Computer Aided Geometric Design 30 (2013) 557–566

Contents lists available at SciVerse ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

# Topology analysis of time-dependent multi-fluid data using the Reeb graph ☆

Fang Chen [a,*], Harald Obermaier [b], Hans Hagen [a], Bernd Hamann [b], Julien Tierny [c], Valerio Pascucci [c]

[a] Department of Computer Science, University of Kaiserslautern, Kaiserslautern 67663, Germany
[b] Institute for Data Analysis and Visualization, Department of Computer Science, University of California, Davis, One Shields Avenue, Davis, CA 95616, USA
[c] 72 South, Central Campus Drive, Salt Lake City, UT 84112, USA

ARTICLE INFO

*Article history:*
Available online 24 April 2012

*Keywords:*
Multi-phase fluid
Level set
Topology method
Point-based multi-fluid simulation

ABSTRACT

Liquid–liquid extraction is a typical multi-fluid problem in chemical engineering where two types of immiscible fluids are mixed together. Mixing of two-phase fluids results in a time-varying fluid density distribution, quantitatively indicating the presence of liquid phases. For engineers who design extraction devices, it is crucial to understand the density distribution of each fluid, particularly flow regions that have a high concentration of the dispersed phase. The propagation of regions of high density can be studied by examining the topology of isosurfaces of the density data. We present a topology-based approach to track the splitting and merging events of these regions using the Reeb graphs. Time is used as the third dimension in addition to two-dimensional (2D) point-based simulation data. Due to low time resolution of the input data set, a physics-based interpolation scheme is required in order to improve the accuracy of the proposed topology tracking method. The model used for interpolation produces a smooth time-dependent density field by applying Lagrangian-based advection to the given simulated point cloud data, conforming to the physical laws of flow evolution. Using the Reeb graph, the spatial and temporal locations of bifurcation and merging events can be readily identified supporting in-depth analysis of the extraction process.



## 1. Introduction

In chemical engineering, liquid–liquid extraction is a widely used method, where compounds of one liquid are separated by mixing it with a finely dispersed solvent (Drumm et al., 2008). Numerical simulations model this process as a multi-fluid flow with two liquid phases. These multi-fluid simulations can be used to improve the design parameters of extraction devices in order to optimize the extraction process.

The desired output of these simulations is the predicted evolution of the density distribution for each of the phases, which indicates how well the two liquids are mixed. The density of a phase in a given space denotes the fraction of the space occupied by the fluid. In liquid–liquid extraction, a finely dispersed solvent results in low density values for the second phase in large regions of the data set. Examples for Computational Fluid Dynamics (CFD) codes capable of simulating

☆ This research was supported by the International Research and Training Group at the University of Kaiserslautern (IRTG) and Deutsche Forschungsgemeinschaft (DFG, German research foundation).

* Corresponding author.
*E-mail addresses:* chen@cs.uni-kl.de (F. Chen), harald.obermaier@itwm.fhg.de (H. Obermaier), hagen@cs.uni-kl.de (H. Hagen), hamann@cs.ucdavis.edu (B. Hamann), jtierny@sci.utah.edu (J. Tierny), pascucci@sci.utah.edu (V. Pascucci).

0167-8396/$ – see front matter 
http://dx.doi.org/10.1016/j.cagd.2012.03.019

two-phase flows are Fluent (commercial), OpenFoam (open source) and finite point set method (FPM) (Kuhnert and Tiwari, 2003). The latter is used throughout this work and produces scattered point sets, carrying velocity and density information.

Typically, there are three types of multi-phase fluid models tackling different flow regimes: *volume of fluid* (VOF) modeling which focuses on tracking the interface of the two fluids for slug and surface flow; Eulerian multi-phase modeling which deals with heat and momentum transfer between the phases; and a discrete phase modeling of the mixture. The last type of modeling is widely used for simulating bubbly flow and slurry flow, such as fluid mixture in a bubble column reactor for liquid–liquid extraction. However, traditional approaches of capturing fluid material boundaries cannot handle the case where multiple regions within a cell are occupied by different fluids, as is the case in finely dispersed liquids. Thus a higher level visualization technique is required for the understanding of these flow density distributions.

We present a topology-based approach for studying the volume fraction field given by an arbitrarily distributed numerically computed point set. The simulation data we are looking at is a two-dimensional, time-varying fluid field discretized by particles with associated density and velocity values. The low time resolution of this reference data set complicates the tracking of material boundaries, as a non-physically-based interpolation scheme can result in wrong topology. For this matter, we make use of a physically-based interpolation scheme that improves correspondences between time steps of the simulation and allows for more robust feature extraction.

The major goal of this paper is to develop plausible and practical interpolation schemes for point-based multi-fluid density data, and to characterize fluid interface behavior with Reeb graphs. To identify interesting time intervals where regions that are densely occupied by a certain phase of fluid split or merge, we first define these regions by a certain *level set* of the density field. Using a physically-based interpolation scheme, we compute a time-continuous density field at a re-sample grid points. Finally we carry out a topological analysis of the extracted time-varying level sets using the Reeb graph. The major contributions of this work are as follows:

- It proposes an interpolation scheme for point-based time-dependent density data sets which have no connectivity information. The proposed interpolation scheme is capable of handling sparse data with large time intervals, preserving both the physical properties as well as topology of the flow.
- It introduces a framework to extract and analyze fluid interface topology. The framework is practical for the analysis point-based multi-fluid data sets. It offers novel views and tools for domain experts for further analysis and estimation of solvent efficiency.

The paper is organized as follows: In Section 1.1, we introduce related work about particle-based fluid simulation and topology-based feature tracking techniques. In Section 2, suitable interpolation schemes are applied separately for time and spatial direction in order to obtain a topological-clean extraction of the level sets. Section 3 contains the example and topology analysis of our result. The final section highlights some areas of possible improvements as well as future work concerning our method.

