Visualization of Time-Dependent Foam Simulation Data

Dan R. Lipşa

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Doctor of Philosophy

Swansea University
Prifysgol Abertawe

Department of Computer Science
Swansea University

2013

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ..... (candidate)

Date .....

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed ..... (candidate)

Date .....

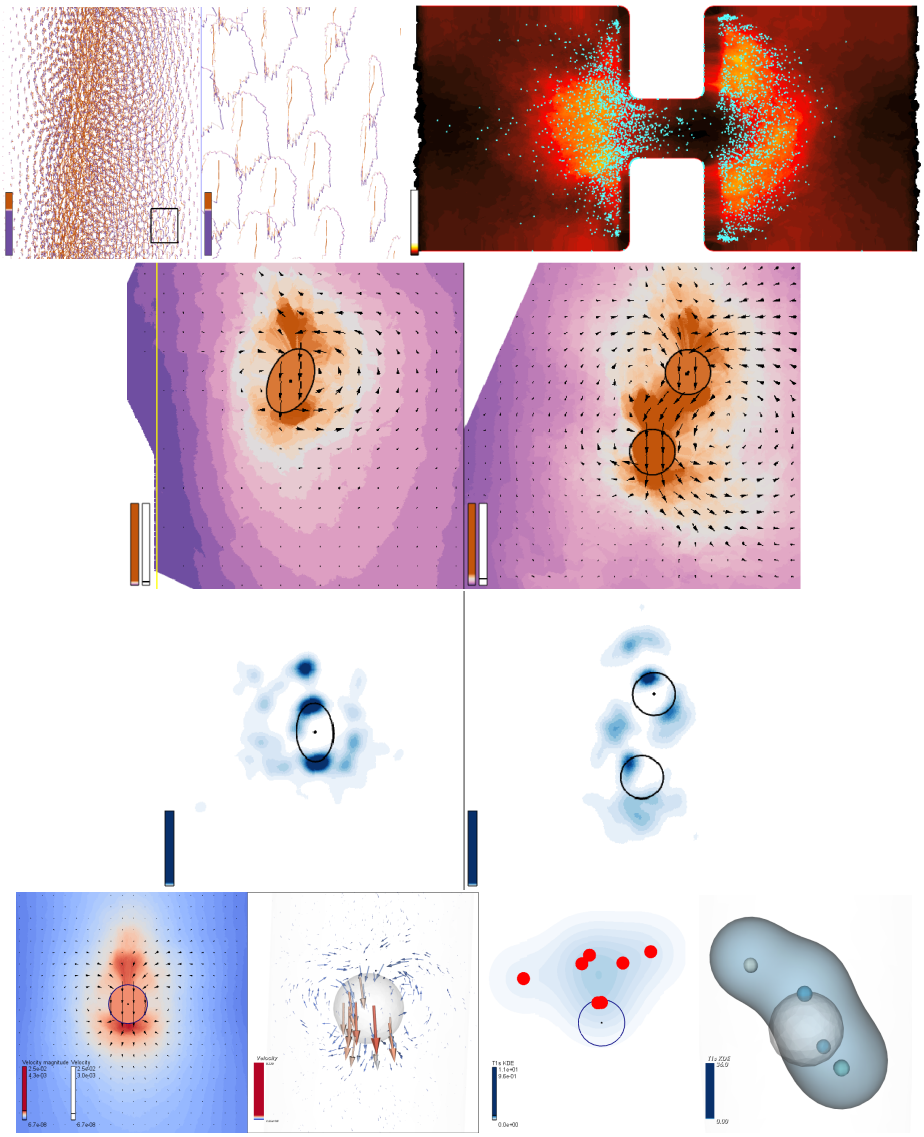
Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate)

Date .....

Visual Abstract



Abstract

Research in the field of complex fluids such as polymer solutions, particulate suspensions and foams studies how the flow of fluids with different material parameters changes as a result of various constraints. Surface Evolver, the standard solver software used to generate foam simulations, provides large, complex, time-dependent data sets with hundreds or thousands of individual bubbles and thousands of time steps. However this software has limited visualization capabilities, and no foam specific visualization software exists. We describe the foam research application area where, we believe, visualization has an important role to play. We present a novel application, called FoamVis, that provides various techniques for visualization, exploration and analysis of foam simulation data. FoamVis can visualize individual time steps or produce time-dependent visualizations. It can process a simulation or it can facilitate comparison of several related simulations. We show new features in foam simulation data and new insights into foam behavior discovered using our application. Based on the many research questions that domain experts are able to address, we believe we provide a valuable tool for visualization and analysis of data in the foam research community.

Related Publications

This thesis is based on the following publications:

- D. R. Lipsa, R. S. Laramee, S. J. Cox, J. C. Roberts, and R. Walker, "Visualization for the Physical Sciences," in Eurographics, (Llandudno, Wales, UK), pp. 49-73, Apr. 2011. State-of-the-Art Reports. [LLC+11]

- D. R. Lipsa, R. S. Laramee, S. J. Cox, J. C. Roberts, R. Walker, M. A. Borkin, and H. Pfister, "Visualization for the Physical Sciences," EG Computer Graphics Forum, vol. 31, pp. 2317-2347, Dec. 2012. [LLC+12]

- D. R. Lipsa, R. S. Laramee, S. J. Cox, and I. T. Davies, "FoamVis: Visualization of 2D Foam Simulation Data," Visualization and Computer Graphics, IEEE Transactions on, vol. 17, pp. 2096-2105, Oct. 2011. [LLCD11]

- S. Cox, D. Lipsa, I. Davies, and R. Laramee, "Visualizing the dynamics of two-dimensional foams with FoamVis," Colloids and Surfaces A: Physicochemical and Engineering Aspects, 2013. In press. [CLDL13]

- D. R. Lipsa, R. S. Laramee, S. J. Cox, and I. T. Davies, "Comparative Visualization and Analysis of Time-Dependent, 2D Foam Simulation Data," tech. rep., Swansea University, 2013. [LLCD13c]

- D. R. Lipsa, R. S. Laramee, S. Cox, and I. T. Davies, "Visualizing 3D Time-Dependent Foam Simulation Data," in Lecture Notes in Computer Science, International Symposium on Visual Computing (ISVC), (Rethymnon, Crete, Greece), July 2013. [LLCD13a]

- D. R. Lipsa and R. S. Laramee, "FoamVis, A Visualization System for Foam Research: Design and Implementation," tech. rep., Swansea University, 2013. [LL13]

- D. R. Lipsa, R. S. Laramee, S. J. Cox, and I. T. Davies, "A Visualization Tool For Foam Research," in NAFEMS World Congress (NWC) Conference Proceedings, (Salzburg, Austria), p. 141, June 2013. [LLCD13b]

Acknowledgements

I would like to thank the Research Institute of Visual Computing (RIVIC) for sponsoring my research; Dan Bergeron and Ted Sparr for advising my first work in visualization; Simon Cox and Tudur Davis, for teaching me about foam research and providing access and support with their work; my adviser Bob Laramee for his knowledge, humor, friendship, and freedom to pursue my ideas; Anca for encouragement and love, and Victor and Paul for being great kids.

Table of Contents

Chapter 1

Introduction

> "Visualize this thing that you want, see it, feel it, believe in it. Make your mental blue print, and begin to build."
> – Robert Collier (1885-1950)[1]

Modern society is confronted with a data explosion. Wide availability and improved performance of sensors enable the acquisition of ever increasing amounts of data. Examples include medical, weather, engineering and video surveillance data. Increased computation and storage abilities of computers enable performing large simulations that generate terabytes or petabytes of data. That data needs to be stored, explored and analyzed. The Internet, social media, and the increasing number of smart phones allow users to generate vast amounts of data, such as video, pictures and communication, that have to be managed and can be mined for information. The Economist refers to this phenomenon as the *Big Data* [Eco10]. Understanding and extracting value from *Big Data* is thought to be the next frontier for innovation, competition, and productivity [Ins11] in the global economy.

1.1 Visualization

Visualization is an application area of computer science, where computer generated images are used to represent data. The goal of visualization is to help people understand or gain new insights into their data. Visualization was born because customers in science, aerospace and the biomedical field needed a way to process and understand the large amounts of data they were generating [Lor04]. Computer graphics, a related field, that studies the creation and manipulation of visual content, focuses more on creating reality for the animation, movie and game industries.

Visualization is used when a problem is not sufficiently well defined for a computer to handle automatically [Mun09]. If an algorithm exists that can completely replace human judgment, visualization is not usually required. Visualization functions as a form of external memory for our brain. We can represent and store data on the computer screen instead of representing it in

[1] Robert Collier was an American author of self-help, and New Thought metaphysical books in the 20th century. (Google)

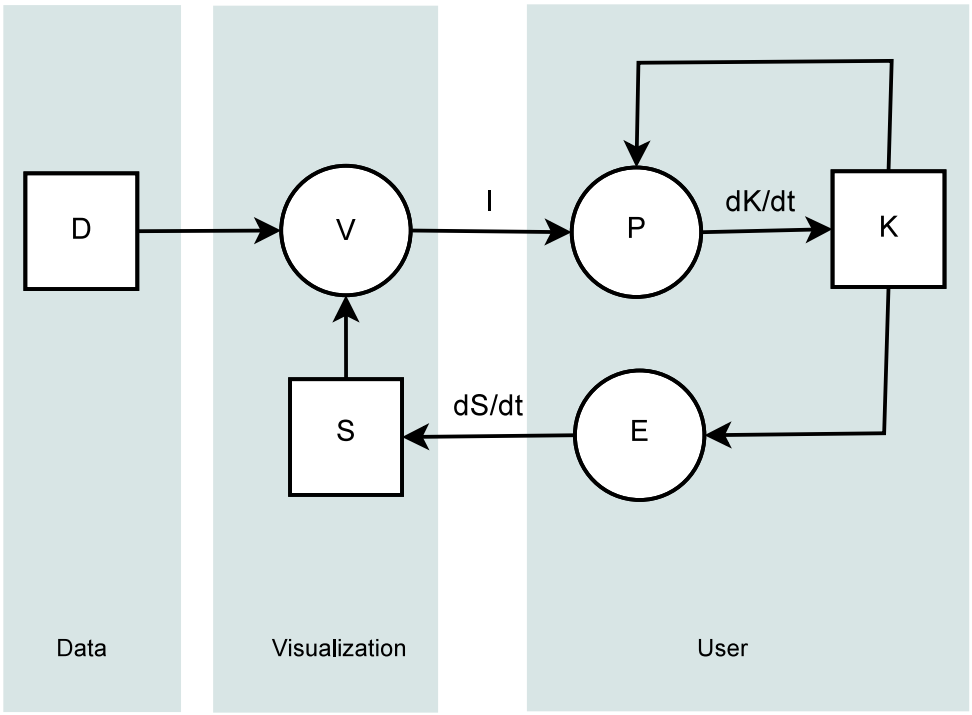


Figure 1.1: A model of visualization [VW05]. Boxes denote containers: D – data; S – specifications for visualization such as algorithms and their parameters, navigation, selection, and encoding; K – knowledge of the user. Circles denote processes: V – visualization; P – perception of the user; E – interactive exploration of data in which the user changes the specification (S) for visualization. Data D is transformed (visualized) according to a specification S into an image I. The image is perceived (P) by a user with an increase in his knowledge K as a result. The user may decide to adapt the specification (S) for visualization using interactive exploration E in order to explore the data further.

our mind. In "reading" this data, we take advantage of the the visual system, a high bandwidth information channel to the brain that enables processing at pre-conscious level. Visualization is used to generate new hypothesis for unfamiliar datasets, confirm existing hypothesis for partially understood datasets or present information about a known dataset to users unfamiliar with it.

A generic model for visualization that identifies its major components and clarifies its processes and goals is presented by VanWijk [VW05]. Data (D) is visualized (V) according to a specification (S) (Fig 1.1). The image produced (I) is perceived (P) by a user leading to an increase in their knowledge (K). The user may modify the visualization specification (S) through interactive exploration (E) to produce new visualizations. Using this iterative process, the user will acquire new knowledge that depends on the visualization method, on the specification used to produce it, on the perception and cognition of the user and on their current knowledge.

The visualization process V consists of a series of steps that manipulate data D and ulti-

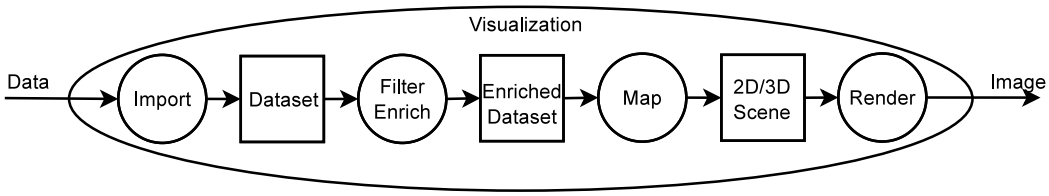


Figure 1.2: The visualization pipeline. Boxes denote containers and circles denote processes [Tel08].

mately delivers an image I [Tel08] (Figure 1.2). In the first step, raw data is *imported* into a specific dataset implementation. This can be as simple as loading the data from an external representation (disc or database), may involve converting from text (parsing), resampling data from the continuous to the discrete domain, or from one resolution or grid type to another. The imported dataset may contain too much data or data may be organized such that research questions can not be answered efficiently. The *filtering and enrichment* step is where the dataset is transformed and optimized for the desired visualization. Operations may include building hierarchical representations, mesh optimizations, feature extraction, zooming and panning or conversions from one dataset to another. The result is an enriched and optimized dataset that directly represents features of interest and can be used to efficiently answer scientific questions. The next step *maps* the enriched dataset into the visual domain. This creates a 2D or 3D scene which contains a set of visual features that correspond to dataset features. Each visual feature is typically a colored, shaded, textured and animated 2D or 3D shape with a specific position and size. Each of these attributes (color, illumination, texture, shape, position and size) can be used to encode attributes of the data. The mapping step is the operation in the visualization pipeline that is most specific to the visualization field. Importing, and filtering shares techniques with computer graphics, computer vision and signal processing while the last step in the pipeline, the *render* step, applies computer graphics techniques to render a 2D or 3D scene.

Based on the type of data it works on, visualization has two fields: scientific visualization and information visualization. Scientific visualization concentrates on datasets that contain a sampling of continuous quantities over the three dimensional space. On the other hand, information visualization works on abstract data that does not have an explicit spatial association such as graphs, trees, database tables, text and computer software. A major challenges in information visualization is to find an appropriate mapping between abstract data and space.

One of the most intuitive and wide spread [Tel08] taxonomies of scientific visualization algorithms (the mapping step in the visualization pipeline) is based on the type of attribute they work on. These techniques include scalar, vector and tensor visualization methods. Complementary algorithms can be used to modify the domain representation rather than show the attributes. These techniques include grid warping techniques, cutting and selection techniques, and resampling.

1.2 Contributions

In his significant work "On the Death of Visualization" [Lor04], Lorensen argues that visualization is in decline because it has lost its customers, the scientists. He argues that close relationship with life and physical sciences as well as alliances with other fields will provide the cure because they would expose our community to exciting application areas and present us with new and challenging problems.

Lorensen's insights inspired us to pursue scientific work in collaboration with physicists studying dynamic behavior of foam. We chose this collaboration field because we were intrigued by the challenges and potential benefits of improving the scientists' understanding of general foam behavior while little visualization work has been published in this area in the past [LLCD11]. We met our collaborators through the Research Institute of Visual Computing (RIVIC)[RIV10], a collaboration between research programs in Aberystwyth, Bangor, Cardiff and Swansea Universities. The goals of the institute are to make Wales an internationally leading country in visualization and other visual computing fields.

Liquid foams have important practical applications which include mineral separation, enhanced oil recovery, fire fighting, food and beverage productions. Foam research can help to improve the quality of products and efficiency of processes in these areas by predicting and controlling foam behavior. We give one example of financial benefits obtained through improving scientists' understanding of foam behavior.

In the mineral separation process, extracted rock is ground up, and then it is separated based on their hydrophobicity. Chemically treated minerals are hydrophobic, so when the grind is washed with foam, minerals stick to the bubbles while rock sinks to the bottom of the foam solution. The created froth is transported away for further processing. Improving this process has significant financial benefits. In a typical mine, about 90% of the mineral is recovered. An extra 1% mineral recovered is worth 20 millions dollars a year [Cil11]. It is estimated that 5% of the consumed world energy goes into grinding rock [Cil11]. Increasing the size of particles that can be separated, by improving the froth flotation process, will result in important energy savings.

Our main contribution is FoamVis, the first comprehensive visualization tool for time-dependent foam simulation data. Chapter 2 presents background information about foam and its simulation and it describes sample simulations studied with our visualization tool. We outline questions to which foam scientists seek answers, both in general and specifically for the presented simulations. We conclude by presenting existing visualization methods that foam scientists use to analyze foam simulations. In Chapter 3 we present the main foam research challenges and describe visualization solutions to address these challenges. We show individual time-step and time-dependent visualizations as well as solutions designed to enable comparison of related simulations. In Chapter 4 we describe relevant design and implementation notes that would allow a visualization scientist to extend the FoamVis system with new algorithms and adapt it to new requirements. Our work is a design study. We describe visualization solutions that address foam research challenges and we show, in Chapter 5, how scientists using our tool make new discoveries, validate hypotheses, and gain insights into foam behavior.

Chapter 2

Foam Research

> "A meringue is really nothing but a foam. And what is a foam after all, but a big collection of bubbles? And what's a bubble? It's basically a very flimsy little latticework of proteins draped with water. We add sugar to this structure, which strengthens it. But things can, and do, go wrong."
> – Alton Brown[1]

2.1 Introduction

Liquid foams have important practical applications in areas such as mineral separation, oil recovery, food and beverage production, cleaning and fire extinguishing [WH99]. Foam research can help to improve the quality of products and efficiency of processes in these areas by predicting and controlling foam behavior.

Foams are probably most seen in everyday life in cleaning products, however their presence is mostly due to consumer demands. Foam's ability to cling to a surface until washed away gives the perception that is better for washing but that is not the case [PK96]. Due to the presence of hydrocarbons in everyday life, extinguishing a fire with water is not only impossible but can be very dangerous. Water is too dense therefore it sinks below the surface of a burning liquid hydrocarbon where it can boil and explode, often with catastrophic consequences. Foam is much lighter than water so it flows over a burning liquid hydrocarbon providing a barrier and removing the supply of oxygen to the fire. The work presented in this thesis improves the general understanding of foam response which is of interest to manufacturers of cleaning products and in fire fighting.

In mineral separation ground ore is washed with foam. The efficiency of the separation of mineral from rock depends on how particles with different properties interact with foam. Some food products contain objects such as raisins or nuts within a solid foam. It is desirable to have the objects spread out throughout the product instead of being concentrated at the bottom due to gravity. Thus, a manufacturer needs to know what kind of liquid foam precursor is needed to support these objects so that it can make the best product in a competitive market.

[1] Alton Crawford Brown is an American television personality, celebrity chef, author, actor, and cinematographer. (Wikipedia)

Scientists idealize these processes by considering falling objects in an otherwise stationary foam (Figure 2.3).

For enhanced oil extraction, foam is pushed through porous rock to displace oil. Domain experts want to understand how the tortuous geometry of the rock pores affects the flow of foam. An important question related to this process is outlined next. When forced to flow through a constricted channel, many complex fluids, such as polymer melts, show regions in which material circulates in the upstream corners (salient corner vortices). As a consequence, material issuing from the channel can show markedly different ages, and therefore possibly different properties, and the flow-rate for a given pressure drop will change. In the case of foams, such recirculation may lead to particles dropping out of the foam before they can be captured or, in the case of food foams, material becoming unusable because of its age. Scientists are interested in determining if and when such recirculation occurs for various channel geometries. During the processing of many materials, including foams, extrusion is often used to fill moulds and trigger foaming. An important research question is how to design an optimal container shape to deliver the foam in such a way that its properties, such as bubble size and deformation, are controlled. The constriction simulations (Section 2.3) idealize the described processes and facilitate their study.

In contrast to solid foams constructed from, for example, aluminum or polyurethane, liquid foams evolve in time, presenting a more difficult challenge for researchers. Physicists use simulation to study basic properties of foam that are still not well understood. For example, can the path that a bubble traverses be predicted? How do bubbles and soap films behave under stress and shear? To what extent do objects falling through a foam interact [Dav09]? Does the whole foam flow when pushed through a constricted geometry?

The main goal of foam research is to determine foam behavior from measurable properties such as bubble size and its distribution, liquid fraction, and surface tension. One way to study this dependence is to simulate foam at the bubble level. This type of simulation makes it possible to model foam properties and see their influence on general foam behavior, to better inform continuum models of foam dynamics, and to make direct comparison with experiments. Surface Evolver (SE) [Bra92] is the *de facto* standard for studying foam simulations at the bubble-scale, which are the most accurate both in terms of structure and flow.

2.2 Background

We show a continuum model of foam rheology and present equilibrium laws that determine foam structure. This section informs our visualization work for foam simulation data.

2.2.1 A Continuum Model of Foam Rheology

An aqueous foam is a two-phase material, for example detergent-laden water and air, yet its response can often be solid-like. To accurately predict the rheological properties of foams, including stiffness (shear modulus) and viscosity as well as the complex response described in Equation (2.1), requires a treatment that differs from the usual methods for predicting fluid flow. The solid-like properties are often attributed to a shear modulus, defined as the derivative of stress with respect to strain. Plasticity is described by a yield stress σ_y, that is, a critical applied

stress σ below which the material does not flow, and then fluid-like properties are captured by an effective viscosity, defined loosely as stress divided by strain rate $\dot{\gamma}$. Both the yield stress and the rate of strain are part of the Herschel-Bulkley constitutive relation, well-known in the field of complex fluids:

$$\begin{cases} \dot{\gamma} = 0 & \sigma \leq \sigma_y \\ \sigma = \sigma_y + K\dot{\gamma}^n & \sigma > \sigma_y \end{cases}, \tag{2.1}$$

where K (consistency) and $0 < n \leq 1$ (power-low exponent) are fitting parameters [KMv08, Den08]. Such an expression for the stress can now be implemented in a number of commercial CFD packages as a generalized Newtonian model. What is unclear in this approach is the dependence of the parameters K and n on material properties, for example bubble size (and its distribution) and liquid fraction.

A possible solution simulates foams at the bubble scale, where these contributions can be teased out [CW06]. Surface Evolver (SE) allows a precise representation of the bubbles based upon the observation that a soap film minimizes its surface area. In contrast to other methods, SE can span a large range of liquid fractions and allows for the container geometry to be varied. It is the only standard method that allows the bubble pressures to be calculated, because of its precision. Indeed, where accuracy is required, both in terms of static structure and rheology/flow, SE is the *de facto* standard for foam simulation.

A challenge of using SE to simulate foam behavior, at least for bulk flows (flow far from the walls of the foam container), is to be able to simulate a sufficiently large number of bubbles to accurately reflect reality. Thus, many simulations are performed in two dimensions, where simulation time scales with the number of bubbles slightly greater than linear [Wyn09]. Fortuitously, a good approximation to a two-dimensional foam can also be made experimentally, for example by squeezing bubbles between parallel glass plates [Smi52] or using the Bragg bubble raft [BN47], thus providing a means to validate simulations.

2.2.2 Foam Structure

Foam is a two-phase system of gas bubbles separated by a thin liquid network. The liquid-gas interface of the foam follows specific equilibrium laws that determine its local structure.

Surface tension causes soap films to minimize their area. Surface tension results from an imbalance of the cohesive forces between molecules. These forces act on a molecule in the liquid from all directions while they act on a molecule at the surface of the liquid only from inside the liquid. If surface molecules would be displaced slightly, they would be attracted back by nearby molecules. This causes a soap film to act as if it were a stretched elastic membrane. Surface tension can be viewed as the result of forces acting in the plane of the surface tending to minimize its area [bri14].

Lapace-Young law links the curvature of the film separating bubbles (κ) with the pressure difference across the gas-liquid-gas interface (Δp) and the film surface tension (2γ): $\Delta p = 2\gamma\kappa$. In two-dimensions, films have a single curvature so they are circular arcs. In this case, the film curvature is given by $\kappa = 1/r$, where r is the radius of the circular arc (see Figure 2.1).

Plateu observed [Pla73] that minimization of film areas results in geometrical constrains. *Plateau law* states that for dry foams at equilibrium films always meet threefold at an angle

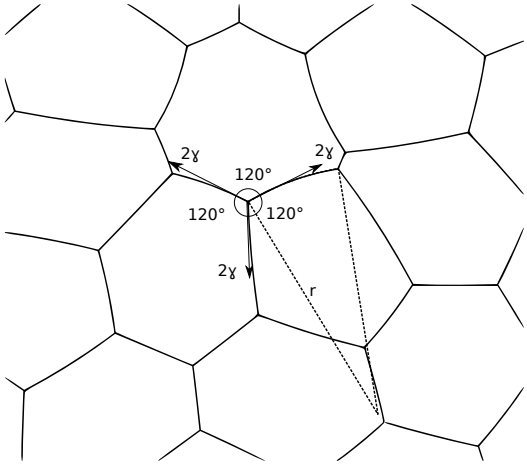


Figure 2.1: The structure of a dry, two-dimensional foam. Bubbles are separated by films which are circular arcs with curvature dependent on bubble pressure and surface tension. Films meet threefold at 120° angles.

of 120° to each other (see Figure 2.1). This result was proved by Taylor [Tay76], more than a hundred years after Platau stated it.

An important foam parameter is the liquid fraction, which indicates where a given foam lies between the limits of a dry foam, in which the liquid walls of the structure (soap films) are thin and the gas bubbles polyhedral, and a wet foam, in which the gas bubbles are spherical. Liquid fraction is defined as $\Phi = V_l/V_g$, where V_l and V_g are, respectively, the volumes of liquid and gas in the foam. A 2D foam with $\Phi = 0.16$ resembles a close packing of circular discs [Dav09]. Our work focuses on simulations of dry foam where $\Phi < .05$. In this case, most of the liquid is contained within the *Plateau borders* of the foam, which are channels of finite width that replace the lines in a dry foam. Further more the simulations we examine consider an ideal dry foam with $\Phi = 0$.

2.3 Foam Simulation Cases

Foam dynamics studies physical processes that affect the structure of foam. These processes include coarsening, drainage and rheology.

Coarsening is a process in foam where gas diffuses from one bubble to another so that some bubbles grow and others shrink and may disappear altogether [WH99]. In drainage, the liquid in foam drains downwards because of gravity, viscous forces oppose this and surface tension provides a "capillary" effect such that at equilibrium, some liquid is kept in the foam. These processes are ignored in the simulations we study, so building specific visualizations to study them is left for future work.

Our main goal is to help foam scientists to improve predictions of the complex rheology of foam. Foam behaves like an elastic solid under low stress meaning that it is able to return to its original form once the applied stress is removed. Its elastic modulus is of the order of 10 Pa, in

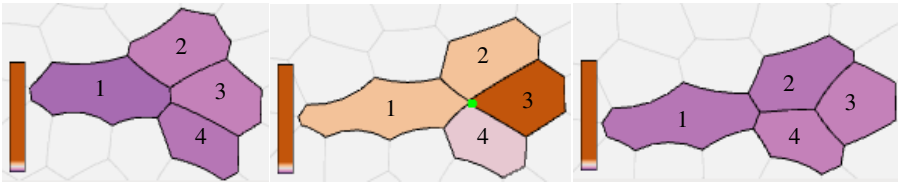


Figure 2.2: Topological change (T1 event). The three images show time steps t=20, t=72 and t=73 in the *constriction* simulation. A T1 event can be observed between the bubbles color-mapped by velocity magnitude. In the left image bubbles meet only at three-fold vertices, with bubbles 1 and 3 touching. The four bubbles move to an unstable configuration in which they meet at a four-fold vertex in the middle image. Note the high velocity that bubbles have after the T1 event. The right image shows the four bubbles after the T1 event where now bubbles 2 and 4 touch.

comparison with the value for 8×10^{10} Pa for steel [WH99]. Foam elastic behavior is entirely due to the surface tension of the soap films. Bubbles become elongated or squeezed without rearranging.

