Eurographics Conference on Visualization (EuroVis) (2014)
N. Elmqvist, M. Hlawitschka, and J. Kennedy (Editors)

Evaluation of Visualizations for Interface Analysis of SPH

M. Krone[1], M. Huber[1], K. Scharnowski[1], M. Hirschler[2], D. Kauker[1], G. Reina[1], U. Nieken[2], D. Weiskopf[1], and T. Ertl[1]

[1]Visualization Research Center, University of Stuttgart, Germany
[2]Institute of Chemical Process Engineering, University of Stuttgart, Germany

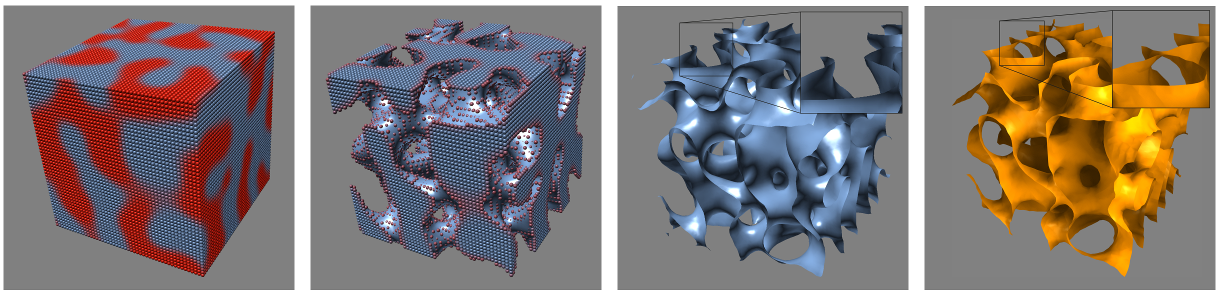

Figure 1: *The leftmost image shows the particle positions of the SPH simulation. Two phases are discernible (red and blue). The two center images show the isosurface that separates the two phases. The surface extraction is based on the kernel function that was used in the underlying simulation. The rightmost image shows a surface that was extracted using the method of Onderik et al. [OCD11]. The magnifications in the two images to the right show a case where the two methods create different surfaces.*

Abstract

We present a GPU-accelerated visualization application that employs methods from computer graphics and visualization to analyze SPH simulations from the field of material science. To this end, we extract the isosurface that separates the stable phases in a fluid mixture via the kernel function that was used by the simulation. Our application enables the analysis of the separation process using interactive 3D renderings of the data and an additional line chart that shows the computed surface area over time. This also allows us to validate the correctness of the simulation method, since the surface area can be compared to the power law that describes the change in area over time. Furthermore, we compare the isosurface that is based on the simulation kernel with an established method to extract smooth high-quality SPH surfaces. The comparison focuses on demonstrating the applicability for data analysis in the context of material science, which is based on the resulting surface area and how well the two phases are separated with respect to the original particles. The evaluation was carried out together with experts in material science.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Boundary representations, J.2 [Computer Applications]: Physical Sciences and Engineering—Chemistry

1. Introduction

Polymer membranes are widely used in chemical engineering applications such as in battery systems. They are semi-permeable, thus allowing only one specific substance to pass through. To develop and improve membranes it is recommended predicting the morphology in dependence of the manufacturing conditions. In the preparation process, the so-called phase inversion plays an important role. If a homogeneous fluid mixture separates into two stable phases, it is called phase inversion. One way to describe phase inversion is to use the Cahn-Hilliard equation [CH58], which is a fourth-order partial differential equation for diffusive mass

DOI: 10.2312/eurovisshort.20141166

transport. We discretize it with the Smoothed Particle Hydrodynamics (SPH) method [HHS*14]. For simplicity, we consider a binary, isothermal, incompressible, and equimolar fluid mixture to validate the dynamics of 3D simulations. In general, SPH particles move on their trajectory. In this special case, the momentum per particle is constant and the particles are fixed at their initial positions. Therefore, we only solve the transport equation for the concentration of one component. Nevertheless, the position of the interface between two phases changes in time. Therefore, the analysis of surfaces should also be accurate for moving particles.

We present a visualization application that allows us to analyze the membrane morphology for three-dimensional SPH simulations. Our application interactively visualizes the underlying phase separation process. This allows us to visually analyze the morphology of the simulated membrane. For a qualitative and quantitative analysis, we compute the total area of the extracted surfaces. The development of the surface area over time is plotted in a line chart. This line chart can also be used to verify the simulation method by comparing it to an analytic power law that describes the decrease of the surface area for the specific phase separation. We also compare two established surface extraction methods for SPH simulations. The comparison of the respective surface areas allows us to assess the applicability of the two surface extraction methods for SPH. Our application was developed in close contact with collaborators working in the field of material science to satisfy their requirements.