### 1.1. Related work

A *level set* of a scalar field is given by the set of points with identical scalar value. Our idea of studying the level sets of the density field was inspired by existing research which focuses on material boundary and fluid interface tracking, including the *front tracking* (*FT*) *method* (Unverdi and Tryggvason, 1992), *level set method* (LSM) (Osher and Sethian, 1988), and *volume of fluid method* (*VOF*) (Hirt and Nichols, 1981).

The FT method (Unverdi and Tryggvason, 1992; Terashima and Tryggvason, 2009; Gloth et al., 2003) advects the marked interface from an initial configuration and keeps the topology of the interface constant during the simulation. Therefore, this method is limited to topological changes in multi-phase fluids, such as merging or breaking of droplets.

The LSM was introduced by Osher (1988) in 1988. The material boundary or interface is defined as the zero set (Osher and Fedkiw, 2001, 2003; Sethian, 1985) of the given scalar field. Sethian (2003) and Lakehal (2002) applied the concept to fluid simulation. In 2002, Enright et al. (2002) combined Lagrangian marker particles with LSM to obtain and maintain a smooth geometrical description of the fluid interface. However, it has been pointed out by Müller (2009) and Garimella et al. (2005) that material volume is not well-preserved in the level set method, which is a main drawback of this approach.

The VOF method (Hirt and Nichols, 1981) is one of the best established interface volume tracking methods currently in use (Marek et al., 2008; Sussman and Puckett, 2000). It employs the idea of using mass conservation for the volume of each fluid. Apart from VOF, other interface tracking algorithms include *simple line interface* (SLIC) (Noh and Woodward, 1976) and piecewise linear interface construction (PLIC) (Rider and Kothe, 1998). In the SLIC method, the interface is taken to be perpendicular or parallel to coordinate axes directions, while in PLIC, the interface is given by a piecewise linear function with arbitrary orientation.

However, the interface tracking algorithms mentioned above do not fully apply to liquid–liquid extraction simulation as one phase of fluids is fully dispersed and no continuous interface exists between the two fluids. Therefore, a material interface is not traceable in the case of slurry flow. Instead, topology-based techniques for the analysis of level sets are more suitable and useful in this context because of their ability to capture the absolute and relative behavior of small-scale features like dispersed bubbles directly (Tierny et al., 2009).

The topology of 2D scalar fields can be studied via *contour trees* introduced by de Berg and van Kreveld (1997). Carr et al. (2000) extended this approach by presenting a simple and robust algorithm for the computation of contour trees. In three or more dimensions, a Reeb graph (Reeb, 1946) is more efficient as it automatically encodes the behavior of level sets defined on an arbitrary manifold (Tierny et al., 2009). Applications of the Reeb graph can be found in fast isosurface extraction (Carr et al., 2004) and feature tracking (Weber et al., 2009).

Bremer et al. (2010) presented a hierarchical segmentation strategy, allowing feature tracking of time-dependent isosurfaces. Isotherms extraction and vertex classification at original time steps are carried out on discrete time slices with the structured spatial discretization. Unlike Weber et al. (2009), Bremer et al. (2010), the extraction approach presented in this paper is aimed for dynamic particle-based data sets, whose spatial structure randomly changes over time.

Particle-based fluid simulation is a particular class of numerical algorithms for simulating fluids. Compared to traditional mesh-based simulation algorithms, particle-based methods are capable of handling complex flow problems where geometry changes dramatically (Drumm et al., 2008; Premoze et al., 2003). Smooth particle hydrodynamics (SPH) as the first particle method was introduced by Lucy (1977) in 1977 to simulate astrophysical problems. It has later been applied to simulating flow problems (Monaghan, 1988; Premoze et al., 2003) including free surfaces, multi-fluid interfaces, turbulent flow and incompressible fluids, where deficiencies of grid-based methods occur.

## 2. Reconstruction of the time-varying density field

Given a two-dimensional time-varying density field, we need to provide an appropriate interpolation scheme for obtaining well corresponding dense snapshots in time in order to be able to perform accurate topology extraction and tracking of phase movement. To be able to use concepts from three-dimensional visualization and convey an instantaneous representation of the unsteady field, we explore the 2D time-dependent density field as stationary three-dimensional field by using time as third dimension. We examine a numerical data set which is the result of a finite point set method (FPM) simulation for two-phase fluid. It consists of 7 time steps. The simulation technique advects a highly adaptive set of fluid particles than constitutes an implicit computational grid. Due to particle creation and destruction, there is no correspondence between the particles from one time step to another. Each of these computational particle carries the information of fluid density $\rho$ and fluid velocity $\vec{v}$.

The reconstruction of a smooth time-dependent density field requires interpolation along both spatial and temporal axes. Our method is a hybrid of a moving lease squares approach for spatial interpolation as well as a Lagrangian-based scheme for temporal interpolation derived from the FPM method. By considering the underlying equations of flow motion and applying Lagrangian advection to the computational particles, we obtain a physically meaningful interpolation of the scalar field over time. In the following subsections, we will describe our scheme in detail.

### 2.1. Time interpolation

Due to the fact that the given numerical data has few time slices (7–10 available), we propose the following algorithm to incrementally interpolate between two neighboring time steps $T_0$ and $T_1$ to obtain $N$ intermediate steps, thus increasing resolution along the time axis significantly. The general idea of our time interpolation consists of three steps: we first re-initialize and re-sample a point cloud at the first intermediate time frame, then we update the density field on the new point cloud, finally we use the underlying physical model to re-compute velocities at each re-sampled point. Given flow fields $f$, $\vec{v}$ at time $T_0$ and $T_1$, we employ an iterative technique to interpolate and re-compute $f$, $\vec{v}$ at intermediate time steps $T_0 + \Delta, \ldots, T_0 + N\Delta$, which is described by the following pseudocode.