Foam behaves like a plastic solid as applied stress is increased which means that foam permanently deforms. When the applied stress forces two three-fold vertices to approach each other and move into an energetically unstable topology of a four-fold vertex, the films rearrange to minimize the foam's total energy (see Figure 2.2).

Foam behaves like a liquid for stress greater than an yield stress. Liquids exhibit viscosity which is a measure of their internal resistance to flow. In the simulations examined, viscous effects of the foam are deemed negligible.

To probe the rheology of foam, we consider two simulation groups containing related simulations: constriction and sedimenting objects. The simulations in both groups are periodic in the direction of motion. The constriction simulation group contains two simulations, one with a *square-constriction* and one with a *rounded-constriction* (Figure 3.7). They simulate a 2D polydisperse (bubbles with different volumes) foam flowing through a constricted channel, with 725 bubbles and 1000 time steps. The channel has unit length and the length of the constricted region is 0.148. Its width is 0.5 and the width of the constricted region is 0.24. The simulations differ from each other in the geometry of the constriction. The radius of the circles creating the rounded corners of the constriction (Figure 3.2) is 0.014 for the square-constriction and 0.069 for the rounded-constriction. These simulations subject foam to different kinds of stress simultaneously and are therefore a testing ground from which to validate the approximations in the model against experiment.

The falling objects simulations group contains the sedimenting ellipse/discs simulations (Figure 5.5) and the falling disc/sphere simulations (Figure 5.8). We wish to understand the interaction between two sedimenting discs by comparing it with the (simpler) behavior of a sedimenting ellipse. We wish to compare and validate the 2D falling disc simulation with the 3D falling sphere simulation. *Sedimenting-discs* simulates two discs falling through a monodisperse (bubbles having equal volume) foam under gravity. It contains 330 time steps and simulates 2200 bubbles. The two discs are initially side-by-side and in close proximity. As they

fall, they interact with the foam and with each other and rotate towards a stable orientation in which the line that connects their centers is parallel to gravity. There are two forces acting on each disc in addition to its weight. A pressure force results from each adjacent bubble pushing against it, while a network force arises because each contacting soap film pulls normal to the circumference with the force of surface tension. Due to the flow, the distribution of films and bubbles pressures around each disc is not uniform (for example, there is a high density of films above each disc, leading to a large, upward, network force there), resulting in a non-zero resultant force. *Sedimenting-ellipse* simulates an ellipse falling through a monodisperse foam under gravity. This simulation contains 540 time steps and simulates 600 bubbles. The major axis of the ellipse is initially horizontal. As the ellipse falls, it rotates toward a stable orientation in which its major axis is parallel to gravity. As for the sedimenting discs, a network and pressure force act on the ellipse and, due to its shape, they give rise to a non-zero torque that rotates it. We seek to validate the idea that the anisotropic two disc system responds to the foam-induced forces in the same way as an elliptical object. For the falling disc/sphere simulations we have 254 time steps and 1500 bubbles in 2D and 208 time steps and 144 bubbles in 3D. Note that the number of bubbles that we are able to simulate in 3D is severely restricted by the duration of the simulation. These simulations are variations of the classic Stokes experiment [Sto51] that is used to probe the rheology of a 2D foam and for which there is a great deal of experimental data. The Stokes experiment is the basis of the falling-ball viscometer that is used to measure the viscosity of a fluid. A spherical ball is left to fall under its weight in a cylindrical tube filled with the fluid to be measured. The terminal velocity of the ball is measured and the viscosity of the fluid is deduced using the Stokes' law that links the force applied to the ball (gravity) to viscosity, radius of the sphere and its velocity.

The falling objects simulations are relevant to industrial processes in separation and oil recovery [WH99]. For these simulations, we address a number of research questions. Do the two discs act as a large ellipse so is it possible to think of a torque acting on the system? If the answer to this question is positive, this will explain the complex behavior of two discs sedimenting in foam, where one disc rotates around the other. For the constriction simulations how does the foam respond to differences in container shape, and can it lead to non-trivial flow, such as recirculation and the recycling of material? In general - under what circumstances does a foam respond plastically or elastically? How does changing the container, or the object shape, affect that balance?

2.4 Simulation Method

The initial structure for each simulation is created from a Voronoi tessellation of the unit square, with random seeds and periodic boundary conditions. Bubbles are removed from two opposite sides to leave a structure which is only periodic in one direction, with vertices fixed to solid walls in the other direction (Figure 2.3). The domain scientists use the Surface Evolver, a finite element code, in a mode in which each film is represented as a circular arc, to find a realistic foam structure by minimizing the total film length subject to the prescribed bubble areas.

During this minimization, topological changes (T1s) (Figure 2.2) are triggered by deleting each film that shrinks below a critical cut-off length l_c and allowing a new film to form in

a perpendicular direction to complete the process. The critical length l_c is a measure of the effective liquid fraction Φ.

To apply a pressure gradient to the foam, a line of films spanning the channel is moved downstream by a small distance (constriction). A sedimenting disc is moved a small distance in the direction of the resultant force on it. In both cases, this motion is followed by a reduction of the film length to a local minimum (in the sedimenting discs simulation, the discs are fixed during the minimization). Either non-slip (films attached to the wall do not move because of high friction; sedimenting-discs) or free slip (no resistance to motion along the wall; constriction) boundary conditions can be applied at the channel walls. In this way, the foam passes through a sequence of equilibrium states, appropriate to an applied strain with strain rate much lower than the rate of equilibration after T1s.

2.5 Foam Research Questions

Foam scientists' main goal is to develop a model that successfully predicts foam behavior from measurable properties such as bubble size, disorder and liquid fraction. It is the aim of simulations to identify and separate the influences of these properties on foam response. Matching simulated foam behavior with experiments provides validation for the simulations.

We outline specific questions that foam scientists try to answer about the presented simulations. For the simulation of foam flow through a constricted channel important questions include: For what range of parameters can recirculation zones be found in the upstream corners? That is, are there regions where bubbles are trapped and move in circles (or not at all) rather than downstream, therefore not contributing to the transport of material? What is the pressure drop required to force the flow through a given constricted geometry? Important questions for the simulation of sedimenting discs include: Do the two discs descend at the same speed? Do they interact? If so, why, and under what conditions? How do the forces exerted on each disc influence its motion and how do they depend on their relative position? For any foam simulation: what effects do topological changes (T1s) have, and for how far do these effects extend?

2.6 Standard Methods for Foam Visualization

We identify the properties of the foam in which we are interested and the standard methods of visualizing the simulations using the constriction example.

Firstly, each bubble acts as a tracer, so its velocity, defined as the motion of the center of mass, provides traditional information about the flow: velocity vectors, trace streamlines and velocity components. In addition, each bubble has a well-defined pressure, which, when averaged over the duration of the simulation, can, as for velocity, be visualized using color mapping or slices across and along the flow direction. A further benefit of the use of bubbles as tracers is that their deformation gives information about the local strain. Quantities such as the texture tensor [GDRM08], which takes an average of distances between bubble centers in a given representative volume element, can also be displayed as color maps, slices, and ellipses.

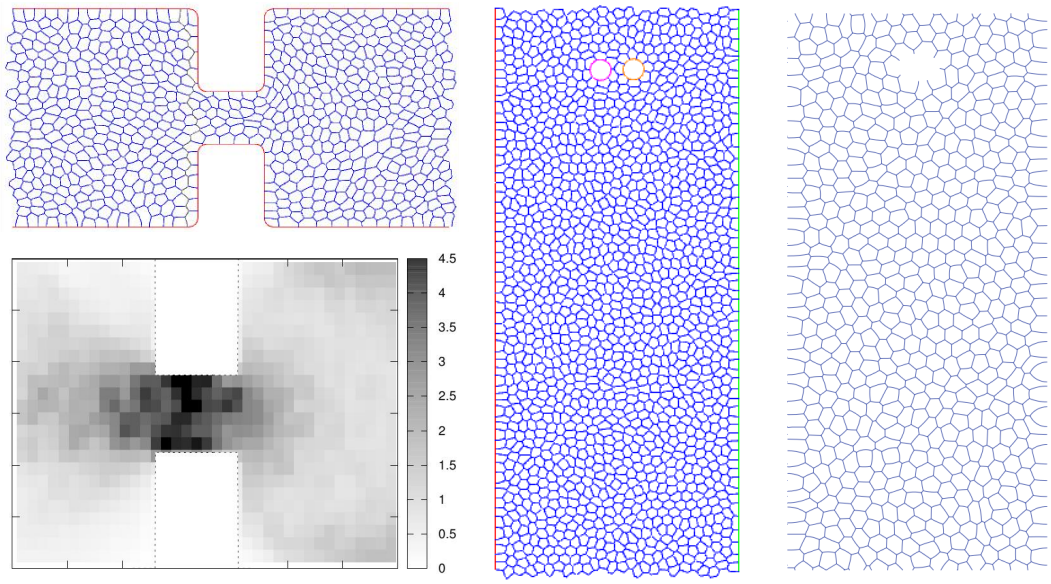


Figure 2.3: Typical visualizations used by the domain experts. Images are visualizations of the simulation of foam flow through a constriction [JDS+11] and of the simulation of two discs falling through a foam [Dav09]. The left-top image shows an instantaneous image of the foam flowing through a constriction. The left-bottom image shows velocity magnitude averaged over all time steps over a 60x30 grid. The middle image shows time step $t = 0$ of the discs falling through a foam simulation. The right image shows time step $t = 0$ of the ellipse falling through a foam simulation. Note that the elliptical object is represented by a void in the foam.

Finally, the skeletonized views of the foam can themselves be animated. Figure 2.3 shows examples of standard visualizations used by the domain experts.

Standard animation of the foam skeleton makes the analysis and visualization of individual bubbles extremely difficult over many time steps. The standard averaging techniques do not provide very much detail and are not well suited for the simulation of sedimenting discs. Line graphs decouple important information from the space-time domain. We demonstrate how our visualizations address these drawbacks.

Chapter 3

Foam Visualization

"What lies behind you and what lies in front of you, pales in comparison to what lies inside of you."
– Ralph Waldo Emerson (1803–1882)[1]

3.1 Motivation

The main goal of foam research is to determine foam behavior from measurable properties such as bubble size and its distribution, liquid fraction, and surface tension. One way to study this dependence is to simulate foam at the bubble level using Surface Evolver (SE) [Bra92]. This type of simulation makes it possible to model foam properties and see their influence on general foam behavior, to better inform continuum models of foam dynamics, and to make direct comparison with experiments.

Foam scientists work with dozens of simulations with a wide range of simulation parameters. Examples include foam container properties (such as channel geometry), foam attributes (such as bubble size and distribution, liquid fraction and surface tension) or the properties of objects interacting with foam (such as particle shape, size or position). The goal of varying these parameters is to model the foam response and to validate simulation against experiments.

Foam scientists use 2D foam simulations to model foam behavior as it is simpler to simulate and analyze. A 2D foam can be created experimentally by squeezing bubbles between parallel glass plates [Smi52], thus providing a means to validate simulations. However, most real foams are 3D. Two-dimensional foam simulations might introduce additional errors and 2D foam experiments suffer from effects such as wall drag. Foam scientists would like to evaluate 3D foam simulations and assess and analyze differences between 2D and 3D simulations, but few visualization solutions exist for 3D foam simulation data.

This work concentrates on specific challenges posed to researchers by Surface Evolver foam simulations.

1. Access to simulation data is difficult and requires domain specific knowledge. Parsing

[1] Ralph Waldo Emerson was an American essayist, lecturer, and poet, who led the Transcendentalist movement of the mid-19th century. (Wikipedia)

and special processing are required to access the full simulation data. Important bubble attributes are not provided by the simulation but inferred using domain specific knowledge.

2. Triggers to various foam behaviors are difficult to infer. Multiple attributes have to be examined and foam properties have to be taken into account. Topological changes (T1s), in which bubbles swap neighbors, have to be considered.

3. It is challenging to visualize general foam behavior. While bubble-scale simulation makes it possible to investigate the influences that material properties have on general foam behavior, it makes it difficult to visualize the general behavior that is of primary interest. Simulation data is complex (unstructured grid with polygonal cells) and time-dependent, with large fluctuations in the values of attributes determined by changes in the topology of the soap film network (T1s).

4. The large number of existing simulations and the variety of simulation parameters makes it difficult to manage simulation data and to understand the influence that simulation parameters have on foam behavior. Comparing related datasets results in a better understanding of various foam behaviors, however previous tools do not facilitate that.

These challenges make it difficult to use a general purpose visualization tool for foam simulations. Domain experts' visualizations only partially address these challenges. They may require intervention in the simulation code and potentially recomputing the simulations for summarizing and saving the relevant data. Their standard visualizations do not have the ability to explore and analyze the data through navigation, selection and encoding operations. They do not have the high level of detail and speed that is achieved using graphics hardware. We address shortcomings of existing visualizations used by domain experts and we provide visualizations to address foam research challenges. To the best of our knowledge, no visualization software exists for foam simulations modeled with SE. FoamVis fills this void by providing a comprehensive solution which facilitates advanced examination and analysis of foam simulation data.

3.2 Related Work

In our survey of the literature [LLC+12], very little work in visualization of time-dependent, physically accurate foam simulation data has been published. We present related works that visualize static foam structures. Bigler et al. [BGG+06] explain and evaluate two methods of augmenting the visualization of particle data using ambient occlusion and silhouette edges. They visualize a foam data set acquired using micro-CT by converting it first to a particle distribution and then using an interactive ray-tracer for rendering. König et al. [KDK+00] present an interactive tool to investigate the structure of metal foam. They employ techniques for real-time isosurface rendering, determine the isosurface threshold value such that the volume of the computed foam sample matches the real-world sample and render cells with certain size and shape criteria. Hadwiger et al. [HLRS+08] present a method for interactive exploration of industrial CT volumes such as cast metal parts. They detect and classify defects in a material

using interactive exploration instead of an offline process of setting parameters and then waiting for the results. These works focus on visualization of static foam or foam-like structures, while our work presents visualization of time-dependent foam simulations.

Computer graphics researchers are also interested in rendering soap bubbles [Gla00a, Gla00b, Ď01], foam [SRK08] and water sprays [LTKF08]. However, they simulate and render the appearance of natural phenomena while avoiding the large computational cost of physically accurate simulations.

Existing tools to manipulate Surface Evolver data include *evmovie*, which is distributed with Evolver, and the Surface Evolver Fluid Interface Tool (SE-FIT) [CSWZ11]. Evmovie scrolls through a sequence of evolver files, while SE-FIT provides a graphical interface for interacting with Surface Evolver. Ours is the first work of its kind (to our knowledge) to focus on visualization of time-dependent foam simulation data.

Comparative visualization refers to the process of understanding the similarities or differences between data from different sources. Differences between simulations and experiments, or between simulations or experiments with different parameters may be of interest. Such analysis can happen at different levels: image, data, derived quantities, and methodology levels. At the image level, the two sources can be compared by using two visualization images shown side by side [AHP+10], superimposed [PP95], as two symmetrical halves or by computing the difference between the two images. If images from several sources need to be compared, a space filling tilling can be used [MHG10]. At data level, data fields from the two sources are combined to produce a new visualization. Derived quantities or features can be extracted and compared, for instance streamlines in a vector field, vortex and shock wave positions [PW95] or detected edges in slices of an industrial scan [MHG10]. Differences in experiment, simulation or visualization parameters may be quantified and compared.

The goals of the reviewed works in comparative visualization are to find the optimal solution from a number of datasets or to understand how the datasets are different. For our work, the goal is to improve understanding of foam behavior, and as a result, produce better foam models. To reach this goal, we use a number of comparative visualization techniques designed to reduce the cognitive load required to integrate two side-by-side views, and we propose a new interaction and processing technique that allows scientists to compare events in related simulations while facilitating the examination of the temporal context for the events.

3.3 Visualization

Our visualization solutions are driven by the foam research and visualization challenges (Section 3.1). Section 3.3.1 describes the processing required to read SE output files and access the complete data generated by the simulation. Our application works with any SE simulation and no changes to the simulation output are necessary to accommodate the application. This processing addresses challenge one.

Visualizations of individual time steps is done using color-mapping for scalars, glyphs or streamlines for vectors and glyphs for tensors. These visualizations are used as the basis for, or for augmenting, more complex visualizations. Enhancements to these visualizations include bubble selection, histogram-guided color-bar clamping and T1 information overlay

that shows the topological changes (T1s) triggered in the current time step. Bubble selection and/or filtering is used for debugging (selection by bubble ID), for studying foam properties at certain locations in the domain (filtering by location) or for analyzing bubbles with certain attribute values (filtering by attribute value).

Overall foam behavior (challenge three) is analyzed using the image-based statistical computation (Section 3.3.3), visualization of bubble paths (Section 3.3.5) and a kernel density estimate (KDE) for the location of topological changes (T1s) (Section 3.3.6). The KDE for T1s is computed to illustrate the distribution of plasticity in foam. With this visualization we address over-plotting issues present if we just render the location of each topological change.

Foam scientists wish to understand what triggers certain behavior in foam simulations (challenge two). Different views provide different information about the various influences on foam behavior. We enable the examination of several views at the same time using the multiple linked-views (Section 3.3.7).

Histograms provide important information about foam data, are used for bubble selection based on the values of attributes, and for providing the context for color map clamping. They are presented in Section 3.3.8. Interaction operations including navigation, selection, filtering, encoding and connections between multiple linked-views [WGK10] are described in Section 3.3.10. These features are used throughout the application and help to address both challenges two and three.

Techniques to facilitate comparing related simulations (challenge four), which include the *two halves* view, the *linked time with event synchronization*, the *reflection* feature and the *force difference*, are presented in Section 3.3.9.

3.3.1 Data Processing

In this section, we present the data processing applied before visualization. We parse Surface Evolver (SE) output files, then we unwrap geometric elements of the foam described by periodic boundary conditions. We process bubble edges that lie on the zero level set of a function such that we accurately represent foam channels and objects interacting with foam. This kind of edges can have an arbitrary shape as described by the function in the simulation file. This processing enables analysis of datasets such as the sedimenting ellipse simulation because we can accurately represent objects boundaries (Figure 2.3 right). Additional processing includes calculation of derived attributes, bubbles' centers, bounding boxes and statistical measures. We accelerate the data processing phase by parsing and pre-processing time steps in parallel on all available CPU cores.

3.3.1.1 Parsing

Foam simulation data consists of a list of SE output files, one per time step. A file stores the entire configuration of the simulated foam at a particular time step. For maximum generality and flexibility, we parse SE files directly instead of using derived files created by foam scientists. This allows our application to work with any simulation created using SE and at the same time it gives us access to the entire state of the simulation.

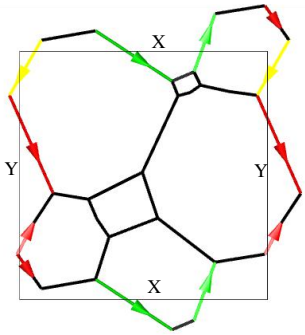


Figure 3.1: Foam with periodic boundary conditions (PBC). The domain is shown as a black rectangle. A tessellation of the infinite plane can be obtained by tiling it with the domain. Edges that do not cross a domain boundary are shown in black. Edges that cross a domain boundary are colored depending on the boundary they intersect: red for the X boundary, green for the Y boundary and yellow for both X and Y boundaries. Arrows show whether an edge enters or exits the domain.

A Surface Evolver (SE) file [sur08] is organized into six parts: definitions and options, vertices, edges, faces, bodies and commands. The first section contains, in addition to options controlling the simulation, constant expressions for foam material and geometric parameters, additional attributes that are attached to geometric elements (vertices, edges, faces and bodies) and functions for defining level-set constraints. A level-set constraint is a restriction of vertices to lie on the zero level-set of a function. For instance, the foam container in the constriction dataset (Figure 2.3 left-top) is defined using a level-set constraint. The lists of geometric elements (vertices, edges, faces and bodies) follow the format described next. Each line is comprised of entries which define an element. The first entry is the element ID followed by geometry data, followed by optional attributes. The geometry data consists of coordinates for a vertex, begin and end vertex for an edge, list of ordered edges for a face, and list of oriented faces for a body. Besides the built-in attributes for each element type, one may specify values for extra attributes. The commands section is used for controlling the simulation and it is ignored by our program.

Our tool can read the following optional data that is saved by the simulation code: a list of T1s and the network and pressure forces that act on a body. T1s are stored one per line. Each line contains the time step at which the T1 occurs, and the position of the T1. The network and pressure forces are stored component wise in SE variables.

3.3.1.2 Unwrapping for Periodic Boundary Conditions

After parsing foam simulation data and creating the corresponding data structures, we perform additional data processing. First we compact each list of geometric elements as there can be numbering gaps in the list specified in a SE file. Then, if the foam described in the SE file contains periodic boundary conditions (PBC) [sur04, sur08] we unwrap the geometric elements so that we can display the foam.

For a foam with PBC, the domain boundary intersected by each directed edge and whether the edge enters or exits the domain (Figure 3.1) is specified. All vertices are defined inside the domain. To unwrap edges, we use intersection information between an edge and a domain boundary. This may create vertices defined outside the domain. To unwrap faces, we follow connected edges along a face. This may create edges defined outside the domain.

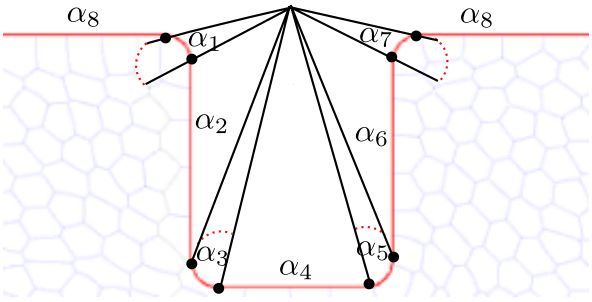


Figure 3.2: Level-set constraints for defining the shape of the foam container for the *square-constriction* simulation.

3.3.1.3 Level-Set Constraints

The shape of foam containers and of objects interacting with foam are specified using level-set constraints, that restrict bubble vertices to lie on the zero level-set of a function. Level-set constraints can apply to vertices or edges. If a constraint is applied to an edge, all points on that edge are on the level-set of the given function. There are two challenges associated with edges on level set constraints. First, when using the simple approximation of representing a constraint edge using a single line segment, time-averaging bubbles with edges on level-set constraints causes artifacts, as the constraint is too coarsely approximated. More importantly, an object described with a level-set constraint is not represented as a standalone entity in the simulation data (Figure 2.3 right), making it difficult to study foam properties around it.

A level-set constraint is defined in the simulation file using an integer ID followed by a function $f(x,y)$ which specifies the curve on which a vertex lies. The function can contain constants and can be defined piecewise using a conditional operator. An example of a piecewise constraint is the top edge of the constriction simulation foam container in Figure 3.2. For each angle α_1 through α_8, the constraint is defined as the intersection between the area determined by the angle and a function. For instance, for α_1, the function is a circle, while for α_2 it is a straight line.

Unfortunately, for α_1, α_3, α_5 and α_7, the constraint is defined in an ambiguous way, as it defines an additional circular arc (represented with a dotted line) in addition to the correct one. This may sometimes cause problems, when a vertex defined to be on this constraint may converge on the wrong arc. This was observed in both our application and in the Surface Evolver, the simulation software used by foam scientists.

In a simulation file, an edge on a constraint is specified using the end points of the edge and the constraint. To obtain other points on the edge, used to approximate the edge using straight line segments, we apply the following method (Figure 3.3). We divide the line segment $\overline{PQ}$ in equally spaced points A_i, where P and Q are the end points of the edge. For each A_i we determine point C_i on the constraint, by computing the intersection between the constraint and the line A_iC_i perpendicular to $\overline{PQ}$. The intersection is computed using a multi-dimensional root solver [gsl11, PTVB07] using the initial guess for the intersection A_i. If the solver fails to converge we try to find the solution using a different initial guess B_i, the corresponding point

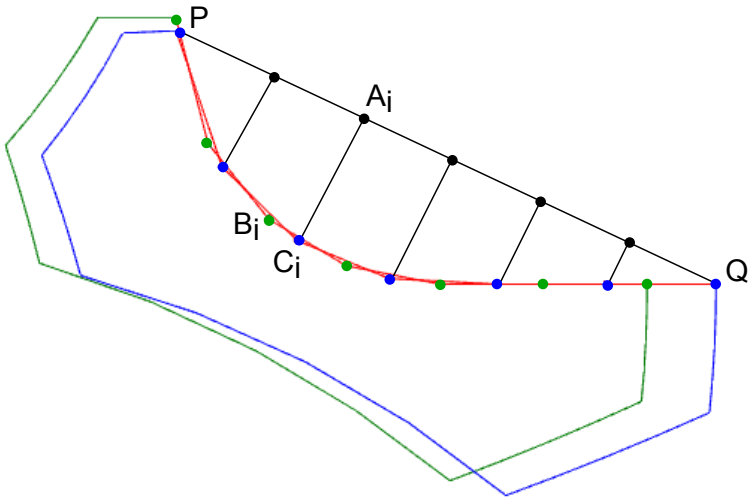


Figure 3.3: Approximating edge PQ on a constraint. We display a bubble at time steps t and $t-1$ from the *square-constriction* simulation. The edge on the level-set constraint is displayed in red. Edges not on a constraint are displayed in blue for time step t and in green for time step $t-1$. We place equally spaced points A_i on segment $\overline{PQ}$, where P and Q are the end points of the constraint edge. We find the intersection between line A_iC_i and the constraint using a root solver. We try two initial guesses, first A_i and then B_i, the corresponding point on the constraint edge in time step $t-1$.

on the same edge in the previous time step. Bubbles often move slowly, making B_i potentially closer to C_i leading to the convergence of the root solver.

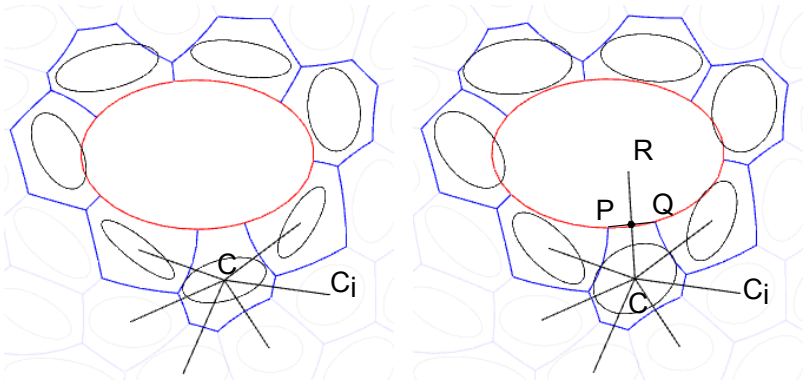
After all edges on constraints are created, we create a new body for each constraint ID that represents an object interacting with foam. These ID are passed as a parameter from the command line. For instance, the ellipse in Figure 3.4 is created from edges on a constraint (compare with Figure 2.3 right).

3.3.1.4 Additional Processing

We calculate each bubble's center of mass, bounding box and the bounding box of the foam at each time step and overall, and calculate statistical quantities such as histogram, minima and maxima for values of attributes.

Two important scalar attributes provided by the foam simulation are pressure and volume. We derive the following attributes important to foam scientists. Velocity, computed by connecting a bubble's center of mass between two consecutive time steps, and elongation, computed by $P/\sqrt{A}$ (in 2D) where P is the bubble's instantaneous perimeter and A is its area.

