_{\rm spiky}(r,h) = \begin{cases} \frac{15}{\pi h^6} (h-r)^3 & 0 \le r \le h \\ 0 & otherwise \end{cases}$$



Figure 3.1: From left to right are Wpoly6, Wspiky, and Wviscosity smoothing kernels.[13]

•

The  $W_{\text{viscosity}}(r, h)$  is used to calculate the viscosity force of each particle:

$$W_{\text{viscosity}}(r,h) = \begin{cases} \frac{15}{2\pi h^3} \left(-\frac{r^3}{2h^3} + \frac{r^2}{h^2} + \frac{r}{2h} - 1\right) & 0 \le r \le h \\ 0 & otherwise \end{cases}$$

First we calculate the density of each particle:

$$\rho_{s}(r) = \sum_{j} m_{j} \frac{\rho_{j}}{\rho_{j}} W(r-r_{j},h) = \sum_{j} m_{j} W(r-r_{j},h)$$

Then we calculate the pressure force based on the obtained density:

$$f_i^{pressure} = -\sum_j m_j \frac{p_j}{\rho_j} \nabla W(r - r_j, h)$$

At last, we calculate the viscosity force:

$$f_{i}^{\text{viscosity}}{=}\mu\sum_{j}m_{j}\frac{v_{j}{-}v_{i}}{\rho_{j}}\nabla^{2}W\left(r{-}r_{j},h\right)$$



Figure 3.2: Partition the Space and Choose Neighbors

So, we can use the following equation to get the acceleration of each particle.

$$a_i {=} \frac{f_i}{m_i} {=} \frac{f_i^{\mathrm{pressure}} {+} f_i^{\mathrm{viscosity}} {+} \rho_i g}{m_i}$$

where g is an external force density field. In general, g stands for gravity field. Since we know the acceleration of each particle, and its original velocity and position, we are able to get each particle's new position.

#### 3.2 Weakly Compressible SPH Model

Water and other liquids are known low compressible fluids. However, there is an obvious issue of the SPH method proposed in [11]; the fluid simulation is not really

incompressible, which makes the simulation not quite realistic. So we use a different computation equation called Tait equation of state to convert density to density force.

In [11], the density force is calculated by the following formulas:

$$P = k\rho$$

where k is a gas constant that depends on the temperature. To make the simulation more stable, it is modified by the following version:

$$P = k\left(\rho - \rho_0\right)$$

where  $\rho_0$  is called rest density. Theoretically, introducing rest density should not change the gradient of the pressure field. However, it permits us to make the simulation more stable.

In our SPH solver, we tried the modified version first, but the result is still significantly compressible in most cases, so we moved to the Tait equation:

$$P = B\left(\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right)$$

In our simulations, we set  $\gamma = 4$ .



Figure 3.3: Comparison Between Tait and Gas Equation[1]

#### **3.3** Boundary Conditions

The boundary conditions are still an open topic in SPH method. When a particle approaches boundary region, its density is much lower than inner particles which reside in the interior of fluid, because we are using an isotropic kernel to compute the density. In order to maintain comparable density with inner particles, particles close to the boundary clump. And then, it causes the density of inner particles be too large. Thus, as we can see from Figure 3.4, there is a gap between inner particles and boundary particles. There are three popular methods to solve this problem: ghost particles, repulsive particles, and dynamic particles.

Ghost particles, as described by Randles and Libersky (1996)[14], are temporary copies of a real particle which moves close to the boundary at a distance closer than



Figure 3.4: Gaps Around Boundary

smoothing kernel r. The ghost particles carry the same density and pressure with the original particles, but opposite velocity. At each time step, the number of ghost particles changes, and it makes the implementation of this method much complicated, especially when the boundaries have complex geometries.

Repulsive particles are boundary particles that exert forces on fluid particles. Monaghan and Kos (1999) [10] refined the method and minimized the inter-spacing effect of the boundary particles on the repulsion force of the boundary. One of the main drawbacks of this method is the computational complexity in complex scenes.