**Algorithm 1** Interpolation scheme

**Input:** $(x_i, \rho, \vec{v})_{T_0}$ and $(x_i, \rho, \vec{v})_{T_1}$
**Output:** $(x_i, \rho, \vec{v})_{T_0 + j\Delta}$, for $0 < j \leqslant N$.

**for** $j = 1 \to N$ **do**
  $T^* \leftarrow T_0 + j\Delta$
  forward advect particles $x_i$ from $T_0 + (j-1)\Delta$ to $T^*$,
  backward advect particles $x_i$ from $T_1$ to $T^*$,
  re-calculate velocities $(\vec{v}_i)_{T^*}$ at advected points using equation (2).
  re-sample particles at time $T$.
  **for all** $x_i$ **do**
    update density values $(\rho_i)_{T^*}$ and velocities $v_i^{T^*}$ at re-sampled points using equation (1).
  **end for**
  $(x_i, \rho, \vec{v})_{T_{j\Delta}} \leftarrow (x_i, \rho, \vec{v})_{T*}$
**end for**

In words, for the first interpolation point $T^* = T_0 + \Delta$, we advect the particles from $T_0$ to $T^*$ forward along the velocity field $v_0$ (see Fig. 1) and advect the particles backward from $T_1$ to $T^*$. The corresponding advection equations yield:

$$(x)T^* = (x)^{T_0} + \vec{v}T_0\Delta$$
$$(x)T^* = (x)^{T_1} - \vec{v}T_1(T_1 - T_0 - \Delta)$$

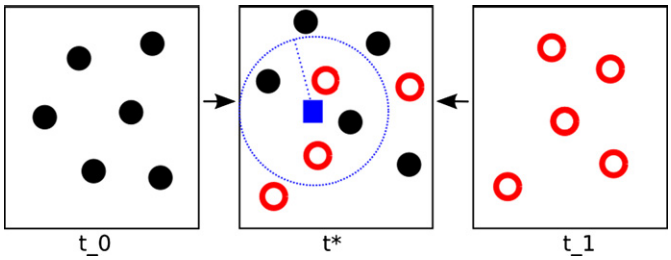



**Fig. 1.** Proposed interpolation scheme: Forward advection of fluid particles from previous time step to intermediate stage $t^*$, and re-calculation of the velocity field. Backward advection of fluid particles from the last time step, re-calculation of the velocity field. Data is re-sampled at intermediate time $t^*$ at blue points using a radial basis function. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

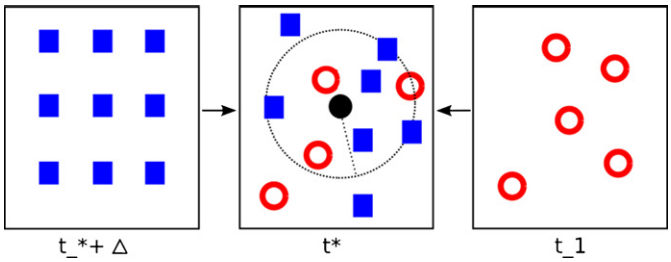



**Fig. 2.** Left: the first inserted time frame with values re-sampled at a regular point. The second inserted frame was created by advecting particles from $T_0 + \Delta$ and $T_1$. Then we re-sample the points at the same grid as used in $T_0 + \Delta$.

Therefore, a new set of particles is formed. We re-sample these points onto a regular grid and approximate the density function using Shepard interpolation (Shepard, 1968):

$$\rho(x) = \frac{\sum_{i=1}^{m} w_i \rho_i}{w_i} \tag{1}$$

Subsequently, we re-compute the velocity vectors on the re-sampled points using a numerical estimation derived from SPM method (Tiwari and Kuhnert, 2007). In computational fluid dynamics, flow properties are commonly described under two different reference frames – the Euler fixed reference frame and the Lagrangian co-moving reference frame. Compared to Euler type of conservation laws, Lagrangian specifications employ the idea of co-moving computational frames along the flow fields. Changes of flow properties are described on the moving computational particle, named *material derivatives*:

$$\frac{Df}{Dt} = \frac{\partial f}{\partial t} + \vec{v} \cdot \nabla f$$

In the proposed algorithm, fluid velocities are re-computed at advected points. Therefore, we consider the momentum equation (Tiwari and Kuhnert, 2007) in the material derivative form. Thus, a Lagrangian formulation of the momentum equation yields

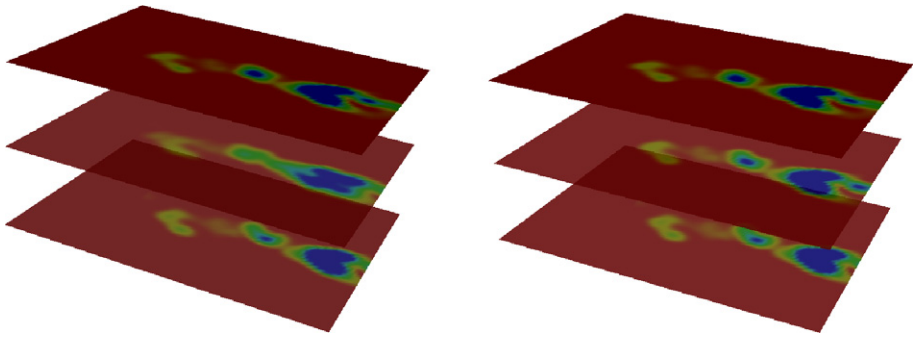
$$\frac{D\vec{v}}{Dt} = -\frac{1}{\rho} + \frac{1}{\rho}\nabla \cdot (2\mu D) + \frac{1}{\rho}\vec{F}_s + \vec{g} \tag{2}$$