In SE simulations, there are high fluctuations of pressure values between successive time steps (Figure 3.5 top). These fluctuations do not have any physical significance, but they are a consequence of the how pressure is recorded in the simulation files. We apply the processing in Algorithm 1 which eliminates pressure median variation and yields a smooth transition be-



(a) (b)

Figure 3.4: Deformation tensors for bubbles around an ellipse. (a) Only tensors for neighboring bubbles are averaged. (b) Tensors for neighboring bubbles and the reflection tensor $\vec{CR} \otimes \vec{CR}$ are averaged. Using the reflection tensor leads to a more accurate representation of deformation for bubbles touching the ellipse.

Algorithm 1 Align median pressure between time steps

Translate all pressures in a time step with a constant value such that the minimum pressure is zero.
Calculate the median pressure for each time step.
Calculate *maxMedian* the maximum of all medians.
Translate pressures in a time step with (*maxMedian* - median) for the time step.

tween time steps and a more meaningful representation of pressure (Figure 3.5, bottom). SE records the pressure values of the bubbles relative to the pressure of a reference bubble, which is considered to have zero pressure. This is because to find the mechanical equilibrium state of the foam it is enough to calculate pressure *differences*, not absolute values. We started by calculating a bubble's center of mass assuming that its mass is concentrated in its vertices. However this results in bubbles that appear to wobble when their center is set stationary (Section 3.3.3). The reason for this is that many vertices on one side of a bubble will pull its center of mass toward that side. We solve this problem by computing the center of mass, assuming the mass is uniformly distributed on the bubble's area [BD94].

3.3.2 Deformation Tensor Computation and Visualization using Ellipses

While visual inspection of individual bubbles provides information about foam deformation, this information is not quantified, and, more importantly, cannot be averaged to obtain the general foam behavior. To address these issues, we compute the bubble deformation measure as defined by Graner et al. [GDRM08]. Let i and j be two neighboring bubbles (which share an edge), let C_i and C_j be their centers and assume that the vector $\overrightarrow{C_iC_j}$ has components (x,y).

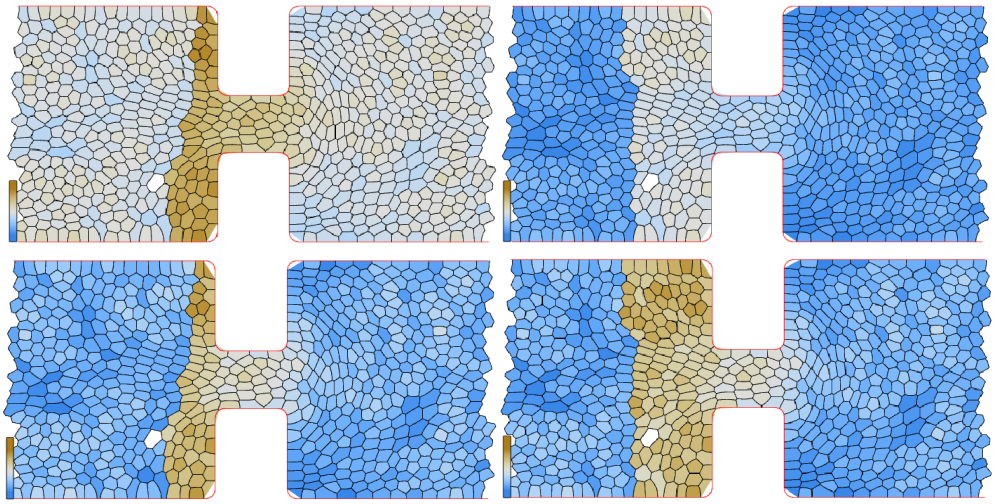


Figure 3.5: Adjustment for relative pressure. Bubbles are color-mapped to pressure values. We show time steps $t = 26$ and $t = 27$: (top) before adjustment and (bottom) after adjustment. The white bubble is the reference bubble. A discontinuity in pressure values is present along the line of films that is moved downstream in each simulation step (Section 2.4).

We define a tensor $\mathbf{T_{ij}}$ as the direct product of $\overrightarrow{C_iC_j}$ with $\overrightarrow{C_iC_j}$.

$$\mathbf{T_{ij}} = \overrightarrow{C_iC_j} \otimes \overrightarrow{C_iC_j} = \begin{pmatrix} x^2 & xy \\ xy & y^2 \end{pmatrix}$$

The deformation tensor $\mathbf{T_i}$ for bubble i is defined as $\mathbf{T_i} = \sum_{j=1}^{n} \mathbf{T_{ij}}/n$ where n is the number of neighbors of bubble i. This tensor is positive definite symmetric (each element of the average is positive definite symmetric) so it has positive eigenvalues. The deformation tensor is represented by an ellipse [GDRM08] that has each axis' length and direction given by the tensor's corresponding eigenvalue and eigenvector. This representation shows both deformation value (ellipse eccentricity) and direction (the orientation of the ellipse) and can be averaged over an area and over time. We use these properties to show large scale (general) deformation behavior in foam obtained by time-averaging bubble-scale behavior.

We extend this definition for bubbles that touch an object (or the foam container) by including another tensor we call the *reflection tensor* in the average computation. The shared edge between the bubble and the object, with end points P and Q, is a constraint edge because objects interacting with foam are defined using level-set constraints. The new tensor included in the average is $\overrightarrow{CR} \otimes \overrightarrow{CR}$, where R is obtained by reflecting C, the bubble center, against the middle of segment $\overline{PQ}$.

The inclusion of the reflection tensor in the deformation tensor computation yields a more accurate representation (compare Figure 3.4a to Figure 3.4b) of the deformation of bubbles neighboring objects or the foam container. The reflection tensor is just one of the possible adjustments for bubbles neighboring constraints. For instance, an alternate adjustment is to reflect the bubble center in the mid point of the constraint edge.

In 2D, the standard measure of deformation is computed by $P/\sqrt{A}$ where P is the bubble's instantaneous perimeter and A is its area. We add a new scalar measure of deformation given by the ellipse anisotropy [GDRM08]: $1 - s_2/s_1$ where s_1 and s_2 are the (positive) eigenvalues of the deformation tensor with $s_1 > s_2$. This scalar improves on the standard deformation measure which depends on the number of sides of a bubble.

3.3.3 Image-based Statistical Computation and Visualization

Bubble-scale simulations can be too detailed for observing general foam behavior and T1 events generate large fluctuations in attribute values that hide the overall trends. A good way to smooth out these variations is to calculate the average of attribute values over all time steps, or over a time window before the current time step. This visualization reveals global trends in the data because large fluctuations caused by T1s are eradicated. This results in only small variations between averaged successive time steps.

We take advantage of the graphics card capabilities to calculate a per-pixel average over all time steps for a given attribute. Our solution is faster than a software solution and it is arguably simpler, as the rasterisation of bubble shapes is done in hardware. We support scalar, vector and tensor attributes.

We use three floating point textures stored in three framebuffer objects: $\vec{step}$, $\vec{previous}$ and $\vec{current}$. Even though these are 2D textures, to simplify the presentation we use a 1D index to access a texel. We use index i between square brackets to denote the texel at index i. We use the field access notation from C++ to access component R, G, B or A of a texel. So, $\vec{v}[i].R$ would access component R of texel i of texture v. $\vec{previous}$ and $\vec{current}$ textures store the result for time steps $t < n$ and for $t \leq n$ respectively where n is the current time step. In each RGBA texel we store a sum of attribute values, a count that keeps track of how many values we have rendered over that texel, the minimum and the maximum values. Texture $\vec{step}$ stores an image similar to Figure 3.5 but instead of colors it stores the actual attribute values. In our algorithm, we use two operations between textures: $\oplus$ adds the values stored in R, increments the count stored in G and calculates the minimum and maximum for the values stored in B and A; $\ominus$ subtracts the values stored in R and decrements the count stored in G. Our image-based statistical computation procedure is presented in Algorithm 2.

To visualize the statistical quantities calculated by Algorithm 2, we use a fragment shader to render the average, count, minimum or maximum depending on a variable passed to the shader. We map one of those values to color using a uni-dimensional texture. We only render texels that have the count ($\vec{current}[i].G$) non-zero.

A visualization of the per-pixel average for velocity magnitude and pressure over all time steps is shown in Figure 3.6. Compared with typical visualizations (Figure 2.3 left bottom) our solution is fast, high resolution and can use different color maps and clamping to emphasize features of interest.

Our application can generate an animation of the rolling average up to time step i where $i \in [1, n]$, and n is the number of time steps in the simulation. With this visualization, domain experts can observe, for the first time, the point at which the overall trends are clearly visible and when any transient behavior ends. This allows the determination of the optimal duration of a simulation such that it can capture the dynamics of the underlying phenomena. For instance,

Algorithm 2 Image-based statistical computation

Require: t_{Total} time step where we stop the statistical calculations, t_{Window} calculate the average for the last t_{Window} time steps behind t_{Total}

Ensure: average and count values for t_{Window} time steps behind t_{Total} are stored in $\vec{current}$. Minimum and maximum are calculated for all time steps t_{Total}.

$t = 0$; $t_{CurrentWindow} = 0$

Set every texel of $\vec{current}$ to $(0, 0, \text{max float}, \text{min float})$

while $t < t_{Total}$ **do**

Render the foam for time step t into $\vec{step}$, but store attribute values instead of colors.

$\vec{current} = \vec{previous} \oplus \vec{step}$

$\vec{previous} = \vec{current}$

$t_{CurrentWindow} = t_{CurrentWindow} + 1$

if $t_{CurrentWindow} > t_{Window}$ and $t >= t_{Window}$ **then**

Render the foam for time step $t - t_{CurrentWindow}$ into $\vec{step}$, but store attribute values instead of colors.

$\vec{current} = \vec{previous} \ominus \vec{step}$

$\vec{previous} = \vec{current}$

$t_{CurrentWindow} = t_{CurrentWindow} - 1$

end if

$t = t + 1$

end while

the rolling average animation (Figure 3.6) shows that, for the constriction dataset, the calculated averages converge around time step 400, which means that very little change can be observed in the calculated averages beyond this time step.

We use the time-average computation engine to compute an average over a time window of a tensor field such as the bubble deformation (Section 3.3.2) or of a vector field such as the bubble velocity, defined as the motion of the center of mass. We average tensors by averaging over time each array used to store a tensor at a given position. We average vectors by averaging over time each vector component for the vector at a given position. We visualize the tensor field using glyphs and the vector field using glyphs and streamlines.

A visualization for the field produced by the average computation is done by dividing the field into a Cartesian grid. The center of each square tile is sampled and the attribute value for that texel is displayed. The position and resolution of the grid are user adjustable parameters, which is useful for sampling the field at different positions. Other sampling strategies for the tile are possible, for instance taking an average of all texels in the square.

The tensor is visualized using an ellipse (Figure 3.7) that has each axis length and direction given by the deformation tensor's eigenvalue and eigenvector. The vector is visualized using an arrow with length proportional to the vector magnitude (Figure 3.8), with an arrow with fixed length or with streamlines (Figure 5.4).

Velocity vectors have a wide range of values because the velocity of bubbles involved in topological changes can be much larger than the average velocity of the flow. To address this issue, the arrow length is clamped to a maximum value with clamping shown in a *length bar*.

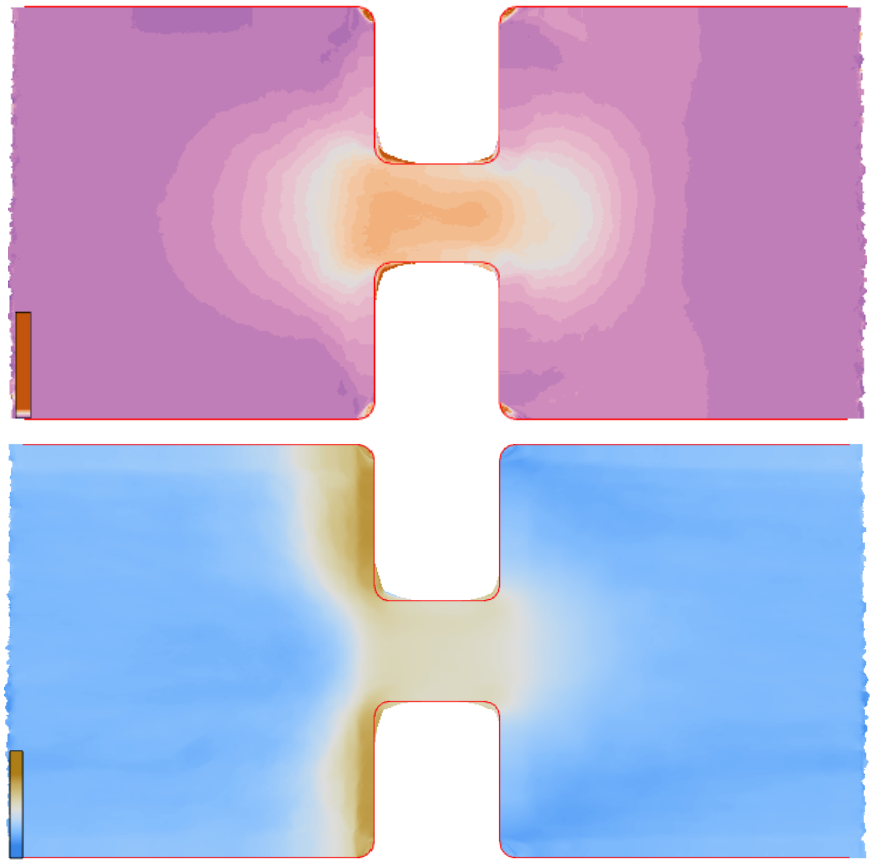


Figure 3.6: Visualization of the per-pixel average of attribute values for all time steps of the simulation. We show the average of velocity magnitude (top) and pressure (bottom). The velocity is color-mapped using purple-orange divergent color map. Pressure is color-mapped using a blue-tan divergent color map (constriction simulation). Compare with Figure 2.3.

The height of the length bar encodes the maximum vector magnitude while the horizontal line inside the bar shows the clamping value (Figure 3.8).

3.3.3.1 Temporal-Averaging of Bubble Attributes around Moving Objects

For foam simulations that include moving objects, we are interested in the forces that determine objects' behavior. These forces are determined by properties of the bubbles adjacent to the objects. However, examining bubble attributes around objects for every time step is not always the best option. There is too much detail and bubble attribute values have large fluctuations caused by topological changes. To address this issue, we compute a temporal average of attribute values *around* the dynamic objects. A statistical calculation using a fixed world coordinate system (foam is stationary and the discs descend through it) would not produce useful results. To explain this (Figure 3.9), let's say that we are interested in calculating the average

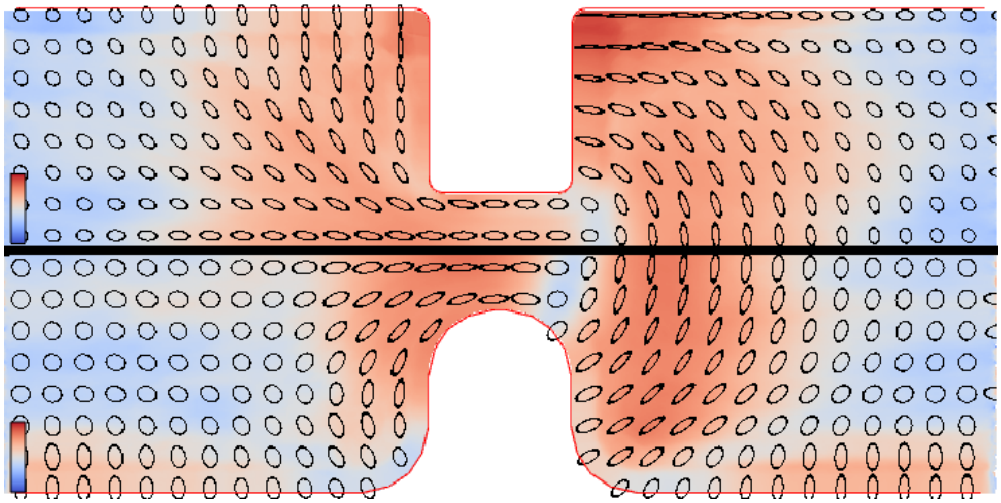


Figure 3.7: We show the square (top) and rounded (bottom) constriction simulations. Foam flows from left to right. Deformation size and direction is displayed with ellipses, deformation size is also color-mapped (with red for high and blue for low deformation). An average over the entire duration of the simulations is displayed. Rounding the corners of the constriction results in reduced elastic deformation of the foam (top versus bottom). In both simulations, there is an area where bubbles are not deformed (ellipses become circular) just downstream from the constriction.

in an area (small circle) situated at three o'clock adjacent to a disc (big circle) that descends through foam (rectangle) due to gravity. The disc is at two different positions p and q at time steps i and k. The interesting area relative to the disc also descends, so it is at positions b_{pi} and b_{qk}. Using a fixed world coordinate system, the average is calculated by overlapping the two foam images (rectangles) and then averaging the corresponding pixels in each image. This computation yields an incorrect formula for the area adjacent to the disc: $(b_{qi} + b_{qk})/2$. By using a fixed disc coordinate system (the disc is stationary and the foam flows around it), the average calculation changes to $(b_{pi} + b_{qk})/2$, the correct result.

We support simulations with objects that undergo general transformations, computing average around two objects and providing the *show rotation* feature that provides the context for dynamic objects. We keep fixed the coordinate system attached to the object. We render the simulation data for a time step in a floating point texture D_t such that each texel covered by a bubble stores the bubble's attribute value. We compute a texture S as the sum of textures D_t for a time window behind the current time-step. Before the texture for the current time step D_t is added to the current sum of textures S, we transform it (translate and/or rotate) so that the object coordinate system stays fixed. As an example, Figure 3.10a shows the *sedimenting-discs* simulation with the two discs starting side-by-side in position 1. The user chooses to show an average around both discs, expecting that the two discs move as a system. The right disc is fixed for the average, and the left disc specifies the rotation of the two discs. At $t = 50$, the discs reach position 2, where both discs descend and the disc on the left starts the rotation

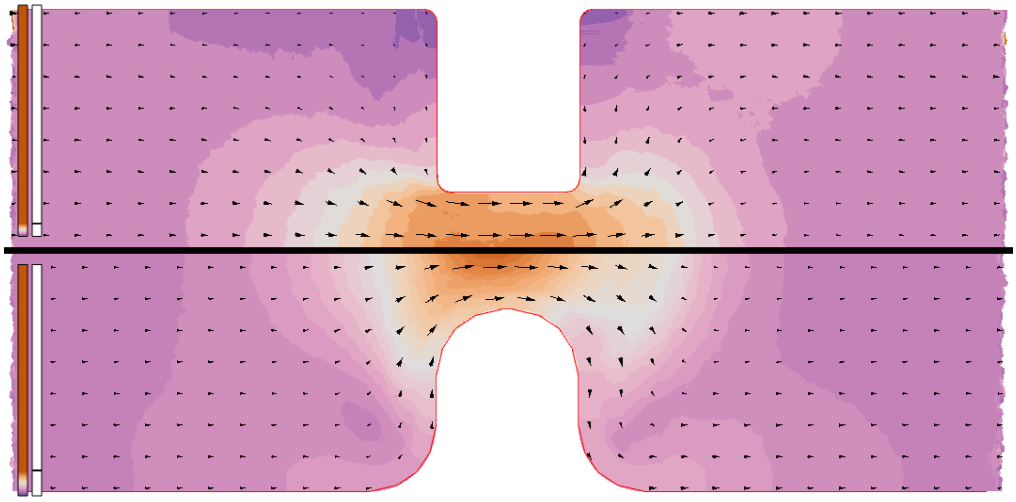


Figure 3.8: We show the square (top) and rounded (bottom) constriction simulations. Velocity is shown with glyphs, velocity magnitude is color-mapped and it is mapped to the size of the glyph. An average over the entire duration of the simulations is displayed. Rounding the corners of the constriction results in the disappearance of the areas of stagnated bubbles visible in the top corners of the square-constriction as dark purple regions. Note that the color bar shows mostly the color associated with high values (orange) because of clamping.

around the disc on the right. Before the texture for the current time step D_t is added to the current sum of textures S, it is transformed so that the coordinate systems in positions 1 and 2 overlap. Note that when averaging tensor and vector attributes, those have to be rotated with the same rotation angle before they are stored in the texture D_t.

Figure 3.10b top shows the average around the two discs for time steps 80 to 99. While this image has the correct foam properties around the two objects, it does not contain information about the position of the two objects in space. To address this we offer the *show rotation* option which rotates the computed average so that it matches the actual position of the two objects in space (Figure 3.10b bottom). Contrast Figure 3.10b bottom with Figure 5.2-middle, where the scalar average is not correct for the rotating disc because scalars are averaged only around the top disc.

3.3.3.2 Sliding Time Window

Calculating the average over all time steps works well for the foam flow through a constriction. However that is not the case for the sedimenting discs simulation, because of the important transient state the discs go through. The two discs are initially side by side (Figure 2.3 middle, 5.2 top), they interact by rotating about one another (Figure 5.2 middle) and they reach a steady state when a line connecting their centers becomes parallel to gravity (Figure 5.2 bottom).

We provide a *sliding time window* user option which allows the calculation of average only for a specified number of time steps behind the current time step. This allows the study of

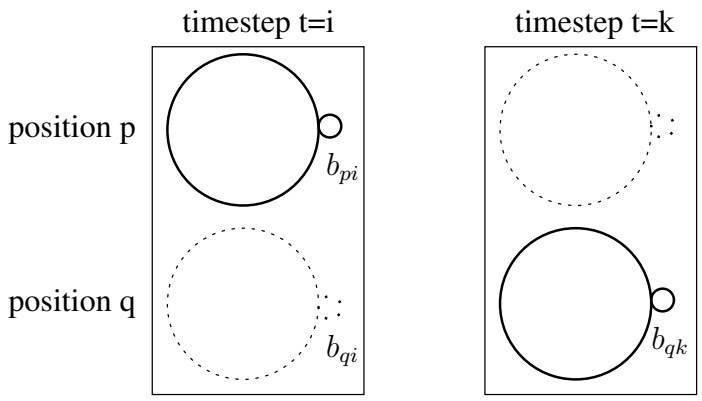


Figure 3.9: Average computation for an area (small circle) around a sedimenting disc (big circle) descending through foam (rectangle). Future (for $t = i$) and past (for $t = k$) disc positions are represented with a dashed line. For a fixed world coordinate system (the foam is stationary and the disc falls through it), the average computation is incorrect $(b_{qi} + b_{qk})/2$). For a fixed disc coordinate system (the disc is stationary and the foam flows around it), the average computation is correct $((b_{pi} + b_{qk})/2)$.

the transient state the two discs go through while maintaining the advantages that averaging of values provide. The length of the sliding time window is a user supplied parameter that depends on the duration (number of time steps) of the transient state.

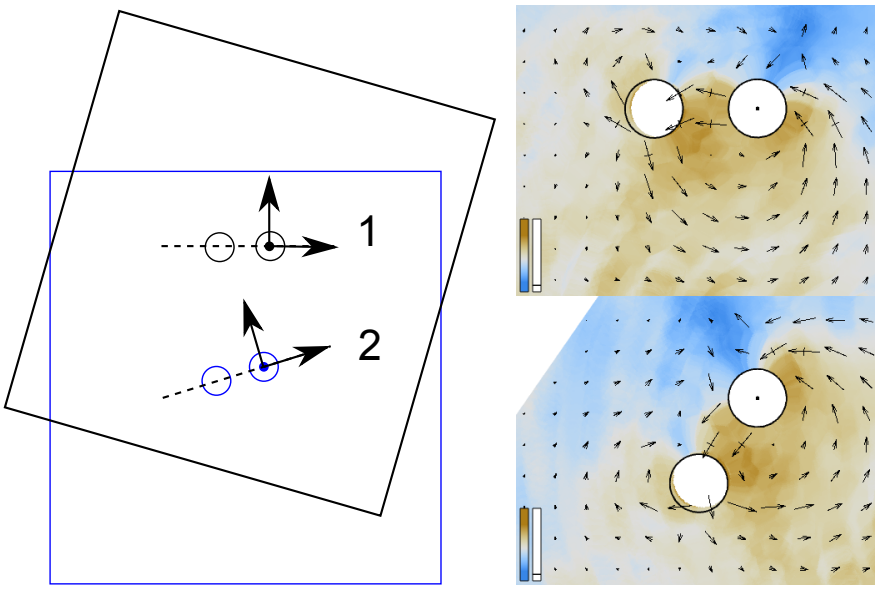
3.3.3.3 Domain Histogram

We can use bubble selection by attribute value (Section 3.3.10) and the count computed together with other statistical values to answer the following question: Where in the domain do the most bubbles with attributes in a certain range of values occur? For instance, Figure 3.11 shows where most bubbles with negative velocity along X occur. Those areas match well with the location of the T1s.

Our novel visualization solution led to insights into the following research questions: Why is a terminal separation of roughly two bubble diameters attained between the two discs once they have rotated about one another into a stable orientation? Why do discs drift laterally as they sediment?

3.3.4 Force Visualization

For the sedimenting-discs simulation, the network and pressure forces acting on each disc are saved by the simulation code. T1 events cause large fluctuations in the value and orientation of these forces, which can make analysis of the simulation data more difficult. To reduce the distraction of the fluctuations, we offer the possibility to smooth the forces over a time window before the current time step. These calculations are integrated with the image based average calculations (Section 3.3.3). Each force (instantaneous or average) acting on a disc is



(a) (b)

Figure 3.10: Average around dynamic objects interacting with foam. (a) The two discs start in position 1, and reach position 2 at $t = 50$. To compute the average around the two discs, the texture for the current time step D_t is transformed before it is added to the sum S such that the coordinate system at position 2 overlaps the coordinate system at position 1. (b) Average around two discs for $t = 99$ and a time window of 19. We show the two objects without (top) and with (bottom) the *show rotation* feature.

represented as a line segment that starts in the center of the disc and has length proportional to the magnitude of the force.

3.3.5 Visualization of Bubble Paths

Visualization of bubble paths complements the image-based statistical computation and visualizations by providing information about the trajectory of individual bubbles in the simulation. The paths are a useful way to compare simulation with experiment. They also provide insight into the overall behavior of the foam. A bubble path is determined by connecting the center of bubbles with the same ID in consecutive time steps. When a bubble exits the domain through a boundary edge, it enters the domain through the opposite boundary edge. Our software computes the correct path in this case.

Despite the problems associated with overlapping bubble paths, tracing all paths simultaneously still conveys useful information about foam behavior. Looking at Figure 3.12 top, gaps where no bubble traverses reveal themselves. This tells us that when bubbles touch a wall they have a strong tendency to remain attached to it.