Dynamic particles are fixed particles[4] or particles that move according to some external functions[5] outside the boundary. The most interesting advantage of this method is the computational simplicity. We are able to apply normal smoothing equations on fluid particles. However, when fluid particles separate from boundaries, the density decreases locally. Thus, some particles will remain stuck on the boundary. In addition, we need to carefully place dynamic particles based on different kinds of SPH implementations to get plausible results.

In summary, none of these methods are perfect. In my simulations, I implement dynamic particles for the boundaries. The dynamic particles are fixed particles outside the boundaries which are carefully placed. They have the same mass with other particles. When a particle approaches boundary region, it can have proper density by summing densities generated by dynamic particles.

## Chapter 4

## Data Analysis

#### **4.1 FTLE**

The finite-time Lyapunov exponent, FTLE, is calculated from the trajectories of particles over a finite-time period T. It is a scalar value which characterizes the amount of stretching about the trajectory of point  $x \in D$  over the time interval [t, t+T]. For grid-based data sets, we simply solve a deformation matrix, and then analyze the eigenvalue of the matrix:

$$\begin{pmatrix} x'\\y' \end{pmatrix} = \begin{pmatrix} T_{11} & T_{12}\\ T_{21} & T_{22} \end{pmatrix} \begin{pmatrix} x\\y \end{pmatrix} + \begin{pmatrix} O_{11} & O_{12}\\ O_{21} & O_{22} \end{pmatrix} \begin{pmatrix} 1\\1 \end{pmatrix}$$



Figure 4.1: After T steps, the new positions of neighbor particles. Star stands for the target position where we want to get the FTLE. Blue balls stand for the neighbor particles.

$$\begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix}$$
 is the deformation matrix. Because we have 4 particles, and 8 equa-

tions, we introduce an offset matrix  $\begin{pmatrix} O_{11} & O_{12} \\ O_{21} & O_{22} \end{pmatrix}$  to get a unique solution. In some cases, the eigenvalue of the deformation matrix may be imaginary number, so the

Cauchy Green deformation tensor  $\triangle$  is computed:

$$\triangle = \left(\begin{array}{cc} T_{11} & T_{12} \\ T_{21} & T_{22} \end{array}\right)^* \left(\begin{array}{cc} T_{11} & T_{12} \\ T_{21} & T_{22} \end{array}\right)$$

And then, the FTLE is:

$$\sigma_{t_0}^T(x) = \frac{1}{|T|} ln \sqrt{\lambda \max(\Delta)}$$

For SPH data sets, the neighbor particles are no longer regularly placed around our target position. And FTLE calculation is very sensitive to the neighbor information. So we moved to another method to calculate the deformation matrix.



Figure 4.2: FTLE Result for DoubleGyre[7]



Figure 4.3: FTLE Result for DoubleGyre by using MLS



Figure 4.4: After T steps, the new positions of selected neighbor particles. Pink ball stands for the target position where we want to get the FTLE. Blue balls stand for the neighbor particles.

We define a kernel size, and select all particles around the target position within the kernel, and do a least square method analysis. We want to minimize the following formula[16]:

$$\sum_{n} \left( S_{new_n}^{i} - A^{ij} S_{old_n}^{j} T \right)^2$$

where  $A^{ij}$  corresponds to the four parameters in the previous deformation matrix.  $S_{new}$  and  $S_{old}$  stand for the separations between each neighbor particle and the target particle.

#### 4.2 Rotation Sum

The movement that an object rotates around a point or center is called rotation. For a three-dimensional object, the center is an imaginary line, called rotation axis. If the axis passes through its center of mass the body within the body, the rotation is called spin. If the rotation is around an external point, it's called a revolution or orbital revolution, e.g. Earth rotates around the Sun because of gravity.

As we can see from Figure 4.5, for 2D vector fields, particles at different sides of the flow separation structure have rather different rotation behaviors.