where $\mu$ is the fluid viscosity and it fulfills the mass conservation law. We use a continuous surface force (CSF) model (Brackbill et al., 1992) to define the surface tension force $\vec{F}_s = \sigma\kappa\vec{n}\delta_s$, with interface normal vector $\vec{n}$, curvature $\kappa$ and normalized surface delta function $\delta_s$. Values for $\vec{n}$, $\kappa$, $\delta_s$ can be obtained by introducing marking functions for fluid type 1 and 2. For details please refer to Brackbill et al. (1992), Tiwari and Kuhnert (2007). The velocities at the re-sampled points on intermediate time step $T^*$ are computed component-wisely by solving the following equation using Chorin's projection method (Chorin, 1968):

$$\vec{v} - \frac{dt}{\rho}\nabla\mu \cdot \nabla\vec{v} - dt\,\frac{\mu}{\rho}\Delta\vec{v} = \vec{v}^n + dt\left(\frac{F_S^n}{\rho} + g + \frac{\nabla\mu}{\rho} \cdot \nabla\left(\vec{v}^n\right)\right) \tag{3}$$

After re-sampling points onto $T^*$ and re-evaluating the velocity field, we then use $T^*$ as the new $T^0$ and repeat computations to interpolate the flow fields at $T_0 + 2\Delta$. (See Fig. 2.)

Two main aspects distinguish our method from other existing interpolation schemes – re-calculation of the velocity field on the re-sampled points using a physical model, and iterative advection of the re-sampled points to the next interpolated time frame using the re-computed velocity. While re-sampling the points and interpolating the density fields, it is important



(a) Left: slice in the middle is a linear blending of the top and bottom ones.

(b) Right: slice in the middle is interpolated by the proposed method.

**Fig. 3.** Linear and nonlinear interpolation.

to appropriately reconstruct the velocity field at intermediate time frames. To obtain a physically meaningful interpolation, we use the velocity updating schemes proposed in the SPM method. By solving the original momentum equation (2) from the simulation model, we obtain a good interpolation by reconstructing the missing time steps.

The additional computational effort for re-calculating the velocities is justified by the quality of the results, as demonstrated in the following example. We compare our method with a linear interpolation scheme without velocity re-computation. A naive way of interpolating the density field between two time steps is a linear combination between the two, such as:

$$f(t_0 + \Delta t) = \frac{t_0 - t_1 + \Delta t}{t_0 - t_1} f(t_0) + \frac{-\Delta t}{t_0 - t_1} f(t_1)$$

However, a linear interpolation yields physically incorrect results as it fails to take flow direction into account, making way for mistakes in tracking or topology extraction. As one can see in Fig. 3a, the middle layer is a linear interpolation of the two slices. With our proposed velocity re-calculation, the interpolation provides a physical meaningful transaction of fluid properties from one time step to another (Fig. 3b).

### 2.2. Spatial approximation

At each given time step $T_i$, not only the positions of point clouds are varying, but also the number of points. In order to achieve a continuous representation of the field that can be re-sampled by a uniform distribution of spatial points, we perform a moving lease squares approximation for fitting the velocity and density values.

The moving lease squares (MLS) method, being one of the most popular mesh-free interpolation schemes, was introduced by Lancaster and Salkauskas (1981) in 1981 to derive a smooth surface from a set of scattered data points in space. To interpolate flow quantities at a given point of evaluation, we construct a weighted least squares fitting considering a local set of neighboring points. A global approximation is obtained by moving this point of evaluation across the whole domain. Differentiability of the global fitting function is ensured by choosing a continuously differentiable weight function (also known as kernel functions), see Levin (1998). For a finite set of points $\mathbf{x}_i$, we approximate the given scalar value $f_i$ by a polynomial $f(\mathbf{x})$ at $\mathbf{x} = (x, y, z)$ in the sense of minimizing a weighted square error:

$$G(x) = \min \sum_i w(r_i) \left\| f(\mathbf{x}_i) - f_i \right\|^2 \tag{4}$$

where $f$ is an $n$ degree polynomial in $m$-dimensional space and weight $w$ is a scaling function depending on distance $r_i = \|\mathbf{x} - \mathbf{x}_i\|$. For instance, a quadratic approximation for two-dimensional scattered data can be written as

$$f(\mathbf{x}) = \mathbf{b}(\mathbf{x}) \cdot \mathbf{c} = \begin{bmatrix} 1 & x & y & x^2 & xy & y^2 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \end{bmatrix}$$

To address the problem of varying point densities in different time steps, the spatial interpolation can be controlled by adjusting the size of the neighborhood radius $r$. We will specify our choice for this parameter later. First, let us take a look at how the minimization problem can be solved.

The minimization problem (4) can be solved by setting partial derivatives with respect to $c_i$ to zero, $\frac{\partial G}{\partial c_i} = 0$.

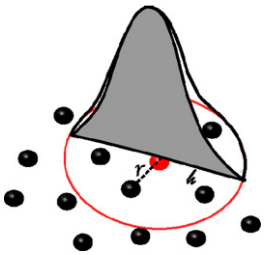



**Fig. 4.** Neighborhood structure of a given point inside the point cloud.

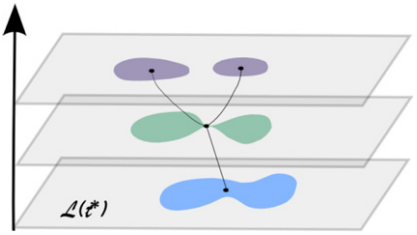



**Fig. 5.** Level sets of a function and its' Reeb graph.

As a result, coefficients $c_i$ can be obtained by solving the following linear system

$$\sum_i w(r_i)\mathbf{b}(\mathbf{x}_i)\mathbf{b}(\mathbf{x}_i)^T\mathbf{c} = \sum_i w(r_i)\mathbf{b}(\mathbf{x}_i)f_i \tag{5}$$

and solution of the coefficients $c_i$ yields

$$\mathbf{c} = \left[\sum_i w(r_i)\mathbf{b}(\mathbf{x}_i)\mathbf{b}(\mathbf{x}_i)^T\right]^{-1}\sum_i w(r_i)\mathbf{b}(\mathbf{x}_i)f_i \tag{6}$$

The idea of using a weight function is that points residing closer to the point $\mathbf{x}$ should have a higher influence than the ones that are far away, see Fig. 4.