While the path visualization for all bubbles presents the overall behavior of the foam, overlapping paths prevent tracking the path of individual bubbles. We offer three solutions to solve

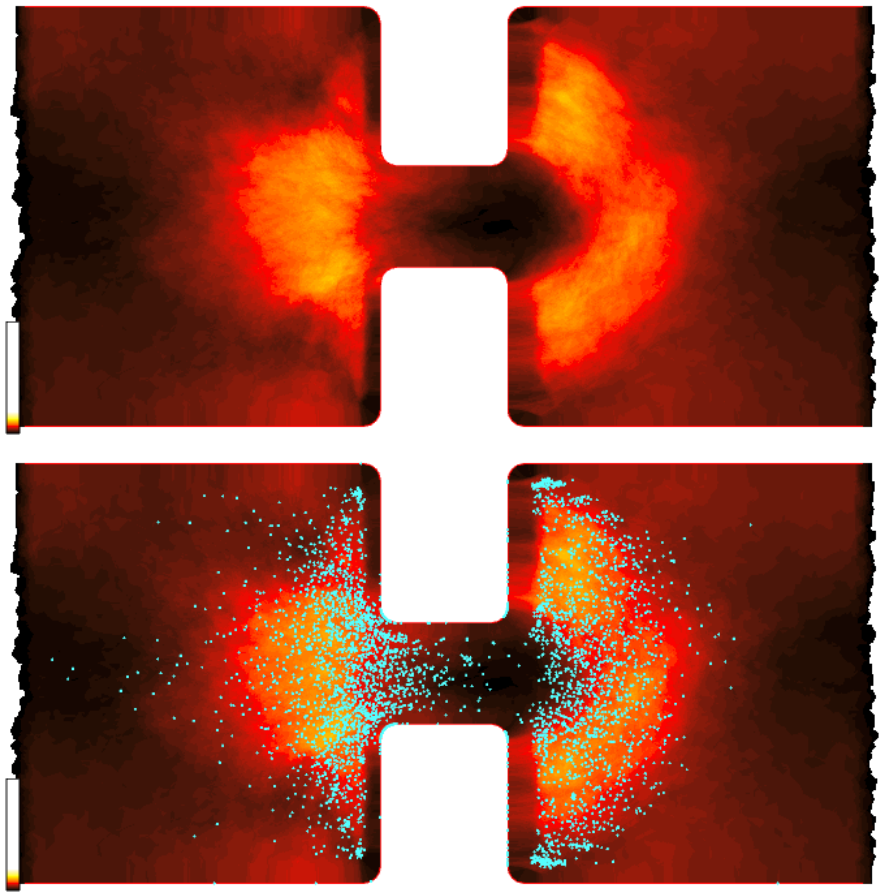


Figure 3.11: Where do most bubbles with negative velocity along X occur? The top image shows the number of times a bubble with negative velocity along X covers a pixel. The range of the color map is 0 to the number of time steps 1000. The bottom image shows the same information overlayed with the location of T1s shown as cyan dots.

this problem. We select paths based on location (Figure 3.13). We can also select paths that include bubbles with attribute values in an interval specified in the histogram view. This way, we can observe not only the time step and bubbles that have certain values but the entire evolution of the selected bubbles. For instance, Figure 3.12 bottom image shows the bubble paths for bubbles that reach at least 80% of the maximum elongation. A bubble's position when it is highly elongated is colored red and the rest of its path is semi-transparent gray. This visualization shows clearly that the bubbles are most elongated as they enter and after they exit the constriction. This corresponds well with regions where the velocity gradient is large (Figure 3.6 top) and with the location of T1s (Figure 3.11 bottom).

Three-dimensional bubble paths alleviate the problem of overlapping paths in 2D through lighting effects. They map time to height and enable multi-variate visualizations. We design 3D bubble paths by assigning each time step to a corresponding height. Consecutive time

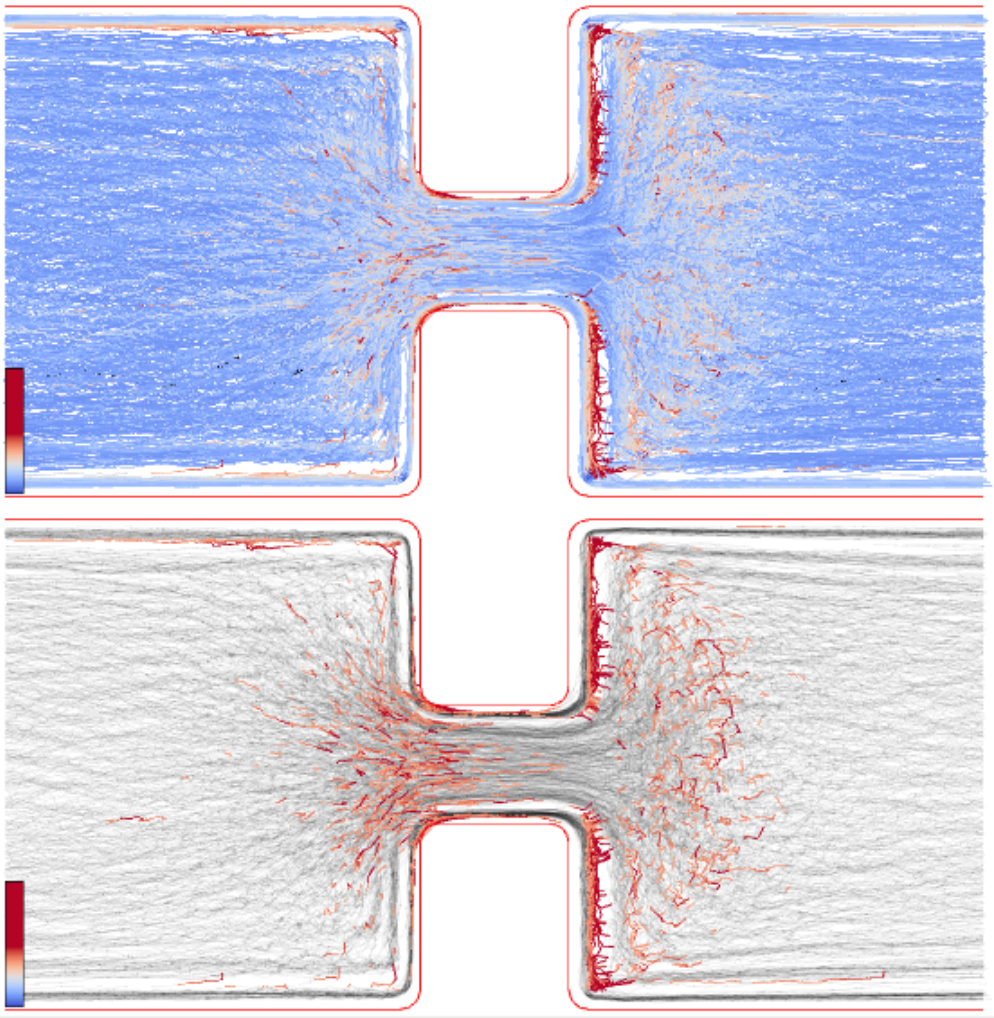


Figure 3.12: Visualization of bubble paths. Path color is mapped to elongation. We show: (top) bubble paths along all time steps for all bubbles in the simulation and (bottom) trajectories that contain bubbles with elongation at least 80% of the maximum elongation (constriction simulation). We use a focus+context approach where the selections are color-coded and the context paths are semi-transparent gray-scale.

steps are assigned to adjacent heights, and all heights are equally spaced. Time steps 0, 1, 2, ... are assigned to heights 0, h, $2h$, ..., where h is a user specified parameter. Velocity is implicitly depicted in this representation. Paths of bubbles which are stationary will appear as vertical, while the higher the velocity of bubbles the closer to the horizontal their paths will appear. Paths can be color-mapped to any other attribute enabling comparison of velocity to other attributes.

We select bubbles upstream of the constriction (Figure 3.13 bottom-left) and we visualize their paths using the 3D bubble paths visualization (Figure 3.13 top). Two important behaviors

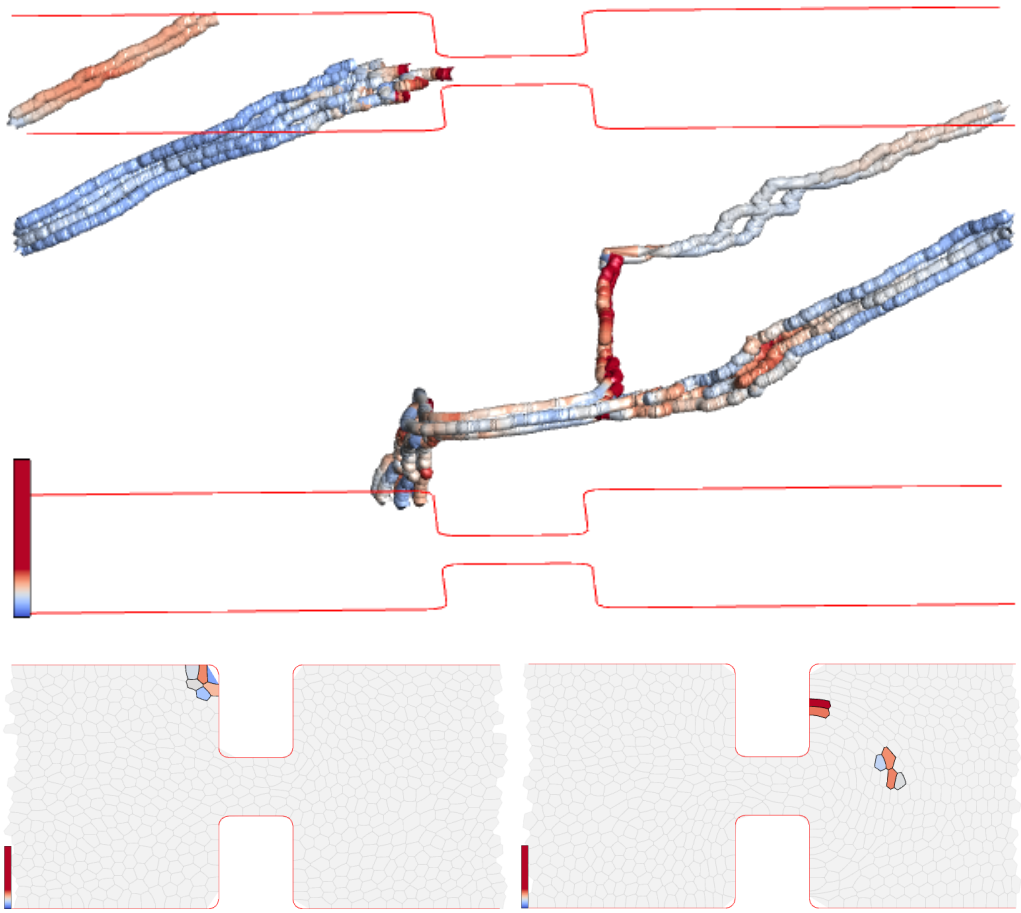


Figure 3.13: 3D bubble paths. Bubble and path color is mapped to elongation. We show visualizations for bubbles that start in the corner upstream of the constriction: (top) 3D bubble paths and (bottom) time step visualizations (t=0, t=361). Vertical 3D bubble paths correspond to bubbles that do not move in the X direction. The two red (high elongation) bubbles in the bottom-right view are attached to the wall. The length of time they stay there is displayed as a vertical path in the top view.

stand out. First, long straight, vertical paths that correspond to bubbles that have zero velocity along the X direction. Those indicate bubbles that are in contact with the wall. As these bubbles have high elongation, they are colored red. Second, we notice long straight edges associated with sharp path angles. These indicate topological changes (T1s).

Of course rendering too many 3D bubble paths causes occlusion. Thus we can filter bubble paths based on any of their attributes such as velocity magnitude, pressure and elongation.

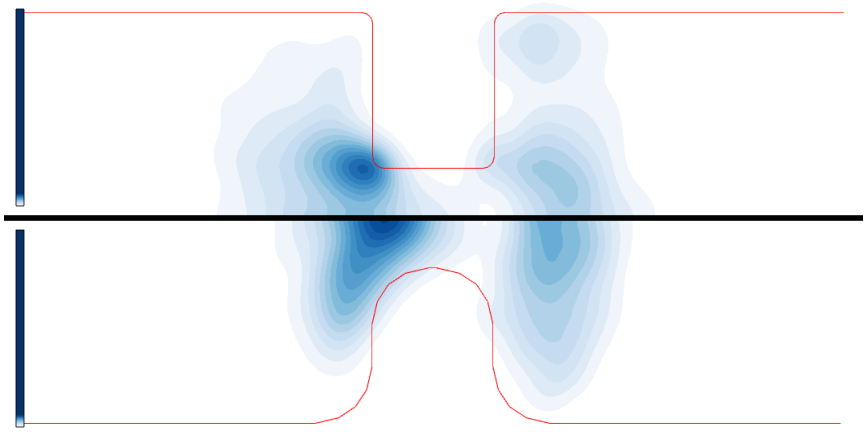


Figure 3.14: Comparison of topological changes KDE. We show the square (top) and rounded (bottom) constriction simulations using the *two halves view*, foam flows from left to right. Rounding the corners of the constriction results in the upstream region of concentrated topological changes moving towards the center of the constriction and downstream. Note as well a region of topological changes in the downstream corners of the square-constriction.

3.3.6 Topological Changes Kernel Density Estimate (KDE)

Topological changes, in which bubbles change neighbors, show plasticity in foam. Domain experts expect that their distribution will be an important tool for distinguishing and validating simulations. Simply rendering the position of each topological change suffers from overplotting, so it may paint a misleading picture of the real distribution (see Figure 3.11-bottom). We adopt the method of Daae Lampe and Hauser [DLH11] to compute a KDE for topological changes (Figures 3.14, 5.5b and 5.11). While traditional histograms show similar information and are straightforward to implement they have drawbacks which may prove important depending on the context, including the discretization of data into bins (which may introduce aliasing effects) and the fact that the appearance of the histogram may depend on the choice of origin for the histogram bins [DLH11]. Kernel-based methods for computing the probability density estimate eliminate these drawbacks.

In foam simulation data, each topological change has two properties specifying when and where the topological change occurred. For each topological change, we place a Gaussian at the T1's position and we average these together. The KDE is computed by dividing by the number of time steps. The standard deviation for the Gaussian is a user defined parameter which determines the amount of detail that is visible in the final visualization.

3.3.7 Multiple Linked-Views

Foam scientists wish to understand what triggers certain behavior in foam simulations, so the ability to see different attributes at the same time and to understand how different attributes relate to one another is very important. We provide up to four different linked-views [Rob07, KLM+08], which can depict any of the following views: a time step visualization, an average

visualization or a bubble path visualization. For maximum flexibility, each view can depict a different attribute, uses its own color-bar and can show the navigation context. Additionally, a histogram view for one of the attributes can be displayed. The histogram is used for specifying a selection criteria on an attribute value. To set up optimal views to analyze data, users can copy viewing transformations and also copy the color mapping between views depicting the same attribute.

We provide two *connection operations* [WGK10]: a time connection and linked-selection connection. Each view is linked to the same time step, as foam scientists want to understand what gives rise to certain effects when they are studying foam behavior. Linked-selection works by showing how data selected in one view appears in other views. This is used to see, for instance, the elongation of high pressure bubbles or both pressure and elongation for bubbles involved in a T1 event.

3.3.8 Histograms

We provide both a histogram of bubble attribute values over one time step and over all time steps. To facilitate data analysis, our histogram is configurable. The user can choose a maximum height, logarithmic or linear height scale and uni-color or color-coded display. Histograms are also used in selection and filtering of data based on attribute value and in color-map clamping used for selecting features of interest in the data. These interactions are described in detail in Section 3.3.10.

3.3.9 Comparative Visualization

Foam scientists often generate related simulations to study how varying just one of the many possible parameters affects the result. Examples of simulation parameters include foam container properties, foam attributes or the properties of objects interacting with foam. We modified FoamVis to enable loading and visualizing several simulations datasets at the same time. This feature is essential for comparing related simulations. We present the *two halves* view and the *linked time with event synchronization* features which address two orthogonal challenges in comparing simulations: space and time. We use the reflection feature and the force difference to facilitate the comparison of two datasets.

3.3.9.1 The two halves view

The *two halves* view facilitates visual comparison of two related foam simulations (Figure 3.7, 3.8, 3.14). It visualizes related simulations that are assumed to be symmetric with respect to one of the main axes. While the same information can be gathered by examining the two simulations in different views, the *two halves* view may facilitate analysis as images to be compared are closer together and it is useful for presentation as it saves space. This type of visualization was previously performed manually by domain experts.

3.3.9.2 The reflection (mirroring) feature

A sedimenting ellipse can rotate clockwise or counterclockwise depending on the initial arrangement of the surrounding bubbles. Similarly, for the interacting discs, the left disc can move around the right disc or vice-versa. Domain experts would like to better compare datasets that have mirrored features such as a sedimenting ellipse or sedimenting discs that rotate in different directions. To address this requirement, we provide a user option that reflects a view about a vertical (or horizontal) axis that passes through the middle of simulation bounding box.

3.3.9.3 Linked Time with Event Synchronization

In a simulation that involves objects interacting with foam, the object's movement in the simulation is controlled by an *effective time scale*, which specifies how much an object is moved (in the direction of the resultant force) at each time step. This parameter may be different for different simulations which means that objects with similar behavior may move at different speeds. Even for simulations with the same effective time scale, we want a similar event in both simulations to happen at the same time so that behavior up to that event can be compared and analyzed together. Examples include comparing two constriction simulations with different flow rates or comparing the sedimenting discs with the sedimenting ellipse simulations. When comparing the sedimenting discs with the sedimenting ellipse simulations, the ellipse and the discs start in similar configurations. The main axis of the ellipse and the line connecting the center of the two discs are horizontal. We want the ellipse and the discs to reach intermediate configurations and the stable configuration at the same time. These configurations are defined in terms of the angle that major axis of the ellipse and the line connecting the centers of the two discs make with a horizontal line. For instance, the intermediate configurations could be defined as angles: 0°, 30°, 60°, and 90° which means that we want both the ellipse and the sedimenting discs reach these orientations at the same time.

The *linked time with event synchronization* addresses these requirements. This technique allows the user to specify time steps in each simulation when significant events happen. The following requirements hold: (i) all simulations have the same set of events; (ii) events are ordered per simulation based on time of occurrence; (iii) corresponding events in different simulations have the same event index. This technique ensures that corresponding events in different simulations are shown at the same time. For each time interval v_{ij} before an event i, one simulation will run at its normal speed (the simulation with the maximum time interval before event i denoted v_{iL}), all other simulations will be "slowed down" (v_{ij} simulation steps will be displayed in v_{iL} time steps). This results in event i being shown simultaneously in all simulations. If we were to "speed up" rather than "slow down" simulations this would result in skipped time steps which in turn results in lost precision. Simulations run at normal speed for the interval after the last event. We formalize our technique next.

Multiple-linked views are used to show a different simulation in each view. Let us assume that the first time step is 0 and let t_j be a time step in simulation j where $0 \leq j < s$, and s is the number of simulations. The time for all linked views is specified using a common *linked time* t_L. The linked time is converted to simulation time t_j which is used to load the specified

simulation time step. FoamVis uses the following default setup for linked time t_L: one time step in the first simulation corresponds to one time step in every other simulation. That means that, by default, all simulations run at their default speed. This default setup is modified if the user desires to examine the context of related events in different simulations.

To specify events of interest the user unlinks the time parameter for the multiple views. After this operation, time can be changed independently for each view. The user specifies times t_{ij} at which event i occurs in simulation j ($0 \leq i < n$, n is the number of events and $0 \leq j < s$, s is the number of simulations), then the user links the time in the s views. Let t_L be the linked time. We analyze how the linked time t_L is converted to simulation time t_j for simulation j.

Let v_{ij} be the time interval between events $i-1$ and i for simulation j, if $i > 0$; or the time interval before event i if $i = 0$. We have that

$$v_{ij} = \begin{cases} t_{ij} & \text{if } i = 0 \\ t_{ij} - t_{i-1,j} & \text{if } i > 0 \end{cases}$$

Let $v_{iL} = \max_{0 \leq j < s} v_{ij}$, the maximum interval v_{ij} for event i and all simulations j. We denote by r_{ij} the ratio by which we "slow down" each simulation j for time interval before event i. We have that $r_{ij} = v_{iL}/v_{ij}$. In linked time, event i happens at time $t_{iL} = \sum_{k=0}^{i} v_{kL}$.

A simulation time t_j (for simulation j) can be deduced from the linked time t_L:

$$t_j = \begin{cases} \lfloor t_L / r_{0j} \rfloor & \text{if} \quad 0 \leq t_L < t_{0L} \\ t_{k-1,j} + \lfloor (t_L - t_{k-1,L})/r_{kj} \rfloor & \text{if} \quad t_{k-1,L} \leq t_L < t_{kL}, \\ & \quad 0 < k < n-1 \\ t_{n-1,j} + (t_L - t_{n-1,L}) & \text{if} \quad t_{n-1,L} \leq t_L \end{cases}$$

Using this approach, related events occur at the same common *linked time* in all s simulations, facilitating the comparison of their temporal context.

The average computation engine computes an average of simulations attributes over a time window behind the current time step. If linked time with event synchronization is used, the time window t_{WL} behind the current time step t_L is specified using the common *linked time*. Earlier we described how to compute the simulation time t_j from the linked time t_L. Similarly, the time window specified that uses the simulation time t_{Wj} (for simulation j, $0 \leq j < s$, s is the total number of simulations) is derived from the time window that uses the linked time t_{WL}. Let us assume that the current time is between events $i-1$ and i, that is $t_{i-1,L} \leq t_L < t_{jL}$. Let us also assume that the beginning of the time window falls between events $k-1$ and k, that is $t_{k-1,L} \leq t_L - t_{WL} + 1 < t_{kL}$. We have that the time window t_{Wj} for simulation j can be computed from the common time window t_{WL} using the following equation:

$$t_{Wj} = \lfloor (t_{kL} - (t_L - t_{WL} + 1))/r_{kj} \rfloor + \sum_{l=k}^{i-1} v_{lj} + \lfloor (t_L - t_{i-1,L})/r_{ij} \rfloor$$

where v_{ij} is the time interval before event i in simulation j and r_{ij} is the ratio by which we slow down simulation j for the interval between events $i-1$ and i.

3.3.9.4 Force Difference and Torque Visualizations

The forces and the torque acting on objects are computed by the simulation code and stored in the simulation data. For the sedimenting discs simulation, the interplay of the network and pressure forces rotate one disc around the other. We provide a user option that displays the difference between the forces acting on the leading disc and forces acting on the trailing disc. This difference allows us to better analyze the causes of the rotation as there is a direct correspondence between the forces displayed on the screen and the movement of the disc (Figure 5.6b right).

The torque $\vec{\tau}$ rotating an object around its center is displayed as a force $\vec{F}$ acting off-center on the object $\vec{\tau} = \vec{r} \times \vec{F}$, where $\vec{r}$ is the displacement vector from the center of the object to the point at which the force is applied. The distance $|\vec{r}|$ is a user-defined parameter, FoamVis calculates the appropriate value of $\vec{F}$ to keep the torque at its given value (Figure 5.6 left).

3.3.10 Interaction

Interaction with the data is an essential feature of our application.

Navigation is used to select a subset of the data to be viewed, the direction of view, and the level of detail [WGK10]. We provide the following navigation operations: rotation around a bounding box center for specifying the direction of view, and translation and scaling for specifying the subset of data and the level of detail. A navigation context (Figure 5.3 left) insures that the user always knows its location and orientation during exploration of the data.

We can **select and/or filter** bubbles and center paths based on three distinct criteria: based on bubble IDs, to enable relating to the simulation files and for debugging purposes; based on location of bubbles (Figure 2.2 and Figure 3.13), to analyze interesting features at certain locations in the data; and based on an interval of attribute values specified using the histogram tool (Figure 3.12 right and Figure 3.15). A composite selection can be specified using both location and attribute values.

Selected bubbles or center paths constitute the focus of our visualization, and the rest of the bubbles or center paths provide the context [Hau06]. The context of the visualization is displayed using user-specified semi-transparency, or it can be hidden altogether. Figure 3.15 shows how selection and filtering is performed using the histogram tool. We select velocity magnitude values greater than 70% of the maximum (using a selection brush). Only six time steps out of 1000 contain bubbles with these velocities and those time steps are indicated on the time slider. The user can easily navigate to those time steps. The upper image shows one of those time steps, with bubbles with velocity greater than 70% in focus.

Encoding operations are variations of graphical entities used in a visualization that emphasize features of interest [WGK10]. We provide encoding operations to change the color map used, to specify the range of values used in the color map and to adjust the opacity of the visualization context. Selection of the interval used in color-mapping is guided by the histogram tool (Figure 3.16). This provides essential information for selecting an interval that reveals features of interest.

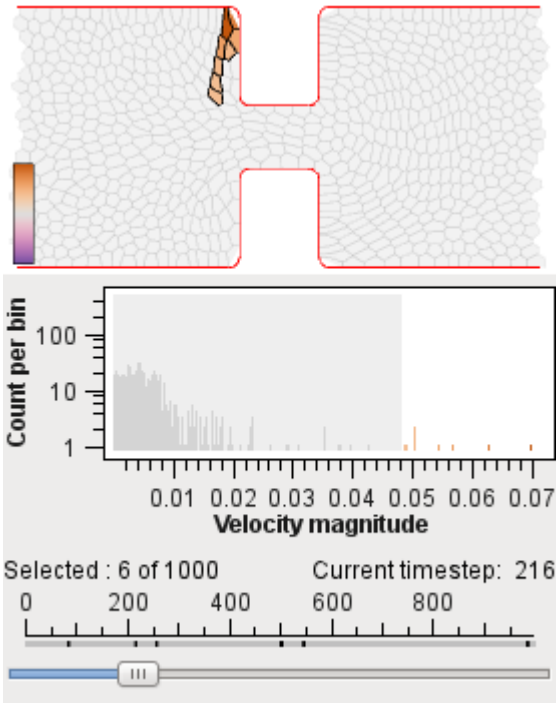


Figure 3.15: Selection and filtering using the histogram tool. We select only bubbles with velocity magnitude values greater than 70% percent of the maximum. Only six time steps out of 1000 contain bubbles with these velocities and those time steps are indicated on the time slider.

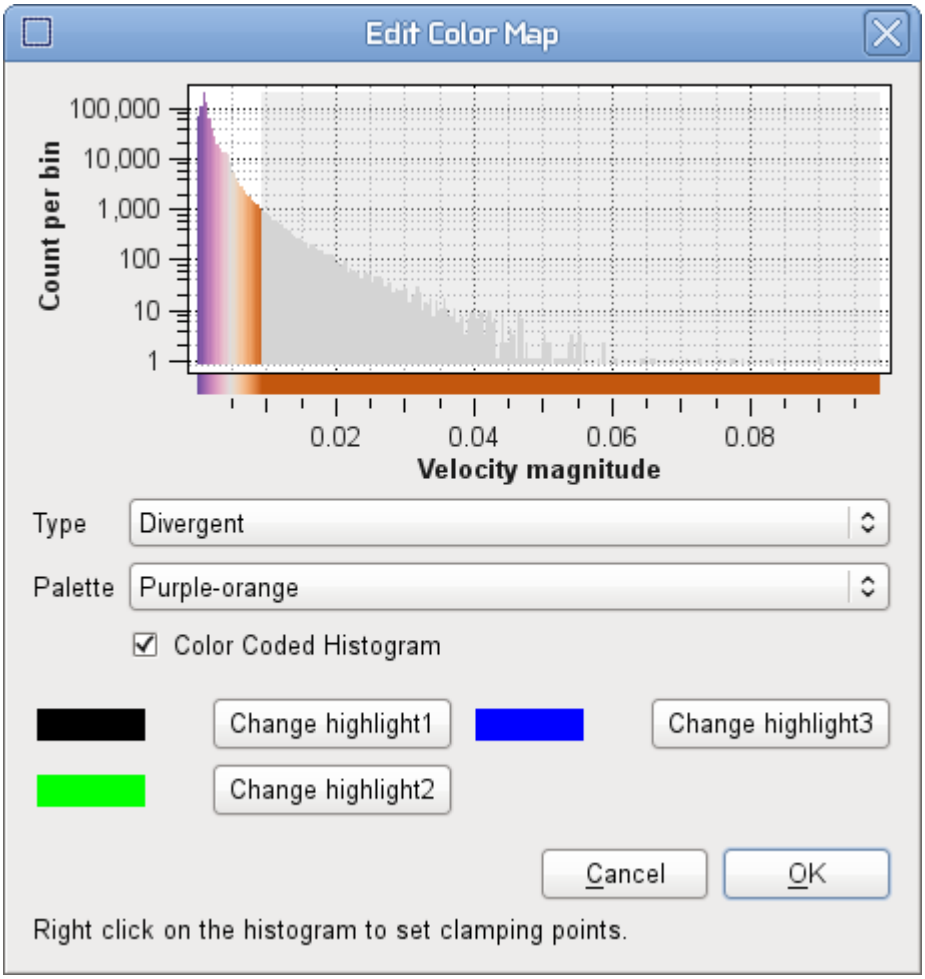


Figure 3.16: Color-map clamping guided by the histogram tool. This is a histogram of the constriction simulation which uses the logarithmic height scale. This histogram shows that relatively few bubbles (determined by T1s) have high velocity.