2. SPH Basics

SPH is a Lagrangian, particle-based, and mesh-free simulation method. Originally developed for astrophysical problems [GM77], the relevance of SPH in engineering science is steadily increasing [Mon12]. In SPH, a quantity $A(\mathbf{x})$ is computed by interpolation of weighted quantities $A(\mathbf{x}')$ in a space V_h using the weighting function W.

$$A(\mathbf{x}) = \int_{V_h} A(\mathbf{x}')W(h, \mathbf{x}-\mathbf{x}')dx', \tag{1}$$

where h is the smoothing length. The kernel function W used in this paper is a Wendland kernel. In the discrete formulation of Eq. 1, each particle represents one element of fluid. A transition to discrete formulation leads to

$$A(\mathbf{x}_i) \approx \sum_j \frac{m_j}{\rho_j} A(\mathbf{x}_j) W(h, x_{ij}). \tag{2}$$

x_{ij} is the distance between particle i and j. m_j and ρ_j are mass and density of particle j. We choose the Wendland kernel with a smoothing length $h = 1.55L_0$ (L_0 is the initial particle spacing), which is a good choice regarding accuracy and performance according to [Mon12]. The Wendland kernel is an ideal candidate for SPH smoothing kernels [DA12]. For a more detailed discussion of the SPH simulation method, we refer to [Mon12].

3. Related Work

In the context of rendering surfaces of particle-based (SPH) simulation data, different surface extraction methods have been proposed. Müller et al. [MCG03] defined the surface using a weighted density field of the particles. However, their approach tends to produce bumpy surfaces. Zhu and Bridson [ZB05] used a distance field to achieve smooth surfaces. Since this method can produce artifacts in irregularly sampled regions and between isolated particles, Solenthaler et al. [SSP07] and Onderik et al. [OCD11] proposed modifications to solve this issue. Recently, Yu and Turk [YT13] have presented a technique based on a density field that uses anisotropic kernels to produce smooth surfaces. To speed up these computationally expensive methods, Akinci et al. [AIAT12] presented methods for the optimization and parallelization that work with common surface reconstruction techniques. The aforementioned methods focus on the extraction of smooth, visually pleasing surfaces for fluid simulations.

In the context of simulation visualization for analysis, Schindler et al. [SFWP11] introduced *marching correctors*, a variant of the marching cubes algorithm. Kolb and Cuntz [KC05] generated a uniform density volume of the particles on the GPU and used point sprites for rendering. Goswami et al. [GSSP10] presented a CUDA-based SPH simulation. They visualized their simulation results by creating a distance field from the particles and rendered it using GPU-based algorithms. Fraedrich et al. [FAW10] presented a method to visualize very large SPH simulations using an adaptive view-dependent discretization of the simulation domain to sample the particle densities. In the work of Molchanov et al. [MFR*13], a feature-rich interactive framework for the analysis of SPH simulations in the application area of astrophysics was described. Besides rendering point-clouds of the simulated data, they use a splatting technique to represent isosurfaces. In contrast to these methods that focus on the visualization of the SPH simulation, we also want to verify the simulation based on the surface area.

4. Algorithms & Implementation

In this section, we briefly explain the two methods to extract SPH surfaces that we compared.

For the SPH simulation we use a Wendland kernel function. For visualization, we weight the particles with this kernel function and compute a density field that represents the simulation data. The resulting density ρ at point $\mathbf{P} \in \mathbb{R}^3$ is

$$\rho(\mathbf{P}) = \sum_j m_j \frac{21}{2\pi h^3} \cdot \left(1 - \frac{r}{h}\right)^4 \cdot \left(4\frac{r}{h} + 1\right), \tag{3}$$

where $r = \left\|\mathbf{P} - \mathbf{P}_j\right\|$, $\mathbf{P}_j$ is the position of particle j, and the smoothing length $h = 1.55L_0$. If $r \geq h$, particle j does not contribute to the density ρ at point $\mathbf{P}$. That is, only neighboring particles within the cutoff radius of $2h = 3.1L_0$

contribute to $\rho(\mathbf{P})$, as in the simulation. We implemented the computation of the density volume using CUDA. Each point $\mathbf{P}$ in the density field (that is, each voxel) has a limited neighborhood that is defined by h. For m voxels, the density ρ can be computed in $\mathcal{O}(m)$. Since ρ of each voxel can be computed independently of all other voxels, the computation is embarrassingly parallel. All particles are sorted into a uniform acceleration grid with a grid spacing equal to h. For each voxel, only $3 \times 3 \times 3$ grid cells have to be evaluated to find all neighboring particles $\mathbf{P}_j : \left\|\mathbf{P} - \mathbf{P}_j\right\| < h$. Our implementation is based on the work of Krone et al. [KSES12], who presented an optimized CUDA implementation that uses a Gaussian density kernel. Their method is available in the open source software VMD [HDS96] and includes an optimized CUDA marching cubes implementation (see [KSES12] for details).