Intuitive choices for weight functions are therefore radial basis functions which depend on the inverse distance. In mesh-free/particle-based numerical simulations, such as smoothed particle hydrodynamics (Bonet and Kulasegaram, 2002), cubic splines and Gaussian functions are commonly used, see Drumm et al. (2008), Kuhnert and Tiwari (2003), Tiwari and Kuhnert (2007).

To define a good weight function, one needs to restrict the number of points by imposing a small compact support on the weight function, such that points outside the compact support will have zero influence. Moreover, a minimal number of points should be guaranteed in order to have a non-singular system of equations for a second order approximation (Drumm et al., 2008).

To be consistent with the FPM simulation data, we choose a Gaussian weight function. The size of $h$ is chosen such that there are at least 6 points within the circle of radius $h$:

$$w(\mathbf{x}-\mathbf{x}_i,h) = \begin{cases} \exp(-2\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h^2}), & \text{if } \frac{\|\mathbf{x}-\mathbf{x}_i\|}{h} \leqslant 1 \\ 0, & \text{elsewhere} \end{cases}$$

To approximate vector fields of the flow, for instance velocity, we apply the moving lease squares method to each component of the vector. In our implementation, we use a quadratic basis function for $b(\mathbf{x}_i)$. Only when there are not enough points in the neighborhood, the basis function is reduced to a linear function.

Previously, we obtained a continuous density field in spatial directions using MLS and in time direction using a Lagrangian method. In the next section, we discuss the use of Reeb graphs in analyzing this interpolated density field.

## 3. Reeb graph

After constructing a continuous density function over time, we are able to extract a spatio-temporal isosurface in all three dimensions at any given density value. We perform Reeb graph extraction to analyze the topology change of these surfaces, using time as a Morse function.

Recall the definition of the Reeb graph (see Reeb, 1946; Tierny et al., 2009 for more details): A real-valued function $f : M \to \mathscr{R}$ is defined on a manifold $M$. The Reeb graph is a graph which describes the topology change of $f$'s level sets. For a given value $t^*$, the level set $L(t^*)$ (Fig. 5) is defined as the inverse image of $t^*$ onto $M$ through $f$:

$$L(t^*) = f^{-1}(t^*)$$

Each connected component of the level set $L(t^*)$ is called a contour, see Fig. 5. In Morse theory (Milnor, 1963), topology change of contours occurs only at the critical points of $f$ (Patan et al., 2009). A Reeb graph is a graph whose nodes correspond to the critical points of $f$ and the edges represent the connectivity of the contours of $f$ while $t^*$ evolves continuously in $\mathscr{R}$, see Fig. 5.

Taking a double torus as an example (see Fig. 6a), let the height field $z$ be our scalar function, connected level sets $L(z)$ of $z$ are contours at a certain height value $z = z^*$ (shown in Fig. 6b). There are four types of nodes on a Reeb graph which correspond to four different events of contour evolution.

(a) Torus Manifold.

(b) Level sets of the height field. Each color of ribbons indicate one level set of the height field

(c) Reeb graph of the height function shows the dynamics of the level sets

**Fig. 6.** A double torus example of Reeb graph. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

*Birth node.* A birth node in a Reeb graph corresponds to a minimum critical point of $f$. On the torus example, this is the point where bottom purple node lies, see Fig. 6c. At the position of the birth node, a contour of scalar function $z$ emerges for the first time, hence the name.

*Bifurcation node.* The blue node on the torus example is a bifurcation node. As the name implies, it is a node where a contour of the given scalar function splits. In mathematical terms, a bifurcation node is one type of saddle points, whose second derivative degenerates.

*Merging node.* Compared to the bifurcation case, the situation where two contours merge into one is characterized by a merging node. (See second node from the top in Fig. 6c.) It can be seen as symmetric to the bifurcation node with respect to time-reversal (inverting the $z$ direction). Therefore, merging and bifurcation nodes are topologically equivalent and both correspond to a saddle point on the manifold $\mathscr{M}$.

*Death node.* The pink node at the very top in Fig. 6c is a death node on the torus. At this point, the contour of $z$ vanishes, hence the name death node. A death node corresponds to a maximum critical point of the scalar function $z$. And similar to the bifurcation/merging pair, death nodes can be considered symmetric to birth nodes.

*Extraction of the high-density area and computation of the Reeb graph.* Since we are interested in the regions where the density of the solvent is above a certain threshold $\varepsilon$ (the well-mixed regions with high concentrations of the solvent phase), we extract an implicit surface $f(\mathbf{x}) = \varepsilon$, enclosing regions with higher solvent density. Each segment of the surface (which is perpendicular to time axis) corresponds to the original given 2D slices or the interpolated intermediate slices. Level sets of the implicit surface therefore yield isocontours on the original 2D data. For a straightforward interpretation, our surface can be interpreted as a sweeping of the 2D isocontours.