Chapter 4

Design and Implementation

"A good idea is about ten percent and implementation and hard work, and luck is 90 percent."
– Guy Kawasaki (1951–)[1]

4.1 Overview

In this section we present the structural relationships between FoamVis' main components (Figure 4.1). For this purpose we use a UML 2 components diagram [Bel04]. Briefly, a component, represented in our diagram as a rectangle, is a design unit that is typically implemented using a replaceable module. A component may provide one or more public interfaces, represented with a complete circle at their end (lollipop symbol). Provided interfaces represent services that the component provides to its clients. Similarly, a component may require services from other components. These services are formalized through the required interfaces, represented with a half circle at their end (socket symbol).

FoamVis starts by executing the **Parser** component. This component, uses services from the **UI** component to allow the user to specify the simulations to be analyzed and additional information about the simulations. This is done either through command line or through graphical user interface. Then, the **Parser** parses the specified simulation files, creates an in-memory representation of the simulation data and yields the execution to the Controller module.

The main logic of the program uses the Model-View-Controller design pattern [Mic13]. This pattern separates the data and program state (**Model**), the presentation (**View**), and the interaction with the user (**Controller**) in three different components. This architecture has two main benefits. First, because views are separated from data, several views of the same data can be displayed in the same time. Second, because the **Model** does not depend on the **View** or **Controller** components, changing the user interface or adding new views generally do not affect the **Model**. This results in a more modular and maintainable code and in quicker development cycle.

[1] Guy Kawasaki is a Silicon Valley author, speaker, investor and business adviser. He was one of the Apple employees originally responsible for marketing the Macintosh in 1984. (Wikipedia)

The **Controller** manages the interaction between a user, the **Model** (that stores data and program state) and the views that show foam simulation data.

The **Model** component (Figure 4.2) is composed of three sub-components: **Data** which provides interfaces to create the in-memory representation of the foam simulation data and to read that data; **Settings** which stores the program state; and **Average** which stores and provides interfaces to compute and read time-averages of simulation attributes.

The **View** component provides visualizations for 2D and 3D foam simulation data as well as histograms for scalar attributes.

Each of these logical components contains several implementation files which in turn contain one or several related C++ classes. Logical components (modules), their implementation files and classes, and groups of member functions (member groups) are also documented [Lip13] using Doxygen [vH13]. We are going to refer to the doxygen documentation as we describe the main features of the program and present implementation notes for those features. Here we provide a brief summary of the main components of the program: `Parser`, `Model`, `View` and `Controller`. For brevity we omit the `Display` and `UI` components. We include the name and a brief description for each file part of a component. We use a file name without an extension, to refer to both the interface (`.h`) and the implementation (`.cpp`) files with that name.

The **Parser** parses Surface Evolver `.dmp` files and calls the **Data** component to build a memory representation of the simulation data. It contains the following files:

- AttributeCreator – Create attributes which can be attached to vertices, edges, faces, and bodies.
- AttributeInfo – Information about attributes for vertices, edges, faces, and bodies.
- EvolverData.l – Lexical analyzer for parsing a `.dmp` file produced by Surface Evolver.
- EvolverData.y – Grammar for parsing a `.dmp` file produced by Surface Evolver.
- ExpressionTree – Nodes used in an expression tree built by the parser.
- main.cpp – Drives parsing of SE .dmp files and creates FoamVis main objects.
- NameSemanticValue – Tuple (name, type, value) used for a vertex, edge, face, and body attribute.
- ParsingData – Stores data used during the parsing such as identifiers, variables and functions.
- ParsingDriver – Drives parsing and scanning.
- ParsingEnums – Enumerations used for parsing.

The **Controller** manages the interaction between a user, the **Model** (that stores data and program state) and the views that show foam simulation data. This component contains one implementation file:

- MainWindow – Stores the OpenGL, VTK and, Histogram widgets, implements the Interface and manages the interaction between a user, the **Model** and the views.

The **Data** component creates, processes and stores foam simulation data. It contains the following implementation files:

- AdjacentBody – Keeps track of all bodies a face is part of.
- AdjacentOrientedFace – Keeps track of all faces an edge is part of.
- ApproximationEdge – Curved edge approximated with a sequence of points (Figure 4.3).
- Attribute – Attribute that can be attached to vertices, edges, faces and bodies.
- Body – A bubble.
- BodyAlongTime – A bubble path.
- ConstraintEdge – Edge on a constraint approximated with a sequence of points (Figure 4.3).
- DataProperties – Basic properties of the simulation data such as dimensions and if edges are quadratic or not.
- Edge – Part of a bubble face, stores a begin and an end vertex (Figure 4.3).
- Element – Base class for Vertex, Edge, Face and Body. Stores a vector of attribute (Figure 4.3)
- Face – A bubble is represented as a list of faces, a face is a oriented list of edges.
- Foam – Stores information about a time step in a foam simulation.
- ForceOneObject – Forces and torque acting on one object.
- ObjectPosition – Stores an object interacting with foam position and rotation
- OOBox – An oblique bounding box used for storing a torus original domain.
- OrientedEdge – An oriented edge. Allows using an Edge in direct or reversed order.
- OrientedElement – Base class for OrientedFace and OrientedEdge. Allows using a Face or Edge in direct or reversed order.
- OrientedFace – An oriented face. Allows using a Face in direct or reversed order.
- ProcessBodyTorus – Processing done to “unwrap” bodies in torus model.
- QuadraticEdge – Quadratic edge approximated with a sequence of points (Figure 4.3).
- Simulation – A time-dependent foam simulation.

- T1 – A topological change.
- Vertex – Element used to specify edges. An edge has at least two vertices, begin and end. A quadratic edge has a middle vertex as well.

The **Settings** component stores and provides access to program state. This component is composed from the the following files:

- BodySelector – Functors that specify selected bubbles.
- Settings – Settings that apply to all views.
- ViewSettings – Settings that apply to one view.

The **Average** component computes time-average of simulation attributes. It contains:

- AttributeAverages – Computes the average for several attributes in a view. Base class for AttributeAverages2D and AttributeAverages3D (Figure 4.4).
- AttributeAverages2D – Computes the average for several attributes in a 2D view. Casts the computed averages to the proper 2D types (Figure 4.4).
- AttributeAverages3D – Computes the average for several attributes in a 3D view. Casts the computed averages to the proper 3D types (Figure 4.4).
- Average – Computes a time-average of a foam attribute. Base class for 2D and 3D time-average computation classes (Figure 4.4).
- AverageInterface – Interface for computing a time-average of a simulation attribute (Figure 4.4).
- AverageShaders – Shaders used for computing a pixel-based time-average of attributes.
- ForceAverage – Time-average for forces acting on objects interacting with foam (Figure 4.4).
- ImageBasedAverage – Calculates a pixel-based time-average of 2D foam using shaders (Figure 4.4).
- PropertySetter – Sends an attribute value to the graphics card (Figure 4.4).
- RegularGridAverage – Time-average for a 3D regular grid (Figure 4.4).
- ScalarAverage – Computes 2D scalar average (Figure 4.4).
- T1KDE2D – Calculates T1s KDE for a 2D simulation (Figure 4.4).
- TensorAverage – Computes a pixel-based time-average of vector and tensor attributes (Figure 4.4).
- VectorAverage – Computes a pixel-based time-average of vector attributes (Figure 4.4).

- VectorOperation – Math operations for vtkImageData, used for 3D average computation.

The **Model** component consists of the **Data**, **Settings** and **Average** components. It also includes two additional files:

- Base – Simulation data, derived data and, program status.
- DerivedData – Data derived from simulation data such as caches and averages.

The **View** component contains the views for displaying data. It contains the following implementation files:

- AttributeHistogram – A GUI histogram of a scalar attribute useful for one time step and all time steps.
- FoamvisInteractorStyle – Interactor that enables FoamVis style interaction in a VTK [Inc10b] view.
- Histogram – A histogram GUI that allows selection of bins.
- HistogramItem – Implementation of a GUI histogram, modified from Qwt [Bre].
- WidgetBase – Base class for all views: WidgetGl, WidgetVtk, WidgetHistogram.
- WidgetGl – View that displays 2D (and some 3D) foam visualizations using OpenGL.
- WidgetHistogram – View for displaying histograms of scalar values.
- WidgetSave – Widget that knows how to save its display as a JPG file.
- WidgetVtk – View that displays 3D foam visualizations using VTK.

4.2 Parsing and Data Processing

Foam simulation data consists of a list of SE output files, one per time step. A file stores the entire configuration of the simulated foam at a particular time step. Parsing is done using `flex` [fle] and `bison` [gnu] tools using `EvolverData.l` lexical analyzer and `EvolverData.y` grammar. Parsing is run by `Simulation::ParseDMPs` which parses the simulation files, stores the simulation data in memory and performs the additional processing required.

Our tool can read the following optional data that is saved by the simulation code: a list of T1s and the network and pressure forces that act on a body (Section 4.4).

After parsing foam simulation data and creating the corresponding data structures, we perform additional data processing (`Foam::Preprocess` and `Simulation::Preprocess`). First we compact each list of geometric elements as there can be numbering gaps in the list specified in a SE file (`Foam::compact`). Then, if the foam described in the SE file contains periodic boundary conditions (PBC) [sur04, sur08] we unwrap the geometric elements so that we can display the foam (`Foam::unwrap`). Additional processing include calculating each bubble's center of mass(`Foam::calculateBodyCenters`), bounding box and the bounding

box of the foam at each time step (Foam::CalculateBoundingBox and overall (Simulation::CalculateBoundingBox, and calculating statistical quantities such as histogram, minima and maxima for values of attributes (Simulation::calculateStatistics). For 3D foam simulations unstructured simulation data is converted to a regular grid and it is cached in files on disk (Section 4.6).

4.3 Interface

Each dataset consists of a list of data files stored in a folder. The only information about the simulation available without parsing the simulation files is the name of the folder. While this often encodes important parameters of the simulation, their meaning may be cryptic and only known to the scientist that created the simulation. Additionally there is an increasing number of parameters providing additional information about the simulation which is not encoded in the simulation files. To address these issues we create a simulations database and a browsing interface. The simulations database records for each simulation three pieces of information: a simulation name - usually this is the name of the folder that stores the simulation files; a list of labels, each label is used to group simulations based on specific criteria; and simulation specific visualization parameters. The database is stored as a .ini file and is created by the user from a template. The **UI**, Options file contains classes that read options either from the command line or from an .ini file.

The browsing feature (Figure 4.5) presents all grouping labels from the simulation database in a list. When a user selects a label, a list with all simulations tagged by that label is presented. When a user selects a simulation name, a picture of the first time step in the simulation is displayed. The image is saved beforehand so no parsing of simulation files is required. This allows a user to explore existing simulations based on similarity criteria encoded in labels and visually select simulations of interest for individual analysis or comparison. The browsing dialog is implemented by UI, BrowseSimulations class.

FoamVis' main window (Figure 4.6) contains three panels that are used for both visualization and user interaction (Spatial and Information Visualization and Time), and one panel (Interface) that allows the user to specify desired visualizations and visualization parameters. The spatial visualization panel shows multiple views with each view showing a different visualization, a visualization of a different simulation attribute or a visualization of a different simulation. The information visualization panel shows histograms for simulation scalars shown in the spatial visualization panel. The time panel shows the current time step and marks time steps resulting from selections on scalar values. The main window of the application is implemented in **Controller**, MainWindow. This class implements the Interface panel and handles user notifications resulted from user interactions with the panel or with simulation data. The Spatial Visualization panel is implemented by the **View** component. In this component WidgetGl class displays 2D visualizations and 3D attribute and bubble paths visualizations; WidgetVtk class displays 3D attribute time-average and T1s KDE visualizations. The decision to use VTK [Inc10b] rather than plain OpenGL [SWND06] for some of the 3D visualizations was based on the desire to speed-up the development of the application. We believe this was a sound decision which, besides speeding-up development, opened-up a wide range of visualiza-

tion algorithms for adoption into FoamVis.

4.4 Simulation attributes

Scalar bubble attributes include velocity along principal axes, velocity magnitude, edges per face, deformation, pressure, volume and growth rate. Scalar bubble attributes are visualized using color mapping. The user can change the color palette and change the range of scalar values mapped to color through clamping (Section 4.10). Figure 5.5a and Figure 3.7, Figure 4.6 show examples of scalar attributes visualized through color mapping. While domain experts are mostly interested in bubble attributes, in SE attributes can be attached to a body (bubble), face, edge or vertex. Information about predefined attributes that can be attached one of these elements is stored in **Parser**, `AttributesInfoElements` class. New attributes can be defined in a `.dmp` file. The **Parser** calls **Data**,`Foam::AddAttributeInfo` to register a new attribute. The **Parser**,`EvolverData.y` parses a list of attributes on the following grammar rules: `xxx_attribute_list` where `xxx` is vertex, edge, face or body. It creates a list of `NameSemanticValue` objects and it passes them to the `Foam` object for storage.

We visualize bubble velocities using glyphs (2D and 3D) (Figure 5.5a) and streamlines (2D only) (Figure 5.4). We compute the velocity attribute in a processing step after parsing (`Simulation::Preprocess`) in `Simulation::calculateVelocity`. Velocity glyphs are visualized using the **Display** module, `DisplayBodyFunctors` file, `DisplayBodyVelocity` class.

Bubble deformation magnitude and *direction* are important bubble attributes, because they facilitate validation of simulations and provide information about the force acting on a dynamic object in foam. While visual inspection of individual bubbles provides information about foam deformation, this information is not quantified and, more importantly, cannot be averaged to obtain the general foam behavior. To address these issues, we define a bubble deformation measure [LLCD13c] expressed as a tensor. The deformation tensor is visualized using glyphs as shown in Figure 3.7. We compute a deformation scalar and tensor measures in a processing step after parsing: `Foam::CalculateDeformationSimple` and `Foam::CalculateDeformationTensor`. Two-dimensional deformation glyphs are visualized using the **Display** module, `DisplayBodyFunctors` file, `DisplayBodyDeformation` class.

When foam is subjected to stress, bubbles deform (elastic deformation) and move past each other (plastic deformation). Domain experts are interested in the distribution of the plasticity which is indicated by the location of topological changes. A topological change is a neighbor swap between four neighboring bubbles. In a stable configuration, bubble edges meet 3-way at 120° angles. As foam is sheared, bubbles move into an unstable configuration, in which edges meet 4-way, then quickly shift into a stable configuration. Topological changes for the current time step or for all time steps are visualized with glyphs (or spheres) of configurable color and size showing the location of the topological change (Figure 5.4). Topological changes are parsed either from a separate file or from variables inside the `.dmp` file by the overloaded function `Simulation::ParseT1s`. They are stored in the `Simulation` object.

The forces and the torque acting on objects are computed by the simulation code and stored in the simulation data. Each force acting on an object is represented as an arrow that starts in the center of the object and has length proportional to the magnitude of the force.

For the falling discs simulation, the interplay of the network and pressure forces rotate one disc around the other. We provide a user option that displays the difference between the forces acting on the leading disc and forces acting on the trailing disc. This difference allows us to better analyze the causes of the rotation as there is a direct correspondence between the forces displayed on the screen and the movement of the disc (Figure 5.6b right).

The torque τ rotating an object around its center is displayed as a force F acting off-center on the object $\tau = r \times F$, where r is the displacement vector from the center of the object to the point at which the force is applied. The distance $|r|$ is a user-defined parameter, FoamVis calculates the appropriate value of F to keep the torque constant (Figure 5.6b left).

The forces and torques acting on objects are read from the simulation files, from variable names passed as parameters either from command line or from the .ini file. Variable names that store forces and torques are passed as parameters in **Data**, `ForceNamesOneObject` class while the forces and torques are stored in **Data**, `ForceOneObject` in the `Foam` object. Forces are displayed using `ForceAverage::DisplayOneTimeStep`[1] for OpenGL views or using `PipelineAverage3D::createObjectActor` for VTK views.

4.5 Bubble paths

A bubble path is determined by connecting the center of bubbles with the same ID in consecutive time steps. Figure 5.3 shows a pattern of bubbles traversing loops revealed by a bubble paths visualization.

Bubble paths are stored in a processing step after parsing by calling `Simulation::CacheBodiesAlongTime` and they are displayed in **Display**, `DisplayBubblePaths`.

4.6 Time-Average of a Simulation Attribute

Bubble-scale simulations can be too detailed for observing general foam behavior and topological changes generate large fluctuations in attribute values that hide the overall trends. A good way to smooth out these variations is to calculate the average of the simulation attributes over all time steps, or over a time window before the current time step. The time window is a parameter set by the user. We compute the average for the entire simulation (Figure 3.7) if there are no dynamic objects interacting with the foam. In this case, at a high level of detail, there is no difference between different time steps in the simulations. For simulations that include dynamic objects interacting with the foam (Figure 5.5, 5.6) a smaller time window is appropriate, as objects may traverse transient states that have to be analyzed independently.

Time-averages of several 2D foam simulation attributes are stored in `AttributeAverages2D` and are referenced from `WidgetGl`. A pixel-based time-average of one simulation attribute is computed by `ScalarAverage`, `VectorAverage` and `TensorAverage` for scalars, vectors and tensors. Most of this computation is done in the base class `ImageBaseAverage` (Figure 4.4). Displaying an average of attributes is done by the graphics card fragment shader using `AverageInterface::AverageRotateAndDisplay`[1] overwritten for each attribute type.

[1] This function would better fit in the **Display** module, as this would separate the data from its display. We plan address this issue in future work.

For 3D simulations, unstructured grid data is converted to regular grid data and is cached in files inside .foamvis folder using Foam::SaveRegularGrid. This is done in the processing step after parsing Simulation::Preprocess. Time-averages of several 3D foam simulation attributes are stored in AttributeAverages3D and are referenced from WidgetVtk. A time-average for one attribute, for all types of attributes, is computed by RegularGridAverage and displayed using a VTK pipeline created by PipelineAverage3D::createScalarAverageActor and PipelineAverage3D::createVelocityGlyphActor for scalars and respective vectors.

4.7 Topological changes kernel density estimate (KDE)

Topological changes, in which bubbles change neighbors, indicate plasticity in a foam. Domain experts expect that their distribution will be an important tool for validating simulations. Simply rendering the position of each topological change suffers from over-plotting, so it may paint a misleading picture of the real distribution. We compute a KDE for topological changes (Figure 5.5, Figure 5.11).

T1s KDE is computed using the average framework (Figure 4.4) (T1KDE2D or RegularGridAverage classes for 2D or 3D foam simulation). For 2D simulations, for each topological change in a time step, a Gaussian is added to the average using T1KDE2D::writeStepValues. For 3D simulations, a Gaussian determined by a topological change in a time step is returned by Simulation::GetT1KDE. This Gaussian added to the current average in RegularGridAverage::OpStep.

4.8 Histograms

We provide both a histogram of bubble attribute values over one time step and over all time steps. To facilitate data analysis, our histogram is configurable. The user can choose a maximum height, logarithmic or linear height scale and uni-color or color-coded display using HistogramSettings dialog. Histograms are also used in selection and filtering of data based on attribute value and in color-map clamping used for selecting features of interest in the data. These interactions are described in detail in Section 4.10. Histograms are displayed by **View**,WidgetHistogram. Histograms notify the **Controller** when scalar selection has changed using WidgetHistogram::SelectionChanged.

4.9 Multiple linked-views

Foam scientists wish to understand what triggers certain behavior in foam simulations. Foam behavior is determined by many simulations attributes so the ability to see different attributes at the same time and to understand how different attributes relate to one another is very important. At the same time, to understand the influence that simulation parameters have on foam behavior, foam scientists would like to analyze and compare related simulations. Both these requirements are addressed using multiple linked views. We provide up to four different views. For maximum flexibility, each view can depict a different simulation attribute, a different visualization or even a different simulation. Each view uses its own color-bar and can show the

navigation context. Each of the three widgets used to show data (WidgetGl, WidgetVtk and WidgetHistogram) can display up to four views. These three classes are derived from WidgetBase which provides view related functionality; WidgetBase is derived from Base which provides access to data and program status (Figure 4.7).

To set up optimal views to analyze data, users can copy viewing transformations (WidgetXXX::CopyTransformFromSlot where XXX is Gl or Vtk) and color mapping between views depicting the same attribute (MainWindow::CopyColorMapXXX where XXX is Scalar or Velocity).

The *two halves* option facilitates visual comparison of two related foam simulations (Figure 3.7). It visualizes related simulations that are assumed to be symmetric with respect to one of the main axes. While the same information can be gathered by examining the two simulations in different views, the *two halves* view may facilitate analysis as images to be compared are closer together and it is useful for presentation as it saves space. This type of visualization was previously performed manually by domain experts. This option is only available for 2D simulations in WidgetGl. It is set using Settings::SetTwoHalvesView.

We provide three *connection operations* [WGK10] between views: one linked-selection connection and two linked-time connections. The linked-selection connection works by showing data selected in one view in other views. This is used to see, for instance, the elongation of high pressure bubbles or both pressure and elongation for bubbles involved in a topological change. This connection works by copying the selection in one view in any other view using ViewSettings::CopySelection.

The first linked-time connection works by having each view linked to the same time step, as foam scientists want to analyze several attributes at the same time to understand foam behavior influenced by those attributes. The linked-time connection is set to independent time or linked time using Settings::SetTimeLinkage. The second linked-time connection, linked-time with event synchronization, is described next. In simulations that involve dynamic objects interacting with foam, we may want a similar event in both simulations to be visualized at the same time so that behavior up to that event can be compared and analyzed together. When comparing the falling discs with the falling ellipse simulations, the ellipse and the discs start in similar configurations. The main axis of the ellipse and the line connecting the center of the two discs are horizontal. We want the ellipse and the discs to reach intermediate configurations and the stable configuration at the same time. These configurations are defined in terms of the angle that the major axis of the ellipse and the line connecting the centers of the two discs make with gravity. For instance, an angle for the intermediate configuration could be 45° while the angle for the stable configuration is 0°. A new event for the current view and current time is added using Settings::AddLinkedTimeEvent. All views that use linked-time with event synchronization have to have the same number of events. This technique splits simulation times in intervals - an interval before each event and an interval after the last event. For each interval before an event, one simulation will run at its normal speed (the simulation with the longest interval as returned by Settings::GetLinkedTimeMaxInterval), all other simulations will be "slowed down" using Settings::GetLinkedTimeStretch. Simulations will run at normal speed for the time-interval after the last event. Using this approach, related events occur at the same *linked time* in all simulations, facilitating their comparison as well as the comparison of their temporal context. Figure 5.5, 5.6 use linked-time with event synchronization feature. The

complete interface for using the linked-time with event synchronization is in class Settings, member group Time and LinkedTime.

4.10 Interaction

Interaction with the data is an essential feature of our application.

Navigation is used to select a subset of the data to be viewed, the direction of view, and the level of detail [WGK10]. We provide the following navigation operations: rotation around a bounding box center for specifying the direction of view, and translation and scaling for specifying the subset of data and the level of detail. Navigation operations are implemented in the WidgetGl views in mousePressEvent and mouseMoveEvent. These operations are provided by the VTK library in the WidgetVtk views. A navigation context (Figure 5.3 left) ensures that the user always knows its location and orientation during exploration of the data. Focus and context related settings are in ViewSettings, Context view member group.

We can **select and/or filter** bubbles and center paths based on three distinct criteria: based on bubble IDs (WidgetGl::SelectBodiesByIds), to enable data to be related to the simulation files and for debugging purposes; based on location of bubbles (WidgetGl::mousePressEvent and WidgetGl::mouseMoveEvent), to analyze interesting features at certain locations in the data; and based on an interval of attribute values specified using the histogram tool (Figure 4.6) (The histogram sends WidgetHistogram::selectionChanged signal which is handled in MainWindow::SelectionChangedFromHistogram). A composite selection can be specified using both location and attribute values.

Selected bubbles or center paths constitute the focus of our visualization, and the rest of the bubbles or center paths provide the context [Hau06]. The context of the visualization is displayed using user-specified semi-transparency, or it can be hidden altogether.

Encoding operations are variations of graphical entities used in a visualization that emphasize features of interest [WGK10]. We provide encoding operations to change the color map used, to specify the range of values used in the color map and to adjust the opacity of the visualization context. Selection of the interval used in color-mapping is guided by the histogram tool (Figure 3.16) (the implementation is in EditColorMap). This provides essential information for selecting an interval that reveals features of interest.

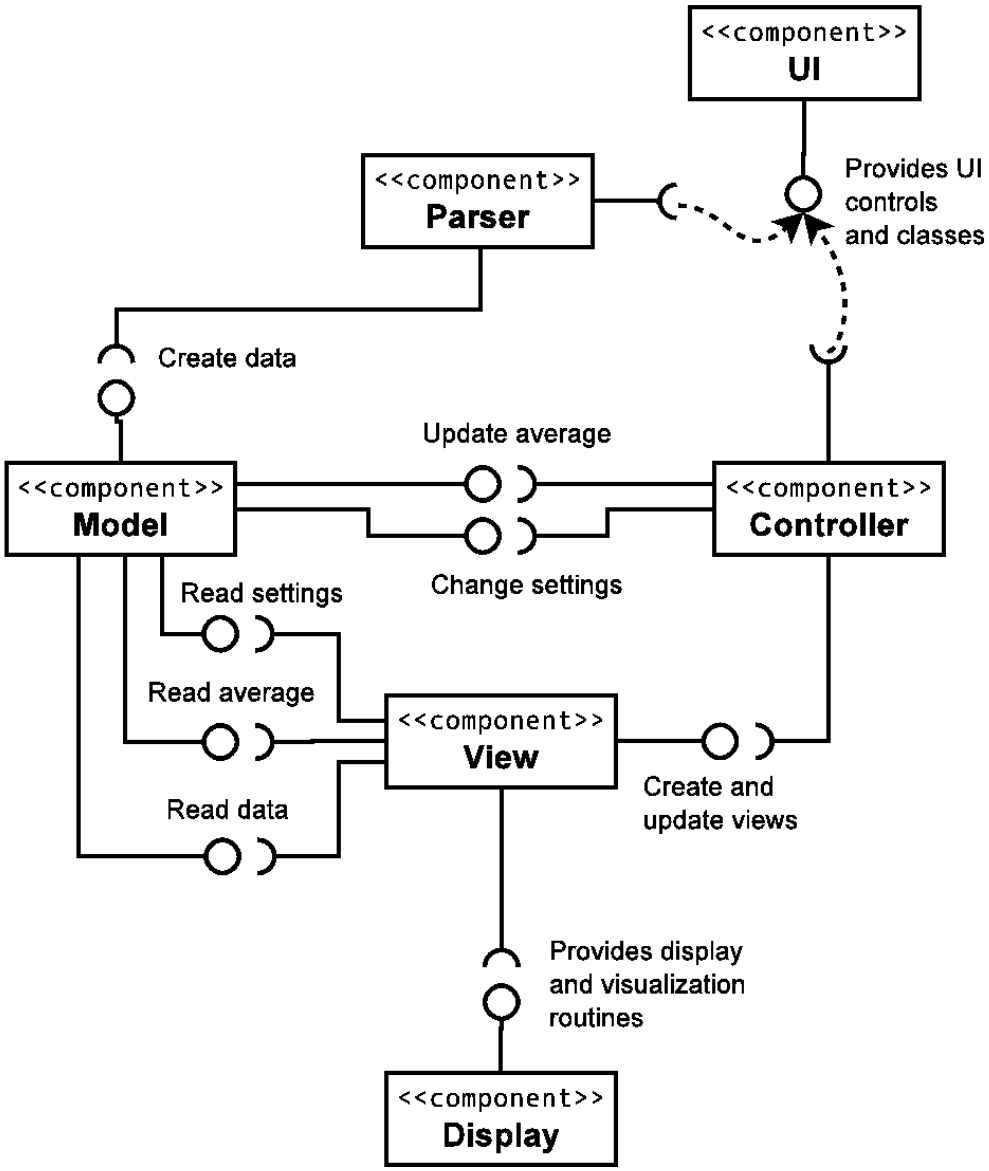


Figure 4.1: FoamVis UML Component Diagram. The **Parser** parses simulation data and stores it in memory. FoamVis uses the Model-View-Controller design pattern to separate the data and program state (**Model**), the presentation (**View**), and the interaction with the user (**Controller**) in three different components. The **UI** provides user interface controls and classes and the **Display** provides display and visualization algorithms.