Figure 4.5: Particles at different sides of the flow separation structure have rather different rotation behaviors.

The rotation sum is calculated from the trajectories of particles over a finite-time period T. Given a smooth integral curve and any point, the rotation of the curve at that point in the integral direction is defined as the angle change of the tangent vectors. The total rotation is defined by summing up all these angle changes. It is a scalar value which characterizes the amount of rotations about the trajectory of point  $x \in D$  over the time interval [t, t + T].



Figure 4.6: A Particle Advects with the Flow

Let's connect the previous position and current position of a particle, it will be a vector. And then connect the current position with the next position, it will be another vector. The angle difference between these two vectors is called a rotation. And we sum all those rotations over a time interval T.



Figure 4.7: Rotation Sum for DoubleGyre with Blue, White, and Red Color Scheme

#### 4.3 Gradient of Rotation Sum

The gradient of the rotation field  $\phi$ , which is a scalar field that is computed as:

$$\nabla \phi = \left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}, \frac{\partial \phi}{\partial z}\right)$$

Rotation sum is good to partition the fluid into different parts which behave differently from others. The boundaries of those separate parts always reveal useful information. In some cases, these boundaries correspond to FTLE ridges, but generally they encode more detailed information that FTLE fields. The gradient of a scalar field at any point reveals the direction in which the scalar value increases the most. Based on our observation, the places that have large gradient typically correspond to the boundaries of different regions that have rather different rotation behaviors as shown in Figure 4.7.



Rotation of two particles from t0 to T + t0

Figure 4.8: Two Particles Advect with the Flow

If two particles move to different direction, the change radio between their rotation sums will be larger than those who move to the same direction. We use Moving Least Square method to extract the gradient of the rotation sum. We want to minimize the following formula:

$$\sum_{n} \left( R_n - A^i S_n^i \right)^2$$

where  $A^i$  corresponds to the two parameters in the tensor matrix  $\begin{pmatrix} T_x \\ T_y \end{pmatrix}$ . *R* stands for the rotation difference between each neighbor particle and the target particle. *S* stands for the separations between each neighbor particle and the target particle.

Then, the gradient of rotation sum is calculated by the following formula:

$$grad = \sqrt{T_x^2 + T_y^2}$$



Figure 4.9: Gradient of Rotation Sum Result for DoubleGyre by using MLS, and with Blue, White, and Red Color Scheme.

When compared to FTLE result (Figure 4.2, Figure 4.3), rotation sum analysis encode more information of the flow dynamics. This is because FTLE computation only utilizes the end positions of the particles in the flow map computation; therefore, important information that is encoded in the intermediate part of the pathlines could be missed.



Figure 4.10: Limitation of FTLE Analysis

#### 4.4 Magnitude of Velocity

Velocity is something which is very closely related to speed. For people who are not specialists, the used of the word "speed" to explain how fast something is moving is normally sufficient. The term velocity is used by physicists when certain problems arise with the use of the term speed. The big difference can be noticed when we consider movement around a circle. When something moves in a circle and returns to its starting point its average velocity is zero but its average speed is found by dividing the circumference of the circle by the time taken to move around the circle. This is because velocity is calculated by only considering the distance between the starting and the end points while speed considers the total distance traveled.



Figure 4.11: Magnitude of Velocity of SPH Fluid Simulation

Generally speaking, the particles tend to carry larger speed in active areas, such as surface, and splash. So we simply analyze the magnitude of velocity:

$$S=\sqrt{v_x^2+v_y^2}$$

where  $v_x$  and  $v_y$  denote the velocity of the particle along x and y direction respectively.