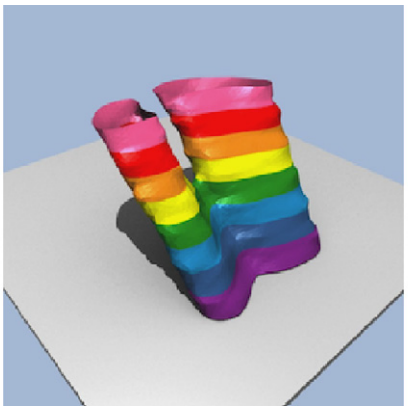
Using a standard Marching Cubes algorithm (Lorensen and Cline, 1987), we extract the time surface from the interpolated density function discussed in the previous chapter. Results are shown in Figs. 7, 8 and 9. A tracking Reeb graph is computed for this time surface using time as the Morse function (similar to the height field in the torus example). We used a vtkReebGraph filter based on Pascucci's streaming algorithm (Pascucci and Cole-McLaughlin, 2003; Pascucci et al., 2007). Note that since the Reeb graph algorithm operates on a closed surface, we introduce ghost cells at the first and last time slices as well as the ingoing and outgoing flow boundaries.

## 4. Results

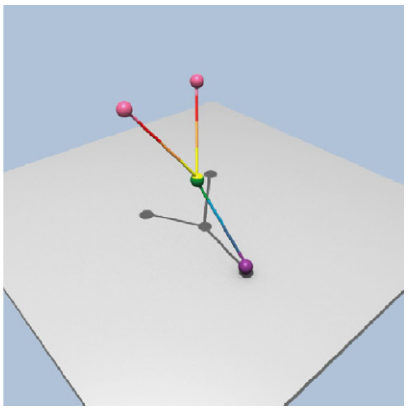
Figs. 7a, 8a and 9a are the extracted isosurfaces (swept level sets over time). Corresponding Reeb graphs are shown in Figs. 7c, 8c and 9c. Figs. 7b, 8b and 9b are colored level sets of time, which provide an intuitive feeling of how the flow propagates over time. Node coordinates $(x, y, t)$ indicate that one of the four types of contour evolution events takes place at point $(x, y)$ at time $t$.

Moreover, the edge connectivity of the Reeb graph defines the type of nodes and gives the connectivity of contours. We discuss the result using the following three examples. The first example (Fig. 7) is a typical case where the high-density flow regions split up into two. Shown in Fig. 7c, a bifurcation event occurs between the yellow and green level sets, which

(a) Bifurcation: regions splittings after a certain time point. Outer normal of the plane points to time direction

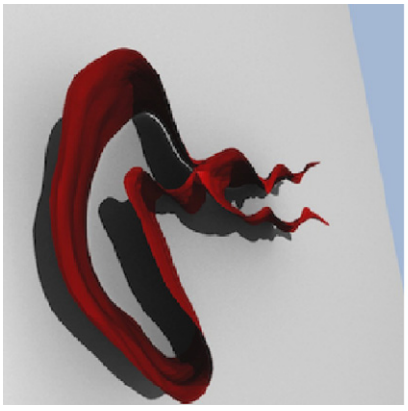


(b) Level sets of time.



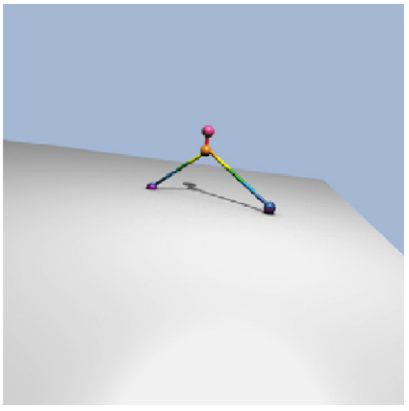
(c) Extracted Reeb graph of the bifurcation case.

**Fig. 7.** Bifurcation example. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)



(a) Merging: regions merge into a large one after a certain time point. Outer normal of the plane points to time direction

(b) Level sets of time.



(c) Extracted Reeb graph of the merging case.

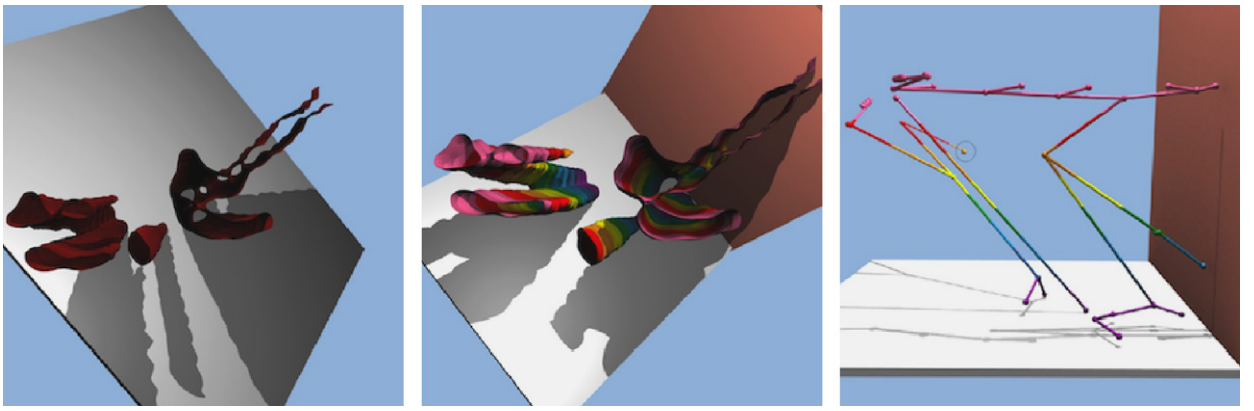
**Fig. 8.** Example of merging regions.

points to a contour splitting event in Fig. 7b. As opposed to bifurcation events, another example (Fig. 8) shows the case where nodes meet and merge to a larger one.

In the classification of Reeb graph nodes, birth and death nodes both have multiple meanings depending on the position of the node: those which are on the ingoing boundary of the flow mean that a new contour is entering the domain because of the feed-in fluid; similarly, for death nodes, those that are on the outgoing boundary means flow is exiting the area while the ones inside $xy$-plane signify the event that fluid blobs smear out and vanish.