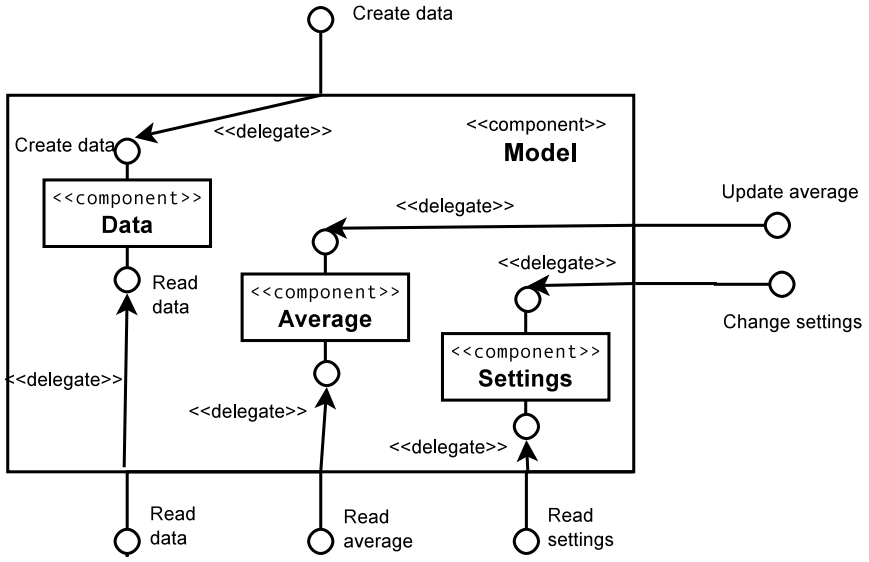


Figure 4.2: The **Model** Component. This component is responsible for storing data and program state. It is composed from three sub-components: **Data** which stores simulation data, **Average** which stores derived time-average of simulation attributes, and **Settings** which stores program state.

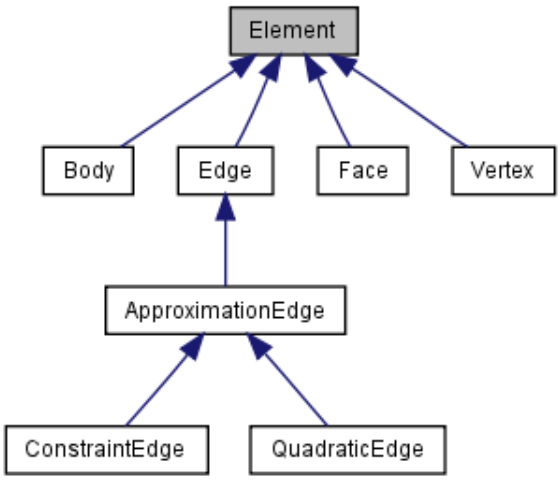


Figure 4.3: `Element` class Inheritance Graph. This class stores a vector of attributes that can be attached to bodies (bubbles), faces, edges, and vertices. This diagram also shows the three types of edges represented in FoamVis: regular (`Edge`) edges that have a begin and an end vertex, quadratic edges (`QuadraticEdge`) that have an additional middle vertex, and constraint edges (`ConstraintEdge`) that are described using a begin vertex, an end vertex and a curve $f(x,y,z)$ on which the edge lies.

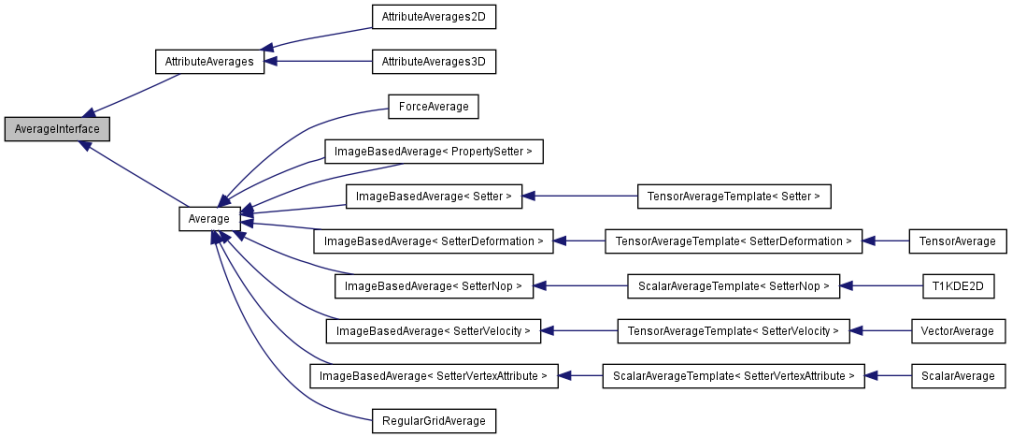


Figure 4.4: `AverageInterface` Inheritance Graph. This class provides the interface for updating an average of attributes. `AttributeAverages` stores an average of many simulation attributes. `Average` provides common computation for averages of forces (`ForceAverage`), 2D simulation attributes (`ImageBasedAverage`) and 3D simulation attributes (`RegularGridAverage`). Two-dimensional averages include scalars (`ScalarAverage`), vectors (`VectorAverage`), tensors (`TensorAverage`) and kernel density estimates (`T1KDE2D`).

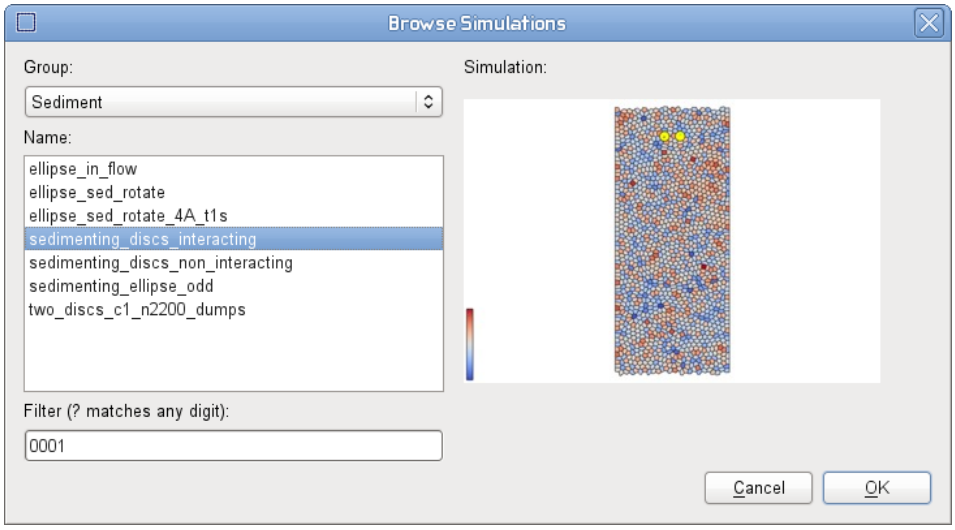


Figure 4.5: The `BrowseSimulations` dialog which allows the user to view related simulations and select simulations of interest for individual analysis or comparison.

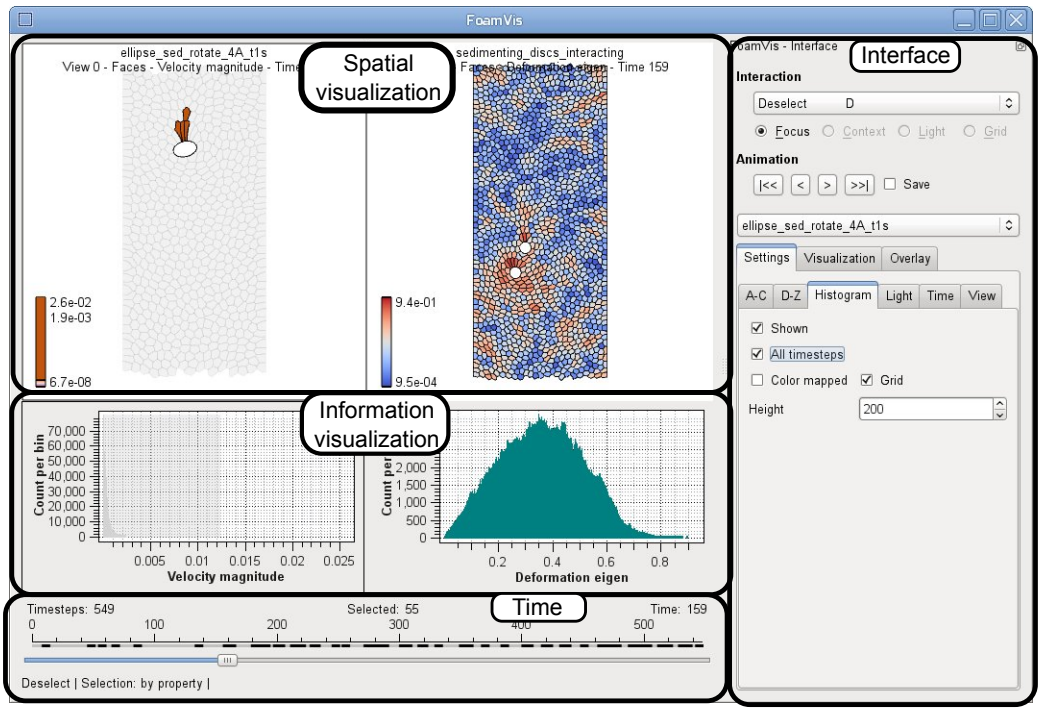


Figure 4.6: FoamVis' `MainWindow` showing the spatial visualization, information visualization, time and interface panels. The spatial visualization panel shows two views: bubble velocity magnitude for a falling ellipse simulation and bubble deformation a falling discs simulation. A selection on velocity magnitude values is performed on the histogram showing this scalar and it is reflected in the spatial visualization and time panels. We can observe in the time panel that only 55 time steps out of 549 contain high velocity bubbles and in the spatial visualization panel we see those bubbles color mapped, the rest of the bubbles are rendered in gray as context information.

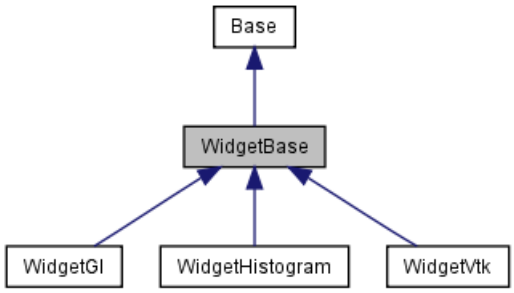


Figure 4.7: `WidgetBase` Inheritance Graph. This class provides functionality common to all views. It inherits from `Base` which stores simulation data and program status. `WidgetGl` displays views rendered with OpenGl, `WidgetVtk` displays views rendered with VTK, and `WidgetHistogram` displays histograms.

Chapter 5

Results

> "Painting does what we cannot do - it brings a three-dimensional world into a two-dimensional plane."
> – Chuck Jones (1912-2002)[1]

Our tool is developed in close collaboration with the foam scientists who design and run these simulations. We present case studies describing the way in which they use FoamVis, and the insights that they gain.

5.1 High velocity bubbles outside the constrictions are caused by T1 events

An unexpected behavior of the foam detected in the simulations and shown by our visualizations is that the largest velocities are exhibited not when foam moves through the constriction but at what first appears as seemingly random times and positions (Figure 5.1).

For this reason the color bar is clamped at about one tenth of the total velocity magnitude range. Previously, foam scientists hypothesized that those high velocities are caused by topological changes (T1s). We can now verify this hypothesis by matching T1 positions with positions of high velocity bubbles. Figure 3.16 shows, by using the velocity magnitude histogram, that only a few bubbles have very high velocity magnitude and determine the upper limit of the velocity magnitude range.

5.2 Why does one disc initially descend quicker than the other?

An important question foam scientists wish to answer relates to what triggers the interaction between the two discs in the simulation of sedimenting discs. That is, why does one disc initially descend quicker than the other? In Figure 5.2 top, we see that the disc on the right

[1] Charles Martin "Chuck" Jones was an animator, cartoon artist, screenwriter, producer, and director of animated films, most memorably of Looney Tunes and Merrie Melodies shorts for the Warner Bros. Cartoons studio. (Wikipedia)

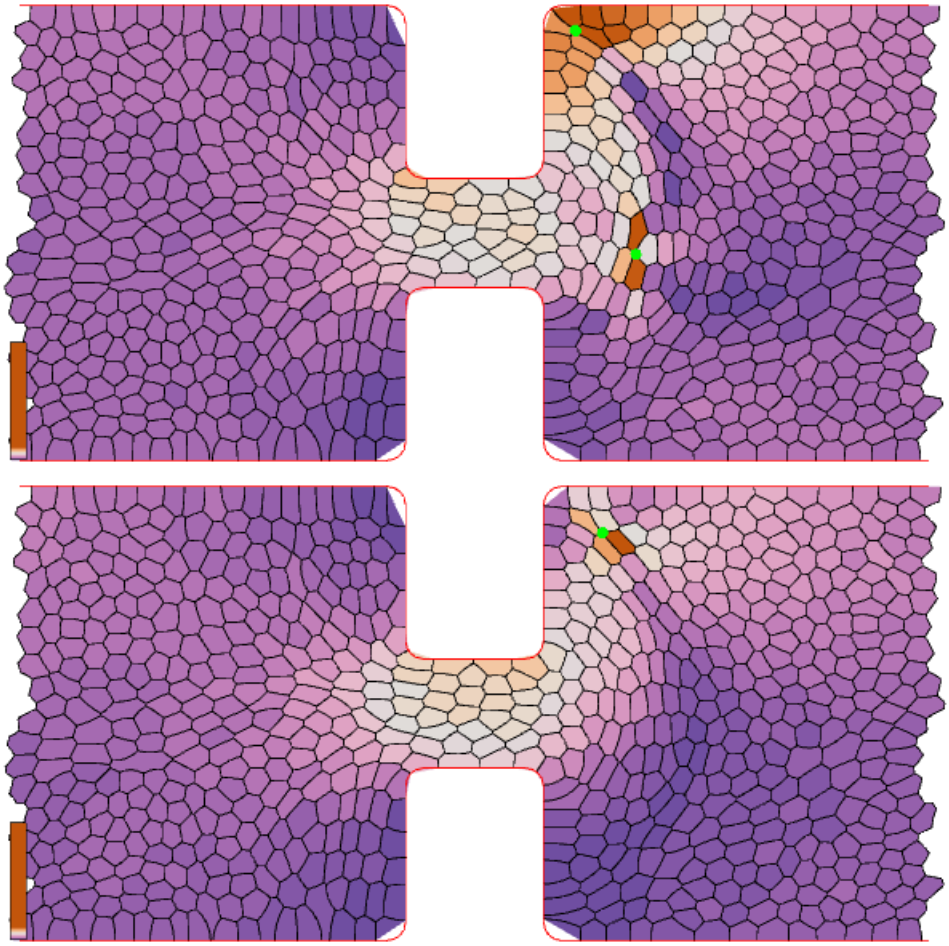


Figure 5.1: High velocity bubbles outside the constriction are caused by T1 events (green dots). Time steps t=62, t=63 for the simulation of foam flow through a constriction. Bubbles are color-mapped to velocity magnitude. The color bar is clamped to about one tenth of its range to highlight the high velocity in the constriction area. The highest velocities are caused by T1 events.

trails the disc on the left as a result of the initial arrangement of the surrounding bubbles. For $t = 0$, the network force acts downwards for the disc on the left while the pressure force is negligible. However, both network and pressure forces act upwards for the disc on the right. The difference in the network force exerted on each disc is due to the different distribution of films in contact with each disc. Similarly, the difference in the pressure force exerted on each disc is due to the differences in the shapes of bubbles in contact with each disc. As a result, the disc on the right is left behind. It is the initial configuration of the foam that decides which disc gets left behind (the disc on the right in the case shown).

5.3 Why is a terminal separation of roughly two bubble diameters attained between the two discs?

To answer this question, we investigate the shape of bubbles between the discs once they have reached this separation. Using a sliding time window average of bubble elongation over 10 iterations, we observe a competition between the effects imposed by each disc on the foam. We expect bubbles in front of the trailing disc to be compressed and bubbles in the wake of the leading disc to be stretched, so their elongation is high in both cases. However, it can be seen in Figure 5.2 bottom-left that the competition between the stresses applied by each disc results in a lower than expected deformation of the foam occurring between the two discs. This competition results in the bubbles in-between the two discs remaining (roughly speaking) undeformed. As a result, the forces experienced by each disc become similar. A terminal separation of two bubbles is attained between the two discs, although why this is two and not, say, one or three is a question to which we will return in future work.

5.4 Why do discs drift laterally as they sediment?

To gain insight into this question, we consider how the pressure and elongation field of the foam evolves as the discs descend. It could be seen from previous visualizations of individual time steps that the pressure field is not symmetric during the rotation of the two discs about one another. This is validated by images produced using image-based statistical calculation and visualization, the *sliding time window* and *stationary body* features in FoamVis (Figure 5.2 middle and bottom). Here, it is observed that a region of higher pressure occurs to the right of both discs during their descent. The deformation of the foam is such that bubble pressures contribute to the drift of the discs as they sediment.

We can also observe that the network force initially contributes to forcing the rear disc to move laterally (Figure 5.2 middle-left). However, once the discs have moved closer to their stable orientation, it is the pressure force that is driving the lateral drift (Figure 5.2 bottom).

5.5 Pattern of bubbles traversing loops

For two discs sedimenting through a foam, our paths visualization shows that bubbles traverse loops to provide space for the descending discs. This behavior is not observed in standard visualizations used by the domain scientists. Figure 5.3 left shows bubble paths color-mapped to velocity along the Y axis, with orange showing downward velocity and purple showing upward velocity. The orange area of high velocity along the Y direction shows the paths of the two discs, and the black rectangle marks the region magnified in the right image. A loop consists of a downward segment (colored orange) and an upward loop (colored purple). A bubble traverses the downward segment as a descending disc approaches it. Then it traverses the upward loop as the disc passes by it. The bubble avoids the falling disc and then fills the space that it leaves.

5.6 What is the effect of varying the shape of the constricted channel on the elastic and plastic deformation in a flowing foam?

Figure 3.7 shows the deformation, averaged over the entire duration of the simulations, using the *two halves* view. Here, we visualize the elastic response of the foam and how it is affected by the roundness of the corners of the constriction. Rounding the corners results in reduced elastic deformation of the foam. In the square-constriction, foam is highly compressed both upstream, as bubbles are pushed against the wall, as well as downstream, as bubbles detach from the wall. This does not occur to the same extent in the rounded-constriction (independent of the length of the constricted region). An area where bubbles are not deformed can be observed just downstream from the constriction in both simulations. Here bubbles move from an area where they are deformed flow-wise (inside the constriction) to an area where they are deformed span-wise (downstream from the constriction).

Comparing Figure 3.7 and Figure 3.14 shows that only in the square constriction are the bubbles attached to the wall downstream of the constriction significantly stretched, which gives rise to a higher density of topological changes as they occasionally detach. In the rounded constriction, the bubbles slide around the wall (Figure 3.8), do not get stretched, and do not trigger topological changes.

As the original FoamVis provided only a visualization for the deformation scalar and locations of topological changes, this analysis would have been difficult. The deformation scalar does not indicate the direction in which bubbles are deformed and the direct visualization of topological changes suffers from over-plotting.

5.7 Do we have circulation and regions of stagnated flow in a constriction?

There have been a number of studies of foam flow through a constriction (see the review of Jones et al. [JDS$^+$11]), but none have ever found recirculation. Most recently, Jones and Cox [JC12] examined both long-time averages and instantaneous vector plots of the velocity and concluded that vortices were not present in any of the different constriction geometries that they examined.

A subset of this data was re-examined using FoamVis, which offers the possibility to easily change the time-window over which the velocity field is averaged and can show streamlines to facilitate observation of recirculatory behavior for one time step. As Figure 5.4a shows, over intermediate time-scales (here 50 time-steps) recirculation appears to occur. However, only a few individual time steps show recirculation, and here it is caused by topological changes (Figure 5.4b). So the effect of the time average is to dilute this circulatory motion, but clearly it does not completely do so. Further, Figure 5.4c shows that an individual step can show recirculation in the absence of topological changes, a surprising and potentially significant finding, although the bubble velocities are very small. Because simulations are quasi-static, i.e. foam is at equilibrium in each time step, a topological change in previous time steps cannot

determine circulatory motion in the current time step. Because the circulation motion does not persist for many time steps bubble paths or pathlines/streaklines do not show it.

In the rounded constriction, there is much less recirculation, which is presumably related to the decrease in the density of topological changes.

This new finding stimulates further questions. With access to more data, we hope to be able to answer questions about circulations' persistence when parameters such as bubble size and polydispersity and constriction shape are varied.

The velocity field visualization (Figure 3.8) shows clear differences between the flow in both geometries. The "dead zones" from the foam are only visible in the top corners of the square-constriction as dark purple regions. Rounding the corners results in the disappearance of these stagnated bubbles.

5.8 Can we approximate the sedimenting-discs behavior with the sedimenting ellipse behavior?

We probe a foam's response to the sedimentation of solid objects with the aim of being able to predict the path and residence time of an object in a foam. This has application in industrial processes such as froth flotation for ore separation. We consider the sedimentation of two interacting circular discs and an ellipse in a dry monodisperse foam.

The interaction between two circular discs sedimenting in a dry foam is such that they reach a stable configuration in which they are directly above one another and separated by (roughly) two bubbles. Thus when discs are initially side-by-side in the foam, they rotate about one another into this stable configuration. They interact in this way as long as they are within a critical separation of each other. The critical separation is dependent on whether the region where topological changes are concentrated around each disc are merged [DC09]. We wish to understand the extent to which this result can be related to the similar but simpler result; that of a sedimenting ellipse in a foam. An ellipse that is initially horizontal in the foam rotates so that it becomes vertical during sedimentation. The main driving force behind this process is the torque exerted on the ellipse by the films as they bunch up at an off-center position on the ellipse's boundary [DC10].

Figure 5.5a shows how the flow of foam is similar when the orientations of the objects match each other. We note that bubbles in-between the two discs are moving at a high velocity, with the discs. We previously noted that this is a region in the foam where bubble deformation is lower than expected (Section 5.3). We also note that compared to regions such as the wake of the trailing disc and in front of the leading disc, fewer topological changes occur between the discs (Figure 5.5b). As a result, the foam behaves (mainly) as an elastic solid in between the two discs. This results in the two discs behaving as a single object in the foam. In this case, one would expect the two discs to behave similar to a longer object such as an ellipse.

The elastic deformation (Figure 5.6) caused to the foam by both types of objects has some similarities: Regions of high deformation appear in the wake (where bubbles are stretched vertically) and in front of the objects (where they are squeezed horizontally). The pressure field is also very similar in both cases (Figure 5.6). For example, Figure 5.6a left shows a region of high pressure mainly on the right side underneath the ellipse which contributes to

the drift of the ellipse toward the left wall. Similarly, for the two discs (Figure 5.6a right), the pressure is higher underneath the right hand disc resulting in a greater pressure drag being exerted on it, which contributes to the initial faster descent of the disc on the left.

However, there are also clear differences: For the ellipse, the region of high deformation in the wake is always positioned at the highest point of its boundary. Once it is tilted slightly from its initial orientation (due to the disorder of the foam) this region moves to an off-center position. Films become bunched up together here and contribute to a large network force that drives the rotation (Figure 5.6b left). Note here that a smaller pressure force opposes the rotation of the ellipse. This is a different mechanism to what is driving the rotation of one disc around the other. The force difference tool (see Figure 5.6b right) shows that both the network and the pressure forces contribute to the rotation of one disc around the other.

The objects continue to rotate into their stable configuration and once they become oriented so that they are vertically aligned, both the pressure and deformation fields become symmetric (see Figure 5.6c). The stability of these orientations is confirmed by the force visualizations: the resultant torque on the ellipse and force difference on the discs are each close to zero. The combined visualizations shown in Figure 5.6 allow us to probe the different contributions of network and pressure forces on these two similar results.

To fully understand the *sedimenting-discs* simulation, we must collate more simulations in which the initial separation between the two discs and the disc size as well as the ellipse eccentricity and size are varied. We know [DC10] that changing the ellipse's eccentricity and size changes its rate of rotation and that changing the disc separation and size affects the rate at which they rotate around each other [DC09]. We want to know which combination of parameters results in the most similar behavior for both types of simulation, which would allow us to further compare and contrast the two simulations.

5.9 New simulation parameters chosen using FoamVis

Noting the similarities between the sedimenting discs and the sedimenting ellipse simulations, we use FoamVis to decide on new simulation parameters. Here we aim to predict the size and shape of the ellipse required to obtain a similar rate of rotation to the two discs in the foam. The similarities in the flow field for both simulations shown in Figure 5.5a suggest that by choosing a larger and more eccentric ellipse, we should obtain better matching simulations. We propose that an ellipse with an area of at least ten times the bubble area and an eccentricity of 0.7 will be adequate. These parameters are chosen so that the shape of the ellipse can be fitted to cover the two discs and the elastic region of foam in-between. (In Figure 5.5a, the ellipse has an area of four times the bubble area and an eccentricity of 0.8.) The larger, more eccentric ellipse experiences a greater torque [DC10] in the foam and therefore rotates at a greater rate during sedimentation.

5.10 Topological change trails for the falling disc (2D) and falling sphere (3D) simulations

In a two-dimensional foam, a T1 occurs when two bubbles approach one another and two move apart. A bubble edge shrinks to zero length, forming an unstable vertex at which four edges meet. This is energetically unstable (Plateau's laws [CCAE$^+$13]), and immediately dissociates into two vertices separated by a new edge. The two bubbles that were initially neighbors move apart, and the two approaching bubbles become neighbors. We represent each of these events as a point on Figure 5.8 left.

In a three-dimensional foam, the situation is more complicated. Bubbles have more degrees of freedom when they move, and there are different cases that we must consider. Firstly, there are two "standard" T1s:

1. if a bubble edge shrinks to zero length in 3D, then the resulting unstable vertex is replaced by a small triangular face (soap film); following Brakke [sur04], we refer to this as an edge_to_tri transition;

2. alternatively, if a small triangular face shrinks to zero area, then the resulting unstable vertex is replaced by a short edge; we refer to this as an edge_to_tri transition;

3. a further T1, in which a rectangular face shrinks to zero area and is replaced by another rectangular face, perpendicular to the first one, can be viewed as a composition of the above two topological changes; we refer to it as a quad_to_quad transition;

4. there are also two topological changes that we use to ensure that the topology of the tessellation remains an accurate representation of foam structure, for example if the structure is such that none of the above changes complete correctly: firstly, an edge may acquire more than three faces attached to it (violating another of Plateau's laws), in which case we perform a pop_edge transition to introduce a rectangular face;

5. secondly, a vertex may become attached to more than four edges (violating the 3D version of Plateau's first law), in which case we perform a pop_vertex transition to introduce a new edge joining two vertices.

We represent each of these T1s with a different color sphere, see e.g. Figure 5.8 right.