Figure 4.12: Density of SPH Fluid Simulation

#### 4.5 Density

The desity, usually referred to volumetric mass density, of a substance is the mass per unit volume. Symoblically, it's most often represented by  $\rho$  (the lower case Greek letter rho). Mathematically, it's defined as mass divided by volume:

$$\rho = \frac{m}{V}$$

where  $\rho$  is the density, m is the mass, and V is the volume. Occassionally, it may be referred to weight per unit volume, e.g. oil and gas industry, density is loosely defined as its weight per unit volume, however, this is scientifically inaccurate – this quantity is more properly called specific weight. In general, the particles tend to carry smaller density in active areas, such as surface and splash.

#### 4.6 Asymmetric Tensor Field Analysis

Given a vector field  $\begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix}$ , it can be decomposed into three parts: rotation, stretching, and dilation[21]:

$$T = \gamma_{d} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \gamma_{r} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} + \gamma_{s} \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix}$$

where  $\gamma_{\rm d} = \frac{T_{11} + T_{12}}{2}$ ,  $\gamma_{\rm r} = \frac{T_{21} - T_{12}}{2}$  and  $\gamma_{\rm s} = \frac{\sqrt{(T_{11} - T_{22})^2 + (T_{12} + T_{21})^2}}{2}$  are the strengths

of isotropic dilation, rotation, and anisotropic stretching, respectively.



Figure 4.13: Asymmetric Tensor Field Analysis for DoubleGyre by Using MLS

In some cases, we can reveal interesting results by analyzing and highlighting the dominant part of a point. For instance, as we can see from Figure 4.11, the DoubleGyre vector field is partitioned into three parts, green means the dominant part is clockwise rotation, red means the dominant part is counter clockwise rotation, and white means the dominant part is anisotropic stretching.

### Chapter 5

## Visualization and Fluid Rendering

#### 5.1 RGB and HSV

The well-known RGB model is the most basic and intuitive color model. In RGB model, every pure color can be expressed as a mixture of the additive primary colors: red, green, and blue. For example, cyan, magenta, and yellow, the secondary colors of RGB, are formed by the mixture of two of the additive primary colors and the exclusion of the third. Yellow comes from the combination of red and green, cyan from green and blue, magenta from blue and red. If we add red, green, and blue together in full intensity, we get white. Naturally, when all colors of the electromagnetic spectrum converge in full intensity, white color is created.

The HSV model is another model to characterize a color. It appears as a color cone that can be described as three parameters: hue, saturation, and value.

The hue of a color refers to the primary attributes of a color. The value of hue,

typically represented as a cycle, ranges from 0 to 360. Each value of hue specifies the position of a corresponding pure color on the color wheel. For example, 0 refers to red, 40 refers to yellow, and 240 refers to blue.

The saturation of a color is a value measuring how pure the hue is with respect to a white reference. Fox example, a fully-saturated red contains only the color red and has no other primary colors or any mixture of blue or green. Slightly or minimally saturated colors contain mixtures of the primary colors and might contain some degree of red, green, and blue. White is the color that has no saturation at all. As we can see from Figure 5.2, the primary colors appear along the edges of the color wheel. They mix toward the center. When they are close to the center, they become lighter in hue and less saturated. White resides at the center, and contains equal amounts of all the colors.

The value of a color describes the brightness of a color. This is a relative description of how much light is coming from the color. We would say a color is bright if the color reflects a lot of light. We are able to adjust the brightness (brighter or darker) of a pure color by adjusting its value. As we can see from Figure 5.2, the color at the center of the HSV model changes from white to black. The color grows to gray, and finally becomes black if followed along the center of HSV model along up-down direction.

In summary, in HSV model, colors are mixed within a single-cone shape based on the mixture values of hue, saturation, and value. Colors appear as primaries along the edges. For example, from a clockwise viewpoint, the colors following green on the perimeter are yellow, red, magenta, blue, cyan, and then, back to green. White



Figure 5.1: RGB Model[18] and HSV Model[2]

is at the center of the cone where all the colors are mixed evenly in full intensity. Since most of our analysis results are scalar values, the HSV model is a good fit for us to transfer the analysis results to RGB color which is well supported by OpenGL. And then we can easily identify useful information in the obtained visualization.