A complete scenario is presented in Fig. 9. The white plane at the bottom corresponds to the $xy$-plane. Its outer normal which points up indicates the time direction. Flow is coming in from the side of the red plane, and exists freely at the left side. This example is generated based on the given ten time slices and one intermediate step of interpolation between each of the slices. All four types of nodes occur in this scenario, for instance, the circled orange node is a birth node which signifies the increase of the density of the second fluid phase in the region. In comparison to a direct coloring on level sets (Fig. 9b), the Reeb graph on the right-hand side (Fig. 9c) serves as a better illustration tool of the topological skeleton of the surface. Moreover, we also provide a comparison between the conventional graph layout (Fig. 10) and our 3D Reeb graph representation (Fig. 9c). While both representations contain exactly the same information, the 3D representation has an obvious advantage in conveying the topological evolution.

Furthermore, the visualization technique presented in this paper allows visual tracking of fluid bubbles in a stationary view. It also enables further analysis and estimation of solvent efficiency, such as to find the regions where no solvent mixing occurred.



(a) Extracted surface for the whole domain over all time steps.

(b) Level sets of time. Colored wall is the incoming boundary of the flow. Outer normal of the white plane points to time direction

(c) Extracted Reeb graph of the left image.

**Fig. 9.** A complete scenario. White plane denotes the spatial domain. Flow is entering the domain from red wall. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

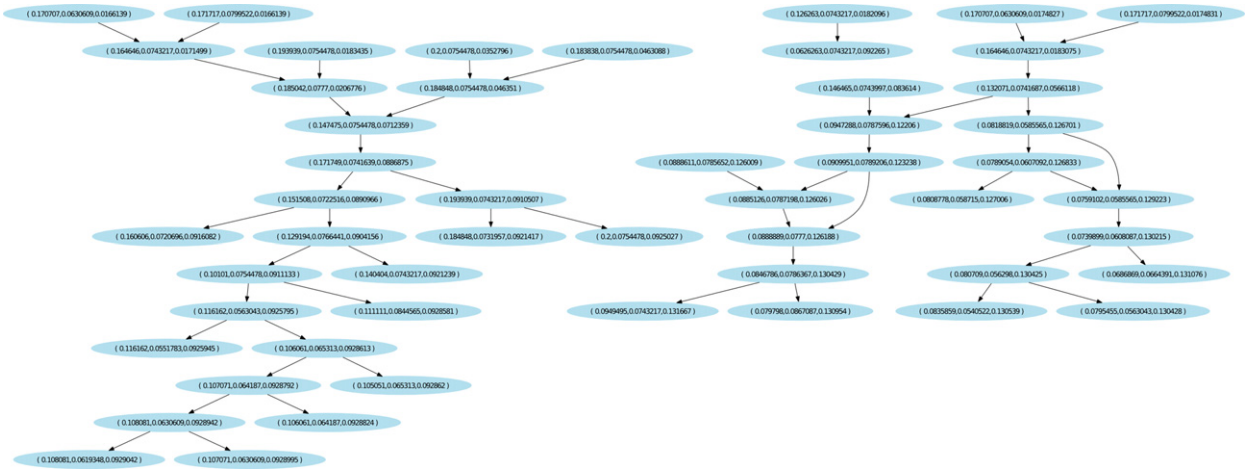


**Fig. 10.** Conventional graph layout. Spatial and time coordinates are printed inside the nodes. This representation is convoluted and hard to understand. In contrast, our 3D Reeb graph representation gives an intuitive impression of how topology is changing.

## 5. Conclusion and future research possibilities

We have presented a method to characterize the evolution of time-dependent density fields by using the topology-based Reeb graphs. The method we proposed can be utilized by mechanical engineers to evaluate the design of mixing devices and the choice of inflow velocities to achieve optimal mixing results. The challenge in our data set is the low time resolution which could result in low tracking accuracy and ill defined topology. To improve the quality of our topology tracking, we have proposed to use Lagrangian advection on the point set utilizing extra velocity information to obtain a physical-based interpolation of density field.

Possible future work could extend our approach to a 3D simulation of multi-fluid data, which implies computing a Reeb graph of a scalar function $t$ over a 4D manifold. Techniques such as tracking the Jacobi set of a time-dependent scalar field could also be a good choice for representing the topology change of the density field. Moreover, further work can be done to evaluate how the time resolution of input data influences the extracted topology. Apparently, the topology of the extracted surface would change when time steps are large. Finding a threshold for time stepping length (below which the extracted Reeb graph does not change) would probably be related to the maximum gradient of the velocity field and Courant–Friedrichs–Lewy (CFL) conditions of the numerical scheme.

## References


Bonet, J., Kulasegaram, S., 2002. Alternative total Lagrangian formulations for corrected smooth particle hydrodynamics (csph) methods in large strain dynamics problems. European Review of Finite Elements 11, 893–912.

Brackbill, J.U., Kothe, D.B., Zemach, C., 1992. A continuum method for modeling surface tension. Journal of Computational Physics 100, 335–354.

Bremer, P.-T., Weber, G., Pascucci, V., Day, M., Bell, J., 2010. Analyzing and tracking burning structures in lean premixed hydrogen flames. IEEE Transactions on Visualization and Computer Graphics 16, 248–260.

Carr, H., Snoeyink, J., Axen, U., 2000. Computing contour trees in all dimensions. In: SODA'00: Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 918–926.

Carr, H., Snoeyink, J., van de Panne, M., 2004. Simplifying flexible isosurfaces using local geometric measures. In: VIS'04: Proceedings of the Conference on Visualization. IEEE Computer Society, Washington, DC, USA, pp. 497–504.

Chorin, A., 1968. Numerical solution of the Navier–Stokes equations. Mathematics of Computation 22, 745–762.

de Berg, M., van Kreveld, M.J., 1997. Trekking in the Alps without freezing or getting tired. Algorithmica 18 (3), 306–323.