Fig 5.8 shows good agreement between the 2D and 3D datasets. Both simulations display a trail of T1s within close proximity of the path of the falling object. This demonstrates where the foam has been deformed the most, or "fluidized", by the influence of the solid object.

The disc in 2D seems to have a more wide ranging effect on the foam than in 3D. This may be the result of the 3D foam being too small for a more fair comparison here. The 3D small sample means that the foam might be over constrained. The bubbles have nowhere to go out of the way of the sphere and will therefore just stay in front of the sphere and move with it. A surprising feature of the simulation discovered using our software is that there are no tri_to_edge topological changes. Through investigation, domain experts realized that the order in which tests for deciding which different types of topological changes are applied matter. In

particular, tri_to_edge and quad_to_quad types of topological changes are exclusive, you get one or the other depending on which you test first. Note this is a feature of the simulation, it is not known which types or what is the distribution of different types of topological changes that happen in real foam. These are interesting questions for future foam research.

5.11 Bubble loops in 3D

This visualization confirms for domain scientists that, as in 2D, bubbles traverse loops in 3D *in an axisymmetric way* to provide space for the descending sphere. Figure 5.9-right shows a bubble and the sphere paths color-mapped to velocity along the Y axis, with blue showing downward velocity and red showing upward velocity. A loop consists of a downward segment (colored blue) and an upward loop (colored red). A bubble traverses the downward segment as the descending sphere approaches it. Then it traverses the upward loop as the sphere passes by it. The bubble avoids the falling sphere and then fills the space that it leaves. The loops get smaller as the distance of the bubbles to the sphere gets larger. A future direction of investigation for domain experts, triggered by our visualization, is to use the loop size to determine the distances to which the sphere influences the foam.

In Figure 5.10 we see that essentially the same thing is happening both in 2D and 3D. For the 3D case, it is not quite as smooth due to the small size of the simulation. We see a circulation flow either side of the disc in 2D and all around the sphere in 3D. This is the result of the volume constraint for both simulations.

5.12 KDE for topological changes around the falling disc (2D) and falling sphere (3D) simulations

Applying a KDE visualization for topological changes around the falling sphere yielded a surprising result: a density sphere centered just above the falling object, instead of the pear shape that we got for a 2D simulation of a falling disc (Fig 5.11b). We investigate possible causes and we discover that certain time steps have a large number of topological changes occurring approximately at the same position - on top of the falling sphere. Note that the maximum value in the color bar for 3D is 36 which denotes the maximum number of topological changes that occur in one time step. Repeated topological changes occurring on top of the falling sphere dominate the final result. These topological changes are an artifact of the quasi-static approximation, which allows faces or edges to repeatedly undergo a T1 and then a "reverse" T1 during convergence. Our collaborators investigate ways in which to eliminate this artifact, for example by introducing dissipation.

5.13 Topological changes cause high velocity bubbles

Previously, foam scientists hypothesized that high velocities are caused by topological changes (T1s) and we were able to verify that this is the case in 2D. We can now verify this hypothesis in 3D by matching T1 positions with positions of high velocity bubbles. The disordered directions

of the arrows in Figure 5.12 right is a result of the topological change. Space left by bubbles moving away from each other close to the topological change (red arrows moving in opposite directions) is filled by bubbles in close proximity (smaller blue arrows pointing upwards).

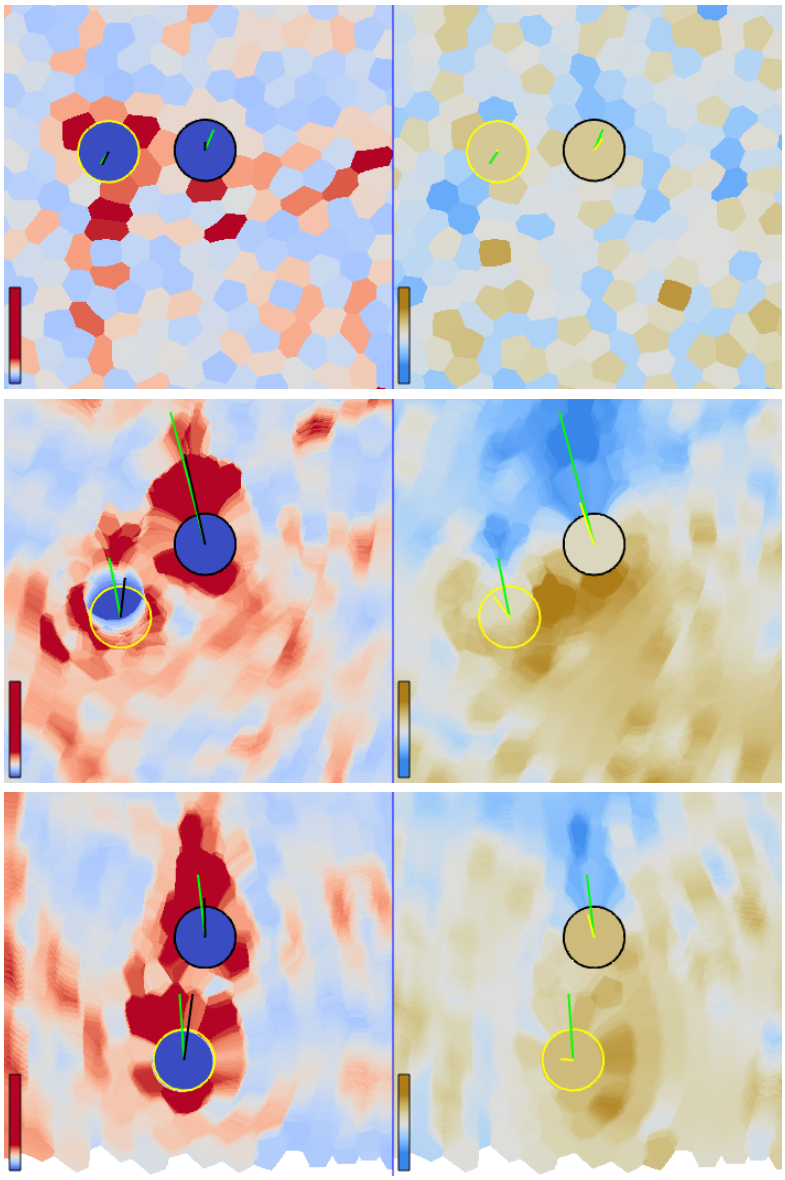


Figure 5.2: Three time steps in the sedimenting-discs simulation: (top) Beginning of the simulation $t = 0$. (middle) The discs rotate about one another, $t = 43$; (bottom) Behavior near the end of the simulation, where the stable orientation of the discs is approached, $t = 246$. Each time step contains two views: with bubbles color-mapped to elongation (blue-red palette) and pressure (blue-tan palette). For time steps $t = 43$ and $t = 246$ we show a sliding time window average over 10 iterations for both scalar fields and forces. The stationary disc is marked with a black circle and the moving disc is marked with a yellow circle. We use black for the network force, yellow for the pressure force and green for the resultant force (pressure + network force) that acts on a disc. Note the incorrect average calculation around the moving disc (yellow circle) with high velocity (middle).

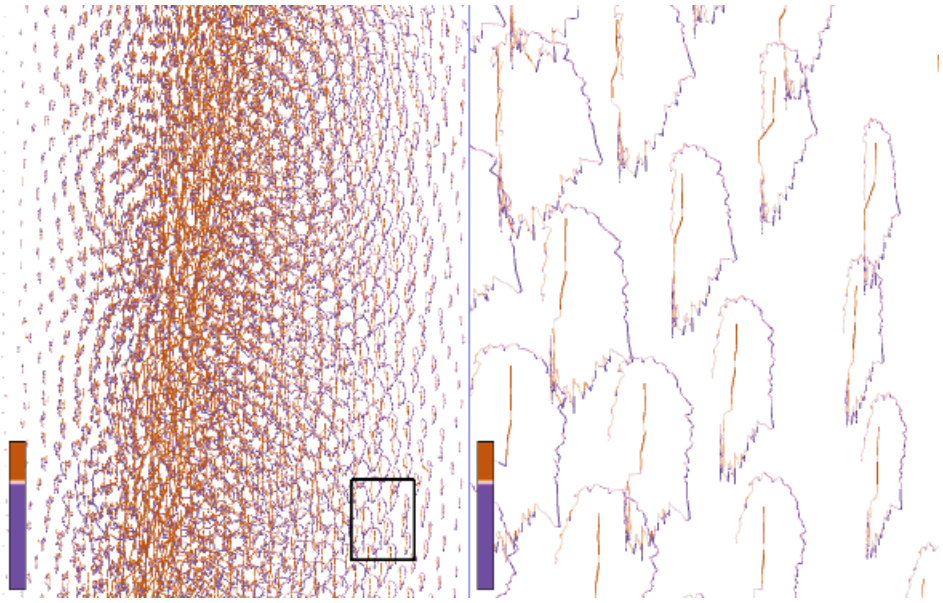
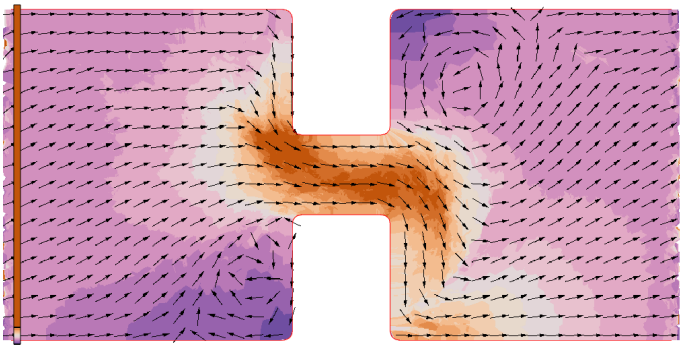
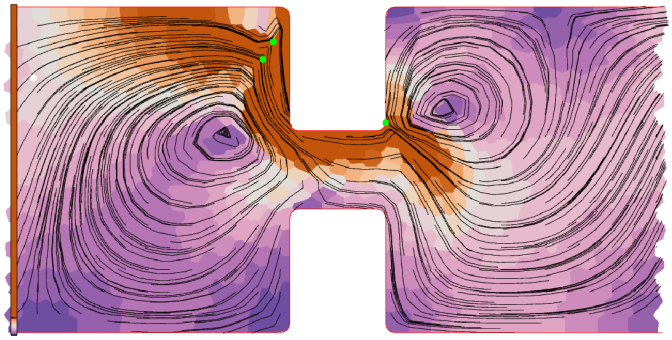


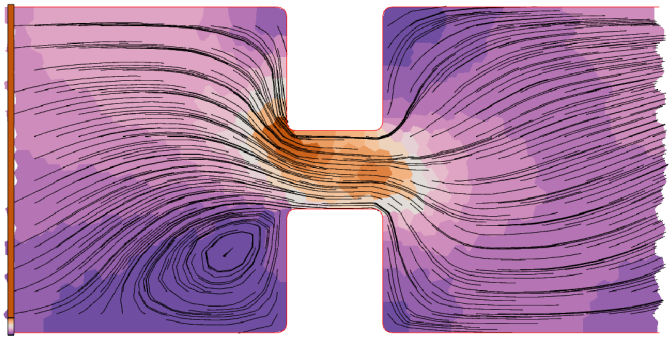
Figure 5.3: Pattern of bubbles traversing loops not previously observed by domain experts. The bubbles paths are color-mapped to velocity along Y, with orange indicating descent and purple indicating ascent. The left image shows the bubble paths over the entire simulation. The red area shows paths of the two discs. The black rectangle shows the region that is magnified in the right image.



(a)

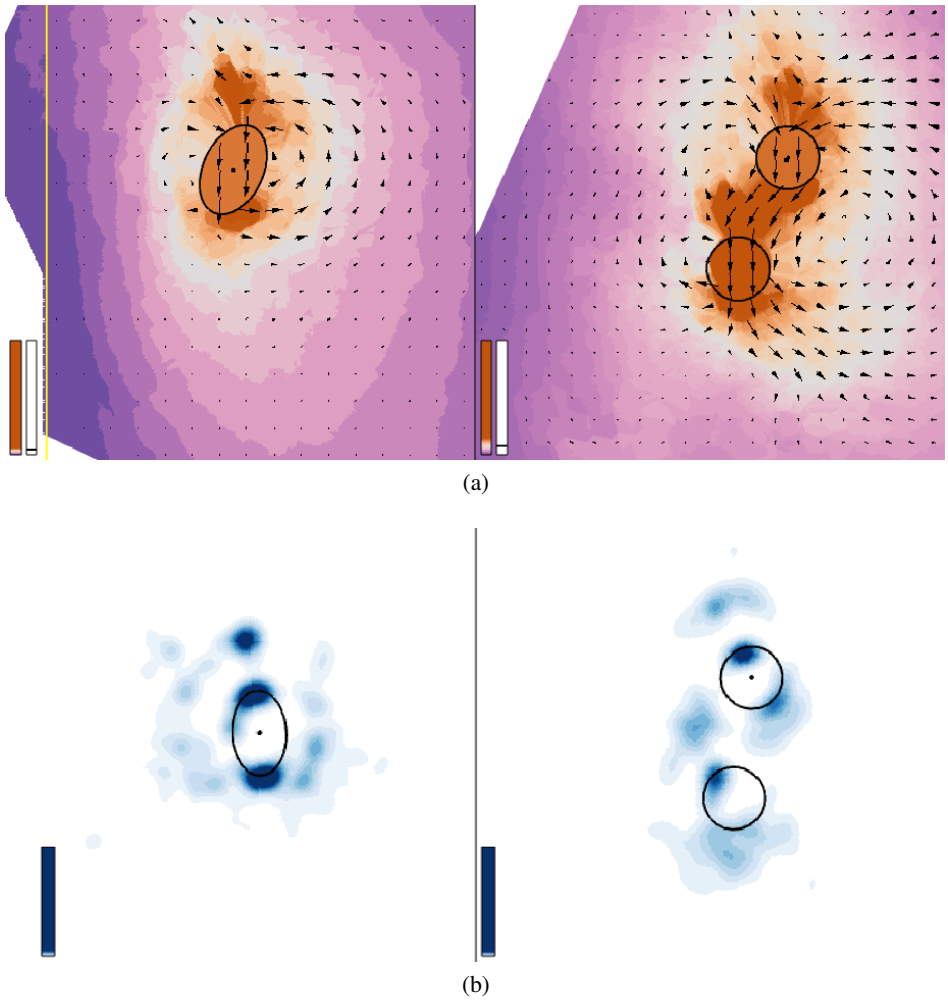


(b)



(c)

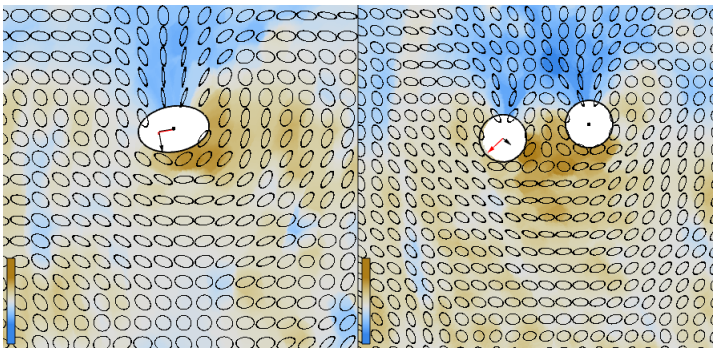
Figure 5.4: (a) Velocity is shown with glyphs of the same size. An average over 50 time steps is computed, for both velocity vectors and velocity magnitude, $t = 441$. The visualization shows apparent circulation of bubbles within the square-constriction flow (bottom-left and top-right). (b) Topological changes are shown with green dots, velocity field is shown with streamlines, $t = 412$. Topological changes cause strong circulation movement. (c) Velocity is shown with streamlines, $t = 417$. We show circulation of bubbles not caused by topological changes, a result never presented before. For all figures velocity magnitude is color-mapped (orange is for high and purple is for low velocity magnitude).



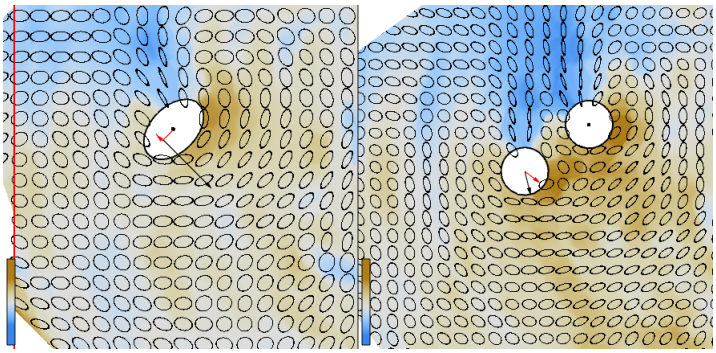
(a)

(b)

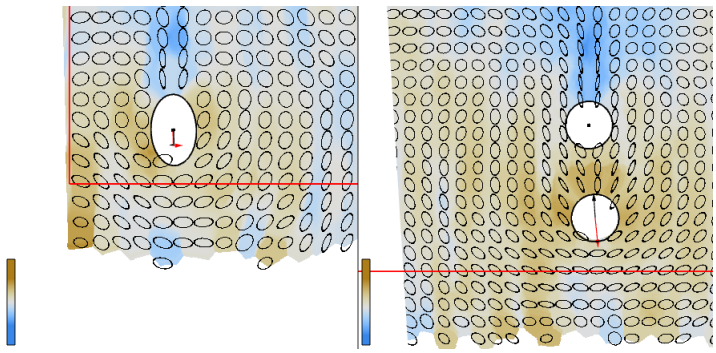
Figure 5.5: Sedimenting-ellipse versus sedimenting discs. (a) Visualization of velocity average, time window of 30 time steps, *around* the ellipse and the two discs, that uses the linked time with event synchronization feature (the ellipse and the two discs reach orientations 0° 30°, 60° and 90° in the same time). Velocity is displayed using glyphs and velocity magnitude is also color-mapped (with orange for high and purple for low velocity magnitude). The foam between the discs moves at high velocity with the discs which creates a similar velocity field as for the falling ellipse. (b) Few topological changes (T1s) occur between the discs, so foam in that region behaves mainly as an elastic solid. Topological changes *around* the ellipse and the two discs (Section 3.3.3.1), over the entire duration of the simulations, visualized using KDE. Images use the *show rotation* option to render the rotation context for the ellipse and the discs.



(a) Beginning of the simulation, $t = 133$.

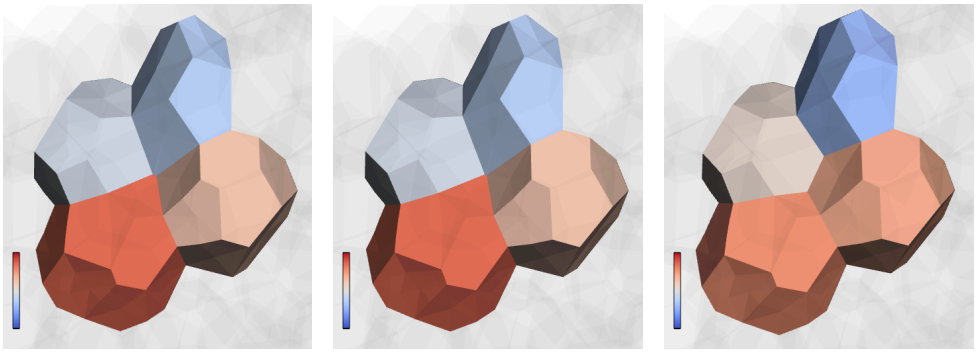


(b) Rotation phase, $t = 283$



(c) Stable orientation, $t = 1169$

Figure 5.6: Sedimenting-ellipse versus sedimenting-discs. The *linked time with event synchronization* feature is used to synchronize the rotation of the ellipse and the two discs such that they reach an orientation of 45° at the same time. Attributes (pressure, deformation and forces) are averaged over 52 time steps for the ellipse simulation (resulting in an average over 15 time steps for the two disc simulation). Pressure is color-mapped (tan for high and blue for low pressure), deformation is shown using ellipses. The force *difference* between the leading disc and the trailing disc and the torque on the ellipse is indicated. The network force and torque are indicated with a black arrow and the pressure force and torque are indicated with a red arrow. The channel wall can be seen in Figure (b) left and Figure (c) left; Figure (c) bottom shows the lower, periodic boundary of the foam.



(a) $t = 36$ (b) $t = 37$ (c) $t = 38$

Figure 5.7: 3D topological change of type tri_to_edge. Bubbles are colored by number of faces per bubble: $(18, 15, 13, 12)$. The first two images show bubbles just before the topological change and the third image shows bubbles after the topological change. After the topology change the number of faces in each bubble changes to $(17, 16, 14, 11)$.

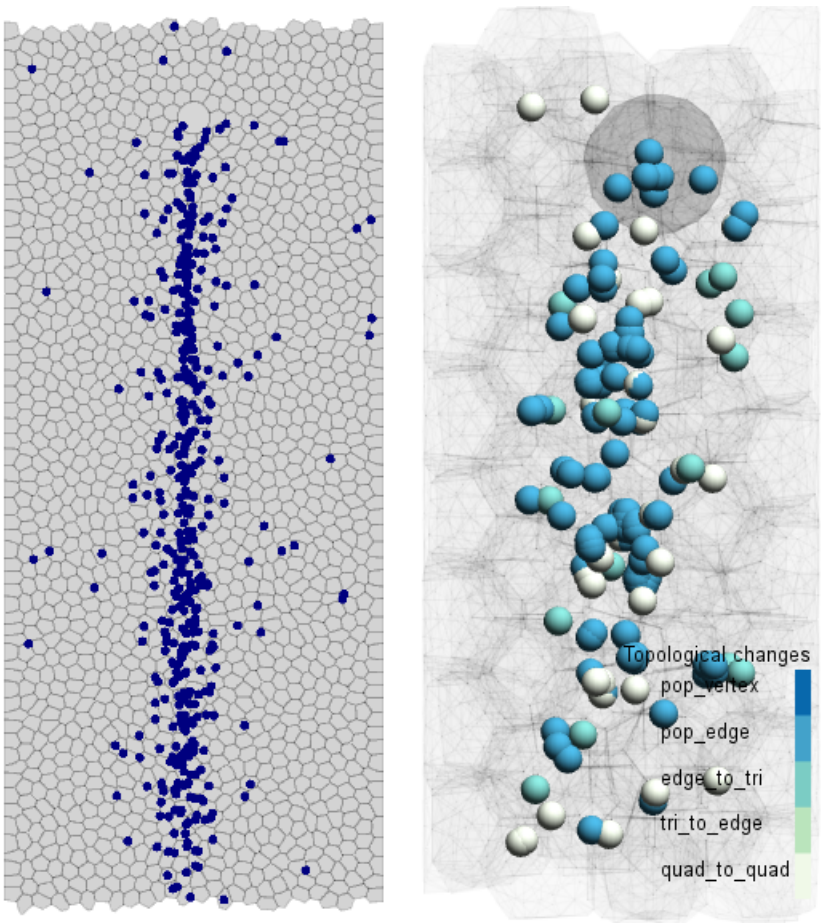


Figure 5.8: Topological change trails for the falling disc (2D) and falling sphere (3D) simulations. In 3D, topological changes are represented as spheres colored by the their type.

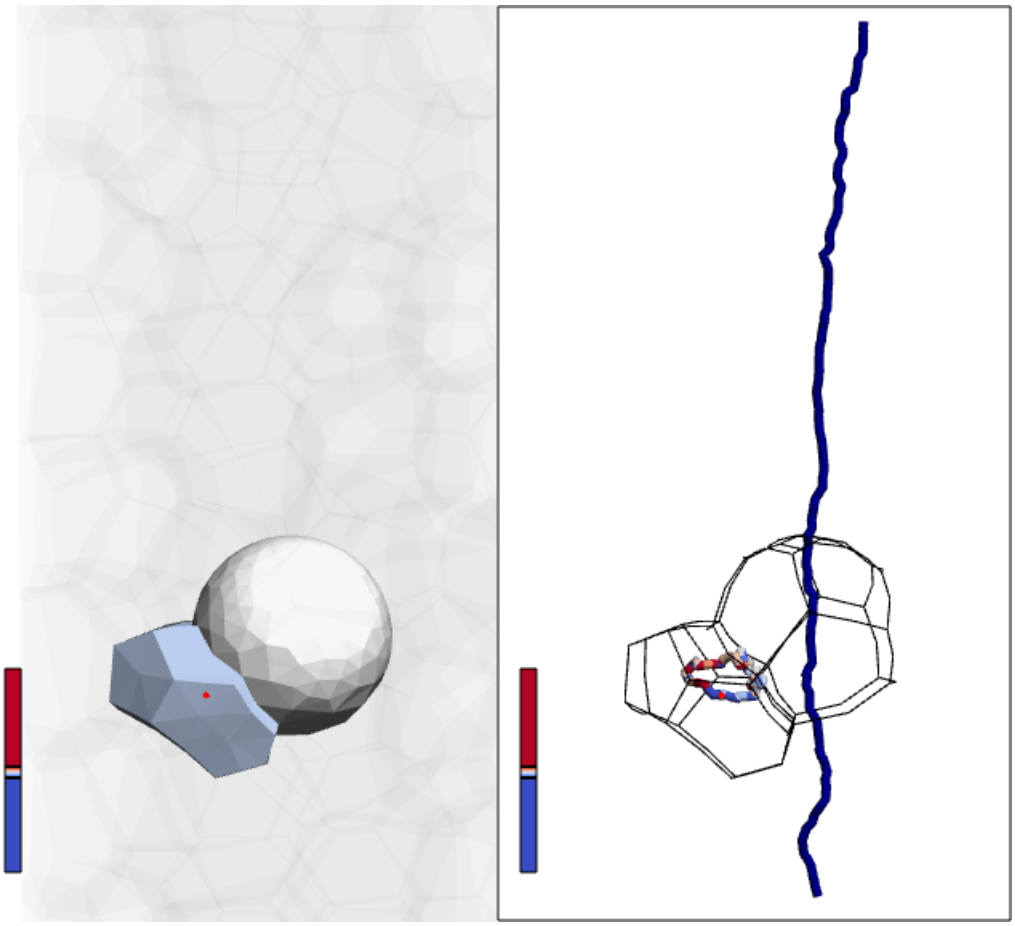


Figure 5.9: A bubble path that forms a loop. This behavior was not previously observed in 3D by domain experts. The two views show the falling sphere and the bubble that traverses a loop. Bubble center is marked with a red dot. The right view shows only edges for the sphere and bubble and the paths traversed during the simulation. Bubble paths are colored by velocity along the Y axis, with blue showing downward and red showing upward velocity.