# 5.2 Apply Transfer Function and HSV to RGB Mapping

Because the maxima and minima values of different analysis results vary a lot, but when we are doing HSV to RGB mapping, the HSV value is determined by the following formula:

$$\mathrm{hsv}\left[0\right] \!=\! \frac{\mathrm{quantity} - \mathrm{minima}}{\mathrm{maxima} - \mathrm{minima}}$$

$$hsv[1] = hsv[2] = 1$$



Figure 5.2: The Color Changes Rapidly in Only One Step without Transfer Function.

According to the formula, we can easily find that the mapping colors of the majority particles are affected by the maxima and minima. Even if these two values are kept the same over time, the mapping colors of individual particles still rapidly change over time. In order to solve this problem, we applied a transfer function on the analysis results, and make the visualization more stable and consistent:

$$mean = \sum_{n} \frac{quantity_i}{n}$$

The standard deviation of the vector field is:

/

$$\sigma = \sqrt{\sum_{n} \frac{\left(\text{quantity}_i - \text{mean}\right)^2}{n}}$$

Then the original quantities are transferred by the following formula:

$$q_{New} = \begin{cases} newMaxima(if q_{Old} > mean + \alpha \times \sigma) \\ q_{Old} \\ newMinima(if q_{Old} < mean - \beta \times \sigma) \end{cases}$$

where  $\alpha$  and  $\beta$  are transfer parameters, and can be changed on runtime. We can manipulate the maximum and minimum thresholds by adjusting the values of  $\alpha$  and  $\beta$ . The new color mapping formula changes to:

$$hsv [0] = \frac{q_{New} - newMinima}{newMaxima - newMinima}$$

#### 5.3 Blue, White, and Red

By simply doing HSV to RGB mapping, the color range will be very large, and it is not good to determine and reveal the useful information in some cases. For instance, the most interesting information in the rotation field is the opposite rotation behaviors in the flow, i.e., counter clockwise rotation versus clockwise rotation. To better convey this information in the visualization, a blue-white-red color scheme can be adopted, as this color scheme is good to emphasize the values close to maxima and minima. With the blue-white-red scheme, the positive (or counter clockwise) rotation will correspond to the areas colored by red, while the negative (clockwise) rotation will correspond to blue. White color will be used to indicate the places that are transiting from a positive rotation area to a negative one, and vice versa.



Figure 5.3: Rotation Sum Result for DoubleGyre by Using MLS, and with Blue, White, and Red Color.

The blue-white-red color coding can be implemented as follows after applying the same transfer function as described above:

$$hsv [1] = \frac{q_{New} - newMinima}{newMaxima - newMinima}$$

if hsv [1] < 0, then hsv [0] = 0, hsv [1] = abs (hsv [1])

else if  $hsv[1] \ge 0$ , hsv[0] = 240

#### 5.4 Gray Color

In some cases, only the values close to maxima need to be highlighted. For this purpose, gray color coding scheme can be used. After we applied similar transfer function on the original data sets, we set the R, G, and B channels by using the following formula:

$$\operatorname{rgb}[0] = \operatorname{rgb}[1] = \operatorname{rgb}[2] = \frac{q_{New} - newMinima}{newMaxima - newMinima}$$

At the same time, we switch the background color to white. According to the above formula, if a particle carries a large quantity, it will be darker. Vice versa, if it carries a small quantity, it will be whiter, and then it is possible to become invisible because it gets the same color with the background.



Figure 5.4: Gradient of Rotation Sum Result for DoubleGyre with Gray Color.