Drumm, C., Tiwari, S., Kuhnert, J., Bart, H.-J., 2008. Finite pointset method for simulation of the liquid–liquid flow field in an extractor. Computers and Chemical Engineering 32 (12), 2946–2957.

Enright, D., Fedkiw, R.P., Ferziger, J.H., Mitchell, I., 2002. A hybrid particle level set method for improved interface capturing. Journal of Computational Physics 183, 83–116.

Garimella, R.V., Dyadechko, V., et al., 2005. Interface reconstruction in multi-fluid, multi-phase flow simulations. In: IMR'05. Springer, pp. 19–32.

Gloth, O., Hänel, D., Tran, L., Vilsmeier, R., 2003. A front tracking method on unstructured grids. Computers & Fluids 32 (4), 547–570.

Hirt, C.W., Nichols, B.D., 1981. Volume of fluid (vof) method for the dynamics of free boundaries. Journal of Computational Physics 39 (1), 201–225.

Kuhnert, J., Tiwari, S., 2003. Finite pointset method based on the projection method for simulations of the incompressible Navier–Stokes equations. In: LNCSE: Meshfree Methods for Partial Differential Equations, vol. 26. Springer, Berlin, pp. 373–388. Chapter 1.

Lakehal, D., Meier, M., Fulgosi, M., 2002. Interface tracking towards the direct simulation of heat and mass transfer in multiphase flows. International Journal of Heat and Fluid Flow 23, 242–257.

Lancaster, P., Salkauskas, K., 1981. Surfaces generated by moving least squares methods. Mathematics of Computation 37 (155), 141–158.

Levin, D., 1998. The approximation power of moving least-squares. Mathematics of Computation 67, 1517–1531.

Lorensen, W.E., Cline, H.E., 1987. Marching cubes: A high resolution 3d surface construction algorithm. SIGGRAPH Comput. Graph. 21, 163–169.

Lucy, L.B., 1977. A numerical approach to the testing of the fission hypothesis. The Astronomical Journal 82, 1013–1024.

Marek, M., Aniszewski, W., Boguslawski, A., 2008. Simplified volume of fluid method for two-phase flows. Task Quarterly 3, 255–265.

Milnor, J., 1963. Morse Theory. Annals of Mathematical Studies, vol. 51. pp. 847–849.

Monaghan, J.J., 1988. An introduction to sph. Computer Physics Communications 48 (1), 89–96.

Müller, M., 2009. Fast and robust tracking of fluid surfaces. In: SCA'09: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. ACM, New York, NY, USA, pp. 237–245.

Noh, W., Woodward, P., 1976. Simple line interface method. In: van de Vooren, A., Zandbergen, P. (Eds.), Proceedings of the 5th International Conference on Fluid Dynamics. In: Lecture Notes in Physics, vol. 59, pp. 330–340.

Osher, S., Fedkiw, R.P., 2001. Level set methods: An overview and some recent results. Journal of Computational Physics 169 (2), 463–502.

Osher, S.J., Fedkiw, R.P., 2003. Level Set Methods and Dynamic Implicit Surfaces. Springer.

Osher, S., Sethian, J.A., 1988. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations. Journal of Computational Physics 79, 12–49.

Pascucci, V., Cole-McLaughlin, K., 2003. Parallel computation of the topology of level sets. Algorithmica 38 (1), 249–268.

Pascucci, V., Scorzelli, G., Bremer, P.-T., Mascarenhas, A., 2007. Robust on-line computation of Reeb graphs: Simplicity and speed. In: SIGGRAPH'07: ACM SIGGRAPH 2007 Papers. ACM, New York, NY, USA. 58 pp.

Patan, G., Spagnuolo, M., Falcidieno, B., 2009. A minimal contouring approach to the computation of the Reeb graph. IEEE Transactions on Visualization and Computer Graphics 15 (4), 583–595.

Premoze, S., Tasdizen, T., Bigler, J., Lefohn, A., Whitaker, R.T., 2003. Particle-based simulation of fluids. Proceedings of Eurographics 22, 401–410.

Reeb, G., 1946. Sur les points singuliers d'une forme de Pfaff complétement intégrable ou d'une fonction numérique. Comptes Rendus de l'Académie des Sciences 222, 847–849.

Rider, W.J., Kothe, D.B., 1998. Reconstructing volume tracking. Journal of Computational Physics 141 (2).

Sethian, J.A., 1985. Curvature and the evolution of fronts. Commun. Math. Phys. 101, 487–499.

Sethian, J.A., Smereka, P., 2003. Level set methods for fluid interfaces. Annu. Rev. Fluid Mech. 35, 341–372.

Shepard, D., 1968. A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM National Conference, ACM'68. ACM, New York, NY, USA, pp. 517–524.

Sussman, M., Puckett, E.G., 2000. A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows. Journal of Computational Physics 162, 301–337.

Terashima, H., Tryggvason, G., 2009. A front-tracking/ghost-fluid method for fluid interfaces in compressible flows. Journal of Computational Physics 228 (11), 4012–4037.

Tierny, J., Gyulassy, A., Simon, E., Pascucci, V., 2009. Loop surgery for volumetric meshes: Reeb graphs reduced to contour trees. IEEE Transactions on Visualization and Computer Graphics 15 (6), 1177–1184.

Tiwari, S., Kuhnert, J., 2007. Modeling of two-phase flows with surface tension by finite pointset method (fpm). J. Comput. Appl. Math. 203 (2), 376–386.

Unverdi, S.O., Tryggvason, G., 1992. A front tracking method for viscous incompressible flows. Journal of Computational Physics 100.

Weber, G., Bremer, P.-T., Day, M., Bell, J., Pascucci, V., 2009. Feature tracking using Reeb graphs. In: DOE Computer Graphics Forum, Monterey, CA.