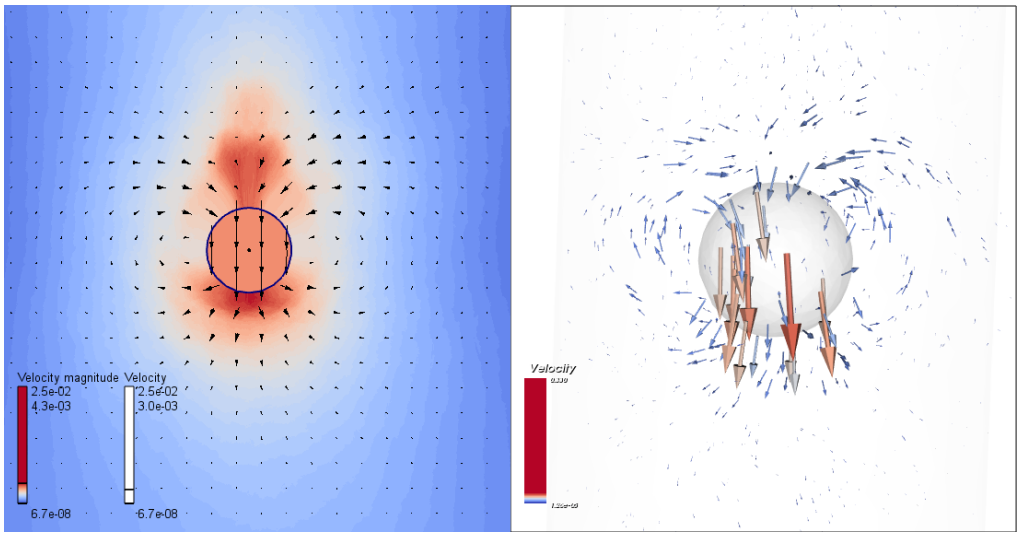
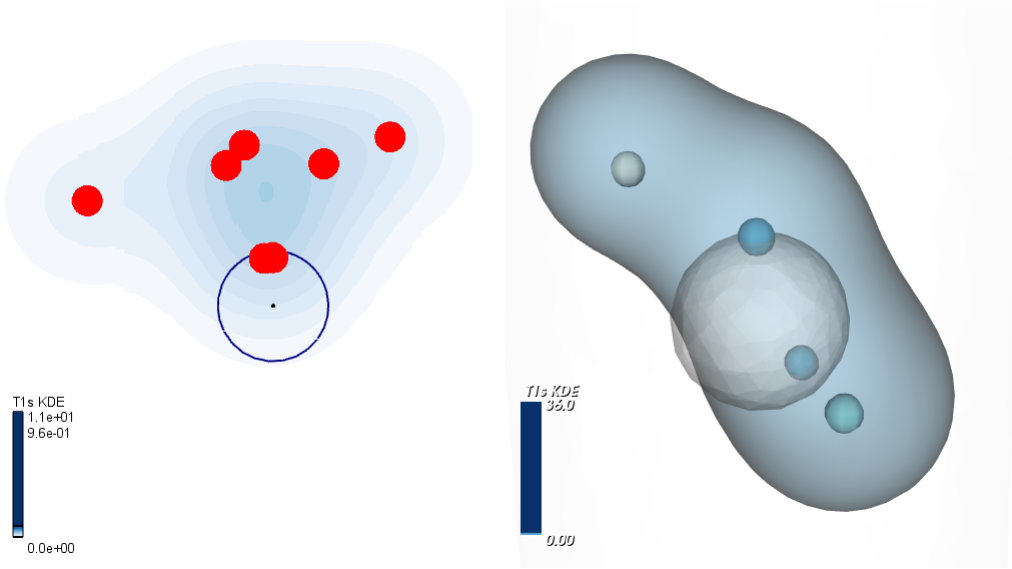
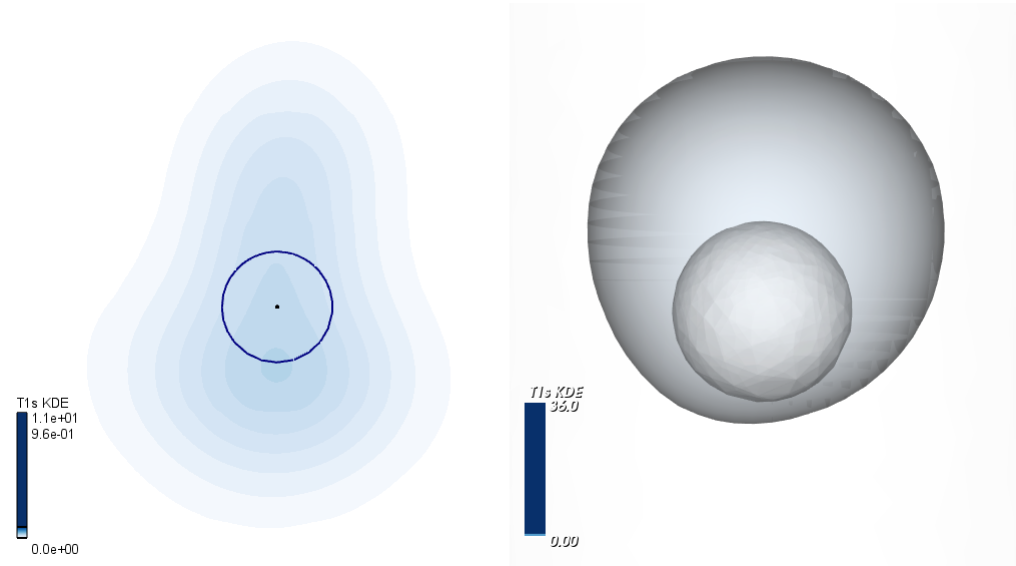


Figure 5.10: Velocity average *around* the falling disc (2D) versus the falling sphere (3D) simulations. A similar pattern can be observed in 2D (left view) and 3D (right view). Bubbles are pushed down by the falling object, they move to the side to make space for it, and then they fill its space as the object passes them. In the left view we show velocity magnitude scalar and the velocity vector. In the right view we show the velocity vector colored by velocity magnitude. Both the scalar and vectors sizes are clamped using the color bars shown in the lower left corners.



(a) KDE for one time step: $t = 18$ left view and $t = 21$ right view. Isosurface density is 0.5 for the right view.



(b) KDE for all time steps. Isosurface density is 0.12 for the right view.

Figure 5.11: KDE *around* the falling disc versus falling sphere simulations. The maximum values in the color bar represent the maximum number of topological changes in a time step. KDE for all time steps (b) shows that, for 3D, topological changes on top of the sphere dominate the final result. This is caused by topological changes in the same area being triggered repeatedly in the simulation code, feature discovered using our visualization.

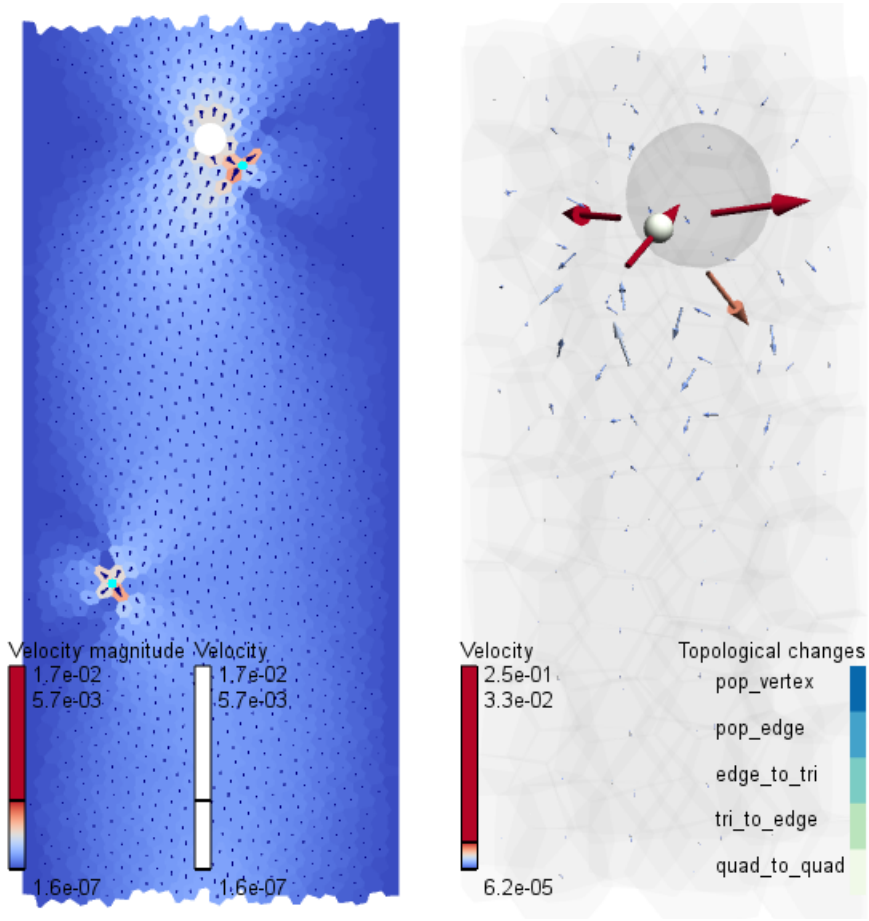


Figure 5.12: Topological changes cause high velocity bubbles. In the left view we show bubble velocity and velocity magnitude scalar as well as the position of topological changes at $t = 6$. In the right view we show bubble velocity colored by velocity magnitude and the position of topological changes at $t = 12$. Note that in both views the velocity (and velocity magnitude) is clamped because velocities caused by topological changes are much larger than bubble velocities caused by the falling sphere.

Chapter 6

Conclusions and Future Work

"To succeed, jump as quickly at opportunities as you do at conclusions."
– Benjamin Franklin (1706–1790)[1]

We described the foam research application area and we presented a new application, called FoamVis, that provides various techniques for the exploration, analysis, and visualization of foam simulation data. FoamVis can process individual foam simulations, it facilitates comparison of related simulations and works for both 2D and 3D data. Our tool validates previous hypothesis and yields a more convenient representation of simulation data. Application scientists using our tool make new discoveries and gain insight into foam behavior so, we believe we provide a valuable tool for exploration, visualization and analysis of data in the foam research community.

There are ample opportunities to continue our work on FoamVis and visualization of foam research and we could extend FoamVis to address questions in other physical sciences that use Surface Evolver for simulation. Next, we present research directions we would like to pursue in the future.

Three-dimensional visualization capabilities are a recent addition in FoamVis. Domain experts work with many more 2D foam simulations so we focused our work in two-dimensions. However, 3D simulations are also important (see Section 3.1), so we would like to provide additional 3D visualization tools. Examples include volume visualization for scalars and visualizations using streamlines for vectors and glyphs for tensors. These extensions are facilitated by our adoption of the VTK [Inc10b] visualization framework. Many visualization algorithms are already implemented and tested in VTK so work would be required only for integration into FoamVis.

An interesting direction for future work is to port our visualization tool to ParaView [Inc10a]. ParaView is an open-source, client-server, multi-platform data analysis and visualization application based on VTK. Data exploration can be done interactively on a desktop, through a web interface or on a mobile device. While visualization capabilities offered by the ported FoamVis would be similar to what we already provide, this work may

[1] Benjamin Franklin was one of the Founding Fathers of the United States. A noted polymath, Franklin was a leading author, printer, political theorist, politician, postmaster, scientist, musician, inventor, ... (Wikipedia)

enhance access to foam research (through the web interface) and provide a more engaging experience for students (through the mobile interface). The porting work will take advantage of the ParaView plug-in architecture which allows the integration of custom data readers and data processing algorithms (filters) into the program without the need to recompile it. We will have to provide the **Parser** component (see Section 4.2) as a ParaView plug-in and implement visualization algorithms using custom VTK pipelines. Algorithms or techniques that are not available in Paraview/VTK will have to be integrated as plug-ins.

In Section 2.3 we describe important dynamic behaviors of foam, such as coarsening and drainage, that are not covered in our case studies. Similarly, problems related to wet foams and their dynamic behavior are not discussed. FoamVis can parse, visualize and analyze these simulations using the provided tools however no questions specific to these simulations were pursued. This direction of work may yield interesting new inquiries, useful additions to our tool and increased understanding of foam behavior.

New imaging techniques such as synchrotron tomography have allowed scientists to probe the internal structure of soft materials such as foams and have led to the availability of imaging data of foam undergoing coarsening, drainage or rheology. For each data acquisition step, a set of images taken at parallel and equidistant planes through the foam are provided. Recently, techniques to process these images to reconstruct the original soap films were presented [DCL13]. However, the reconstructed bubbles do not have an identity, which means that we cannot track a bubble from one time step to the next. Tackling this unsolved problem is a promising future research direction. Bubble IDs together with foam structure would provide similar information for foam experiments as we have for simulations enabling new ways to compare simulations with experiments using FoamVis.

One of the main challenges of using Surface Evolver to simulate dynamic foam processes is to be able to simulate a sufficiently large number of bubbles to accurately reflect reality. The most time consuming operation in SE simulations is the computation of the film surface with minimal energy which is done for each time step (Section 2.4). This computation is done using gradient descent [Wei14], an algorithm for finding the nearest local minima of a function which assumes that the gradient of the function can be computed. Variations of this method have been parallelized [ZWSL10, NRRW11]. We believe that speeding-up foam simulations through parallel gradient descent or other methods is an attractive area of future research. Foam scientists would be able to produce larger simulations which may advance the state of the art in foam research and pose new visualization and analysis challenges.

Last, adding support for visualization and analysis of other kinds of Surface Evolver simulations such as the study of emulsions [RTTD04] or solder in electronic circuits [CC98, HBL99] is a promising direction of future research.

These new research directions open exciting possibilities for the advancement of visualization and for increasing its contribution to foam research and to the development of human knowledge in general.

Bibliography

[AHP+10] J. Ahrens, K. Heitmann, M. Petersen, J. Woodring, S. Williams, P. Fasel, C. Ahrens, Chung-Hsing Hsu, and B. Geveci. Verifying scientific simulations via comparative and quantitative visualization. *IEEE Computer Graphics and Applications*, 30(6):16 –28, nov.-dec. 2010.

[BD94] Gerard Bashein and Paul R. Detmer. Centroid of a polygon. In *Graphics Gems IV*, pages 3–6. Morgan Kaufmann, 1994.

[Bel04] Donald Bell. UML basics: The component diagram, 2004. Online document, accessed 25 June 2013, `http://www.ibm.com/developerworks/rational/library/dec04/bell/index.html`.

[BGG+06] J. Bigler, J. Guilkey, C. Gribble, C. Hansen, and S.G. Parker. A Case Study: Visualizing Material Point Method Data. *EG Computer Graphics Forum*, pages 299–306, 2006.

[BN47] L. Bragg and J.F. Nye. A dynamical model of a crystal structure. *Proc. R. Soc. Lond.*, A190:474–481, 1947.

[Bra92] K.A. Brakke. The Surface Evolver. *Experimental Mathematics*, 1(2):141–165, 1992.

[Bre] Cynthia A. Brewer. ColorBrewer. Online document, `http://www.ColorBrewer.org`, accessed 3 March. 2012.

[bri14] Encyclopædia Britannica, Surface Tension, 2014. Online document, accessed 14 February 2014, `http://www.britannica.com/EBchecked/topic/575080/surface-tension`.

[CC98] K.N. Chiang and W.L. Chen. Electronic packaging reflow shape prediction for the solder mask defined ball grid array. *Transactions-American Society of Mechanical Engineers Journal of Electronic Packaging*, 120:175–178, 1998.

[CCAE+13] Isabelle Cantat, Sylvie Cohen-Addad, Florence Elias, François Graner, Reinhard Höhler, Olivier Pitois, Florence Rouyer, and Arnaud Saint-Jalmes. *Foams. Structure and Dynamics*. Oxford University Press, 2013. Translated by Ruth Flatman, Edited by Simon Cox.

[Cil11] Jan Cilliers. Naked Engineering - Separating Minerals, October 2011. Online video, `http://www.youtube.com/watch?v=twmWdVhIkiY`, accessed 23 Sep. 2013.

[CLDL13] S.J. Cox, D.R. Lipşa, I.T. Davies, and R.S. Laramee. Visualizing the dynamics of two-dimensional foams with FoamVis. *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, 438:28–32, dec 2013.

[CSWZ11] Y. Chen, B.M. Schaeffer, M.M. Weislogel, and G.A. Zimmerli. Introducing SE-FIT: Surface Evolver–Fluid Interface Tool for Studying Capillary Surfaces. In *Proc. 49th AIAA Aerospace Sciences Meeting*, pages 1–11, 2011. `http://se-fit.com/`.

[CW06] S.J. Cox and E.L. Whittick. Shear modulus of two-dimensional foams: The effect of area dispersity and disorder. *The European Physical Journal E: Soft Matter and Biological Physics*, 21:49–56, 2006.

[Dav09] Ioan Tudur Davies. *Sedimentation of Circular and Elliptical Objects in a Two-Dimensional Foam*. PhD thesis, Aberystwyth University, 2009.

[DC09] I.T. Davies and SJ Cox. Sedimenting discs in a two-dimensional foam. *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, 344(1-3):8–14, 2009.

[DC10] I.T. Davies and SJ Cox. Sedimentation of an Elliptical Object in a Two-Dimensional Foam. *Journal of Non-Newtonian Fluid Mechanics*, 165(13):793–799, 2010.

[DCL13] IT Davies, SJ Cox, and J Lambert. Reconstruction of Tomographic Images of Dry Aqueous Foams. *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, 438:33–40, 2013.

[Den08] M. Dennin. Discontinuous Jamming Transitions in Soft Materials: Coexistence of Flowing and Jammed States. *J. Phys.: Condens. Matt.*, 70:283103, 2008.

[DLH11] O. Daae Lampe and H. Hauser. Interactive Visualization of Streaming Data with Kernel Density Estimation. In *Pacific Visualization Symposium (PacificVis)*, pages 171–178. IEEE, 2011.

[Eco10] The Economist. Data, data everywhere, 2010. Online document, accessed 18 July 2013, `http://www.economist.com/node/15557443`.

[fle] flex - The Fast Lexical Analyzer. Online document, `http://flex.sourceforge.net/`, accessed 29 Nov. 2010.

[GDRM08] F. Graner, B. Dollet, C. Raufaste, and P. Marmottant. Discrete Rearranging Disordered Patterns, Part I: Robust Statistical Tools in Two or Three Dimensions. *The European Physical Journal E: Soft Matter and Biological Physics*, 25(4):349–369, 2008.

[Gla00a] A. Glassner. Soap Bubbles: Part 1. *Computer Graphics and Applications, IEEE*, 20(5):76 –84, Sep./Oct. 2000.

[Gla00b] A. Glassner. Soap bubbles: Part 2 [computer graphics]. *Computer Graphics and Applications, IEEE*, 20(6):99 –109, Nov./Dec. 2000.

[gnu] Bison - GNU Parser Generator. Online document, `http://www.gnu.org/software/bison/`, accessed 29 Nov. 2010.

[gsl11] GNU Scientific Library, version 1.15, 2011. Online document, `http://www.gnu.org/s/gsl/`, accessed 23 Nov. 2011.

[Hau06] H. Hauser. Generalizing Focus+context Visualization. *Scientific visualization: The visual extraction of knowledge from data*, pages 305–327, 2006.

[HBL99] K.F. Harsh, V.M. Bright, and YC Lee. Solder Self-Assembly for Three-Dimensional Microelectromechanical Systems. *Sensors and Actuators A: Physical*, 77(3):237–244, 1999.

[HLRS+08] M. Hadwiger, F. Laura, C. Rezk-Salama, T. Höllt, G. Geier, and T. Pabel. Interactive Volume Exploration for Feature Detection and Quantification in Industrial CT Data. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1507–1514, 2008.

[Inc10a] Kitware Inc. ParaView, 2010. `http://www.paraview.org/`, accessed Feb. 18 2014.

[Inc10b] Kitware Inc. *VTK User's Guide*. Kitware Inc., mar 2010.

[Ins11] McKinsey Global Institute. Big data: The next frontier for innovation, competition, and productivity, 2011. Online document, accessed 18 July 2013, `http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation`.

[JC12] S.A. Jones and S.J. Cox. On the Effectiveness of a Quasi-Static Bubble-Scale Simulation in Predicting the Constriction Flow of a Two-Dimensional Foam. *J. Rheol.*, 56:457–471, 2012.

[JDS+11] S.A. Jones, B. Dollet, N. Slosse, Y. Jiang, S.J. Cox, and F. Graner. Two-dimensional constriction flows of foams. *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, 382(1-3):18 – 23, 2011.

[KDK+00] A.H. König, H. Doleisch, A. Kottar, B. Kriszt, and E. Gröller. AlVis-An Aluminium-Foam Visualization and Investigation Tool. In *Visualization (VisSym), EG/IEEE TCVG Symposium on*. Amsterdam, The Netherlands, 2000.

[KLM+08] J. Kehrer, F. Ladstadter, P. Muigg, H. Doleisch, A. Steiner, and H. Hauser. Hypothesis Generation in Climate Research with Interactive Visual Data Exploration. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1579–1586, Nov.-Dec. 2008.

[KMv08] G. Katgert, M.E. Möbius, and M. van Hecke. Rate Dependence and Role of Disorder in Linearly Sheared Two-Dimensional Foams. *Phys. Rev. Lett.*, 101:058301, 2008.

[Lip13] Dan Lipsa. FoamVis, 2013. Online document, accessed 25 June 2013, `http://csgalati.swansea.ac.uk/foam/html/`.

[LL13] Dan R. Lipşa and Robert S. Laramee. FoamVis, A Visualization System for Foam Research: Design and Implementation. Technical report, Swansea University, 2013.

[LLC+11] Dan R. Lipşa, Robert S. Laramee, Simon J. Cox, Jonathan C. Roberts, and Rick Walker. Visualization for the Physical Sciences. In *Eurographics*, pages 49–73, Llandudno, Wales, UK, April 2011. State-of-the-Art Reports.

[LLC+12] Dan R. Lipşa, Robert S. Laramee, Simon J. Cox, Jonathan C. Roberts, Rick Walker, Michelle A. Borkin, and Hanspeter Pfister. Visualization for the Physical Sciences. *EG Computer Graphics Forum*, 31(8):2317–2347, December 2012.

[LLCD11] Dan R. Lipşa, Robert S. Laramee, Simon J. Cox, and I. Tudur Davies. FoamVis: Visualization of 2D Foam Simulation Data. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2096–2105, October 2011.

[LLCD13a] Dan R. Lipşa, Robert S. Laramee, Simon Cox, and I. Tudur Davies. Visualizing 3D Time-Dependent Foam Simulation Data. In *Lecture Notes in Computer Science, International Symposium on Visual Computing (ISVC)*, pages 255–265, Rethymnon, Crete, Greece, July 2013.

[LLCD13b] Dan R. Lipşa, Robert S. Laramee, Simon J. Cox, and I. Tudur Davies. A Visualization Tool For Foam Research. In *NAFEMS World Congress (NWC) Conference Proceedings*, page 141, Salzburg, Austria, June 2013.

[LLCD13c] Dan R. Lipşa, Robert S. Laramee, Simon J. Cox, and I. Tudur Davies. Comparative Visualization and Analysis of Time-Dependent, 2D Foam Simulation Data. Technical report, Swansea University, 2013.

[Lor04] B. Lorensen. On the Death of Visualization. In *Position Papers NIH/NSF Proc. Fall 2004 Workshop Visualization Research Challenges*, 2004.

[LTKF08] F. Losasso, J.O. Talton, N. Kwatra, and R. Fedkiw. Two-Way Coupled SPH and Particle Level Set Fluid Simulation. *Visualization and Computer Graphics, IEEE Transactions on*, 14(4):797 –804, 2008.

[MHG10] M.M. Malik, C. Heinzl, and M.E. Groeller. Comparative visualization for parameter studies of dataset series. *Visualization and Computer Graphics, IEEE Transactions on*, 16(5):829 –840, sept.-oct. 2010.

[Mic13] Microsoft. Model-View-Controller, 2013. Online document, accessed 25 June 2013, `http://msdn.microsoft.com/en-us/library/ff649643.aspx`.

[Mun09] Tamara Munzner. *Visualization*, chapter 27, pages 675–701. AK Peters/CRC Press, 2009.

[NRRW11] Feng Niu, Benjamin Recht, Christopher Ré, and Stephen J Wright. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *Advances in Neural Information Processing Systems*, 24:693–701, 2011.

[PK96] Robert K Prud'Homme and Saad A Khan. *Foams: Theory, Measurements, and Applications*, volume 57. CRC PressI Llc, 1996.

[Pla73] J.A. Plateau. *Statique Expérimentale et Théorique des Liquides Soumis aux Seules Forces Molécularies*. Paris: Gauthier-Villars, 1873.

[PP95] H.G. Pagendarm and F.H. Post. Comparative Visualization - Approaches and Examples. In M. Göbel, H. Müller, and B. Urban, editors, *Visualization in Scientific Computing*, chapter 2. Springer, Wien, 1995.

[PTVB07] W.H. Press, S.Al Teukolsky, W.T Vetterling, and Flannery B.P. *Numerical recipes, The art of scientific computing, Third Edition*. Cambridge University Press, 2007.

[PW95] H.-G. Pagendarm and B. Walter. Competent, compact, comparative visualization of a vortical flow field. *Visualization and Computer Graphics, IEEE Transactions on*, 1(2):142 –150, jun 1995.

[RIV10] RIVIC. Research Institute of Visual Computing, 2010. `http://www.rivic.org.uk/`, accessed Sep. 24 2013.

[Rob07] J.C. Roberts. State of the art: Coordinated multiple views in exploratory visualization. In *Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV '07. Fifth International Conference on*, pages 61 –71, July 2007.

[RTTD04] M. Rayner, G. Trägårdh, C. Trägårdh, and P. Dejmek. Using the Surface Evolver to Model Droplet Formation Processes in Membrane Emulsification. *Journal of colloid and interface science*, 279(1):175–185, 2004.

[Smi52] C.S. Smith. Grain shapes and other metallurgical applications of topology. In *Metal Interfaces*, pages 65–108. American Society for Metals, Cleveland, OH, 1952.

[SRK08] R. Shimada, S. Rahman, and Y. Kawaguchi. Simulating the Coalescence and Separation of Bubble and Foam by Particle Level Set Method. In *Computer Graphics, Imaging and Visualisation, 2008. CGIV '08. Fifth International Conference on*, pages 18 –22, 2008.

[Sto51] George Gabriel Stokes. On the Effect of the Internal Friction of Fluids on the Motion of Pendulums. *Trans. Camb. Phil. Soc.*, 9:8–149, 1851.

[sur04] Surface Evolver Workshop, Apr. 2004. Online document, accessed 1 Dec. 2010, `http://www.susqu.edu/brakke/evolver/workshop/workshop.htm`.

[sur08] The Surface Evolver, Jan. 2008. Online document, accessed 29 Nov. 2010, `http://www.susqu.edu/brakke/evolver/html/evolver.htm`.

[SWND06] Dave Shreiner, Mason Woo, Jachie Neider, and Tom Davis. *OpenGL Programming Guide, Fifth Edition*. Addison Wesley, 2006.

[Tay76] Jean E. Taylor. The Structure of Singularities in Soap-Bubble-Like and Soap-Film-Like Minimal Surfaces. *Annals of Mathematics*, 103(3):489–539, 1976.

[Tel08] A. C. Telea. *Data Visualization, Principles and Practice*. A. K. Peters, Ltd., Wellesley, Massachusetts, USA, 2008.

[Ď01] R. Ďurikovič. Animation of Soap Bubble Dynamics, Cluster Formation and Collision. *EG Computer Graphics Forum*, 20(3):67–75, 2001.

[vH13] Dimitri van Heesch. Doxygen, 2013. Online document, accessed 26 June 2013, `http://www.stack.nl/~dimitri/doxygen/`.

[VW05] J.J. Van Wijk. The value of visualization. In *Proc. IEEE Visualization*, volume 2005, pages 79–86, 2005.

[Wei14] Eric W. Weisstein. Method of Steepest Descent., February 2014. From MathWorld - A Wolfram Web Resource. `http://mathworld.wolfram.com/MethodofSteepestDescent.html`, accessed 22 Feb. 2014.

[WGK10] Matthew Ward, Georges Grinstein, and Daniel Keim. *Interactive Data Visualization. Foundations, Techniques, and Applications*, chapter 10, pages 315–334. A K Peters, Ltd., Natick, Massachussetts, 2010.

[WH99] D. Weaire and S. Hutzler. *The Physics of Foams*. Oxford University Press, Oxford, 1999.

[Wyn09] A. Wyn. *Topological changes in sheared two-dimensional foams*. PhD thesis, Institute of Mathematics & Physics, Aberystwyth University, 2009.

[ZWSL10] Martin Zinkevich, Markus Weimer, Alexander J Smola, and Lihong Li. Parallelized Stochastic Gradient Descent. In *NIPS*, volume 4, page 4, 2010.