#### 5.5 Black, Blue, and White

This technique is very interesting. It is able to partition the vector field into black and white parts, and at the same time, the boundaries between those two parts are colored in blue. While the background color is also black, the black parts become invisible. What is more, there is a new parameter T. If we adjust the value of T, we are able to change the partition conditions, and then change the size of the black areas, and the width of the boundaries. T is defined by the following formula:

newMaxima = mean+  $\alpha \times \sigma$ 

newMinima = mean- $\beta \times \sigma$ 

 $T = (newMaxima - newMinima) \times \gamma + newMinima$ 

where  $\gamma$  is a coefficient that can range from 0 to 1. After we get the *T*, we calculate the HSV based on it.

if quantity 
$$> T$$
, hsv  $[0] = 240$ , hsv  $[1] = 1$ 

$$hsv [2] = 1 - \frac{quantity - T}{newMaxima - T}$$

else if quantity  $\leq T$ , hsv [0] = 240, hsv [2] = 1

$$hsv [1] = \frac{quantity - T}{T - newMinima}$$

Then, we simply do HSV to RGB mapping.



Figure 5.5: Gradient of Rotation Sum Result for DoubleGyre by Using MLS, and with Blue, Black, and White Color.

#### 5.6 Dark Blue and Light Blue

In general, the natural color of water is blue. In order to utilize those analysis results, we try to transfer those results into different magnitude of blue color to make the visualization more realistic. This technique is a good fit for rendering of the simulation results. In contrast to the previous methods, map the scalar value to a range of hsv[1]. The range is [0.5, 1] in our experiments. First of all, we still apply transfer functions on analysis results, then:

 $hsv [1] = 0.5 + 0.5 \times \frac{q_{New} - newMinima}{newMaxima - newMinima}$ 

$$hsv[0] = 240, hsv[2] = 0$$



Figure 5.6: FTLE Result for DoubleGyre by Using MLS, and with Dark Blue/Light Blue

## Chapter 6

## **Results and Conclusion**

#### 6.1 Results

We use 12,800 particles to simulate the fluid. First, we compare and discuss different visualization results by utilizing different analysis results. Then, we demonstrate the analysis results for some complex fluid behaviors. At last, we present the comparison results of FTLE, rotation sum and gradient of rotation sum between fluids with different viscosities.

#### 6.1.1 Comparisons Between Different Visualization Results

First of all, we show and compare different visualization results by utilizing different analysis results. We select two time steps of fluid to demonstrate each analysis result. The left pictures are selected various visualization techniques, and the right pictures





Figure 6.1: Fluid rendering based on density (viscosity = 0.01)

Red means higher density, blue means lower. As we can see, most of the surface particles of the fluid are blue, which corresponds to lower density region. But some surface particles also carry similar density with non-surface particles. We are unable to simply use density result to improve the fluid simulation.



Figure 6.2: Fluid rendering based on FTLE (viscosity = 0.01)

Red means higher FTLE, blue means lower. FTLE of the wave is larger than other areas. Based on the definition of FTLE, the result is reasonable, and it also predicts the stretching region. FTLE measures the stretch of each point in a vector field, and generally speaking, the separation regions carry larger FTLE value. By using dark and light blue visualization, we can see more layers of the fluid simulation based on the FTLE result. However, for free surface vector field, the stretch of each point is closer to others, and many useful information are unable to be captured by FTLE analysis, which make FTLE result not so obvious sometimes. We need more sensitive analysis method to analyze the behavior of the SPH fluid simulation. Another drawback of FTLE analysis for free surface flow is we cannot do backward FTLE analysis due to the always forward moving natural of the simulation. Actually, we miss half of the FTLE analysis result. The same drawback comes with all other analysis methods for free surface flow as well.



Figure 6.3: Fluid rendering based on rotation sum (viscosity = 0.01)

Red means counter clockwise rotation region, blue means clockwise rotation region, white means transition region. After the fluid collides with the boundary, it splits into several partitions in terms of movement directions. The rotation sum method identifies the clockwise rotation particles, and counter clockwise particles. Those white particles are transiting from clockwise to counter clockwise, or vice versa.



Figure 6.4: Fluid rendering based on gradient of rotation sum (viscosity = 0.01)

The result of gradient of rotation sum reveals the boundary of different rotation partitions. In reality, those are the most complex and interesting areas. FTLE measures the stretch of each point in a vector field, and generally speaking, the separation regions carry larger FTLE value. However, for free surface vector field, the stretch of each point is closer to others, which make FTLE not so useful. The gradient of rotation sum encodes more information of the flow dynamics.



Figure 6.5: Fluid rendering based on magnitude of velocity (viscosity = 0.01)

The magnitude of velocity of the wave is much larger than other areas. We can even see the transitions of the magnitude of velocity field.

We demonstrate how to utilize the analysis results to improve the rendering. Our results is more realistic in terms of revealing the underneath behaviors of fluid simulation. We also show the comparisons between different visualization results based on various analysis results.

#### 6.1.2 Visualization Results of Complex Flow

Now, we start with two separate blocks of fluid instead of one block.



Figure 6.6: FTLE for complex fluid behaviors (viscosity = 0.01)

At the beginning, we can easily highlight the boundaries which are also the most stretching areas based on the FTLE. After the two partitions merge together, the most active areas at different steps always carry larger FTLE.



Figure 6.7: Rotation sum for complex fluid behaviors (viscosity = 0.01)

The two partitions move in opposite directions, and we can easily notice it based on the rotation sum result. After the two partitions merge together, we still can identify those two parts based on rotation sum.



Figure 6.8: Gradient of rotation sum for complex fluid behaviors (viscosity = 0.01)

Even though the two partitions move in opposite directions, they should have the same gradient of rotation sum result, and our result proves it. We are also able to notice different rotation regions based on it.



#### 6.1.3 Comparisons Between Two SPH Simulations

Figure 6.9: FTLE for Two SPH Simulations (Left: Viscosity = 0.003 Right: Viscosity = 0.01)

From the result, we can easily tell the similarities between two different fluid simulations, because they have similar large-scale patterns in the FTLE fields. FTLE just measures the separation rate of the nearby particles, it smooth the difference between different moving paths as long as the path has the same start point and end point. In addition, we can notice that the surface area of the left pictures sometimes carry larger FTLE, that is due to the reason that the viscosity force is lower in left fluid simulation, the surface tend to be more perturbations on the surface area. Intuitively, FTLE result of larger viscosity fluid should be smoother, our result also prove it.



Figure 6.10: Rotation Sum for Two SPH Simulations (Left: Viscosity = 0.003 Right: Viscosity = 0.01)

When we change the viscosity of the fluid, the rotation behavior of fluid changes as well. When compared to FTLE result, rotation sum analysis encodes more information of the flow dynamics. This is because FTLE computation only utilizes the end positions of the particles in the flow map computation; therefore, important



information that is encoded in the intermediate part of the pathlines could be missed.

Figure 6.11: Gradient of Rotation Sum for Two SPH Simulations (Left: Viscosity = 0.003 Right: Viscosity = 0.01)

Since rotation sum captures more moving information than FTLE, the gradient of rotation sum is better to reveal the boundaries of different rotation partitions. Thus, the result is less similar compared to FTLE result.

#### 6.2 Conclusion

Analysis supported SPH simulation is novel and promising. We have presented how to use fewer particles to simulate the fluid, and utilize the analysis results to improve the rendering. Our results are more realistic in terms of revealing the underneath behaviors of fluid simulation. We also show the comparisons between different visualization results based on various analysis results.

We have presented that visualization method is very critical to reveal useful information in some cases. When there are huge perturbations of original data, we need to use selected transfer function to process it, and then explore different ways to highlight interesting information.

While we are quite impressed with the analysis supported fluid simulation, applying new analysis methods and rendering the fluid surface in better way certainly remains an open research problem. There is additional work that needs to be done. First, we just apply our method on our SPH solver, which is relatively simple, only supports thousands of particles, and it is only in 2D. We are plan to utilize our analysis framework on larger data sets, and 3D. In addition, it is possible to apply rotation sum and gradient of rotation sum method on other kinds of fluid simulation, and compare those results with SPH fluid simulation.

## Bibliography

- M. Becker and M. Teschner. Weakly compressible SPH for free surface flows. In Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '07, pages 209–217, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.
- [2] A. Buckley. Colors in arcgis symbols, 2012. [Online; accessed 12-April-2012].
- [3] S. Clavet, P. Beaudoin, and P. Poulin. Particle-based viscoelastic fluid simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium* on Computer Animation, SCA '05, pages 219–228, New York, NY, USA, 2005. ACM.
- [4] A. J. C. Crespo, M. Gómez-Gesteira, and R. A. Dalrymple. Boundary conditions generated by dynamic particles in SPH methods. CMC-TECH SCIENCE PRESS, 5(3):173–184, 2007.
- [5] R. Dalrymple and O. Knio. SPH Modelling of Water Waves, chapter 79, pages 779–787.
- [6] N. Foster and D. Metaxas. Realistic animation of liquids. In *Graphical Models* and Image Processing, pages 23–30, 1995.
- [7] R. Fuchs, B. Schindler, and R. Peikert. Scale-space approaches to ftle ridges. In R. Peikert, H. Hauser, H. Carr, and R. Fuchs, editors, *Topological Methods in Data Analysis and Visualization II*, Mathematics and Visualization, pages 283–296. Springer Berlin Heidelberg, 2012.
- [8] F. H. Harlow and J. E. Welch. Numerical calculation of timedependent viscous incompressible flow of fluid with free surface. *Physics of Fluids (1958-1988)*, 8(12):2182–2189, 1965.

- [9] O. E. Krog and A. C. Elster. Fast GPU-based fluid simulations using SPH. In Proceedings of the 10th International Conference on Applied Parallel and Scientific Computing - Volume 2, PARA'10, pages 98–109, Berlin, Heidelberg, 2012. Springer-Verlag.
- [10] J. Monaghan and A. Kos. Solitary waves on a cretan beach. Journal of Waterway, Port, Coastal, and Ocean Engineering, 125(3):145–155, 1999.
- [11] M. Müller, D. Charypar, and M. Gross. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 154–159, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.
- [12] M. Müller, B. Solenthaler, R. Keiser, and M. Gross. Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Sympo*sium on Computer Animation, SCA '05, pages 237–244, New York, NY, USA, 2005. ACM.
- [13] PUKIWIKI. Kernel function of the SPH method, 2012. [Online; accessed 07-December-2012].
- [14] P. Randles and L. Libersky. Smoothed particle hydrodynamics: Some recent improvements and applications. *Computer Methods in Applied Mechanics and Engineering*, 139(14):375–408, 1996.
- [15] P. Rinaldi, E. Dari, M. Vnere, and A. Clausse. A lattice-boltzmann solver for 3d fluid simulation on GPU. Simulation Modelling Practice and Theory, 25(0):163–171, 2012.
- [16] M. Robinson. Turbulence and viscous mixing using smoothed particle hydrodynamics. Ph.D. dissertation, Department of Mathematical Science, Monash University, 2009.
- [17] B. Solenthaler and M. Gross. Two-scale particle simulation. In ACM SIG-GRAPH 2011 Papers, SIGGRAPH '11, pages 81:1–81:8, New York, NY, USA, 2011. ACM.
- [18] Wikipedia. Rgb color model, 2013. [Online; accessed 10-November-2013].
- [19] H. Yan, Z. Wang, J. He, X. Chen, C. Wang, and Q. Peng. Real-time fluid simulation with adaptive SPH. Comput. Animat. Virtual Worlds, 20(2 - 3):417– 426, June 2009.

- [20] J. Yu and G. Turk. Reconstructing surfaces of particle-based fluids using anisotropic kernels. ACM Trans. Graph., 32(1):5:1–5:12, Feb. 2013.
- [21] E. Zhang, H. Yeh, Z. Lin, and R. S. Laramee. Asymmetric tensor analysis for flow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):106–122, Jan. 2009.