As an alternative definition of the SPH surface, a method based on the approach of Zhu and Bridson [ZB05] is used. The idea is to calculate a scalar value at a given point $\mathbf{P}$ by measuring the distance to a weighted sum of the neighboring particle centers $\bar{\mathbf{P}}$. As discussed in [SSP07] and [OCD11], it is possible that the weighted centers are located outside of the desired surface, which can lead to extensive visual artifacts. Therefore, we use the modified implicit surface definition of Onderik et al. [OCD11]:

$$\phi(\mathbf{P}) = \|\mathbf{P} - C(\mathbf{P})\| - Rf(\mathbf{P}), \tag{4}$$

where R controls the distance of the surface to the boundary particles. The weighted sum of the neighboring particle centers is calculated using normalized particle averages with

$$C(\mathbf{P}) = \frac{\sum_j \frac{1}{w_j} \mathbf{P}_j W\left(\|\mathbf{P} - \mathbf{P}_j\|, h\right)}{\sum_j \frac{1}{w_j} W\left(\|\mathbf{P} - \mathbf{P}_j\|, h\right)}, \tag{5}$$

where w_j is determined using the SPH interpolation of the positions of the particles' neighbors and the polynomial smoothing kernel $W(r,h) = (1 - (r^2/h^2))^3$ is used. Furthermore, R is multiplied with a decay function $f(\mathbf{P})$ as defined in [OCD11] to eliminate further artifacts. The isosurface is also extracted using marching cubes.

The total surface area of the extracted isosurfaces is computed by summing up all individual triangle areas. An overview of related, more advanced methods can be found in [DCM13]. For each simulation frame, the surface area is plotted over time in a 2D line chart (see Figure 2).

5. Results & Discussion

We measured the performance of the CUDA implementation that computes the Wendland kernel density volume using four SPH simulations of increasing size. The test system was an Intel Core i7 (3.6 GHz) with 32 GB RAM, and an Nvidia GTX Titan (6 GB VRAM). Table 1 shows the results of our measurements for two different volume resolutions (grid cell length L_0 and $L_0/2$). Please note that, although all frames are processed once at startup to get the

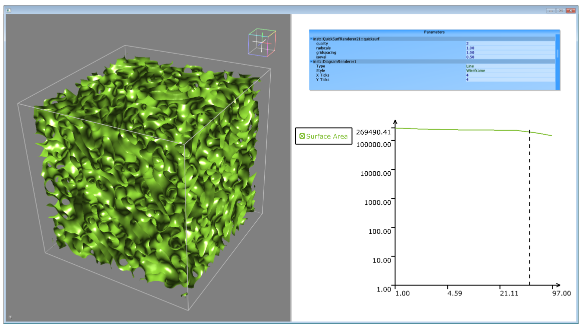
Parameters
- inst::QuickSurfRenderer21::quicksurf
  - quality 2
  - radscale 1.00
  - gridspacing 1.00
  - isoval 0.50
- inst::DiagramRenderer1
  - Type Line
  - Style Wireframe
  - X Ticks 4
  - Y Ticks 4

Figure 2: *Screensot of our application showing an SPH simulation of 10^6 particles. The line chart to the right shows the area of the extracted surface over time.*

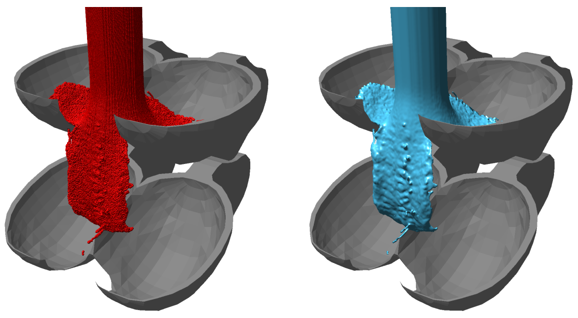

Figure 3: *Fluid simulation comprising 800k particles (simulation of the impact of a free jet). The left image shows the particles as spheres, the right image shows the extracted surface using the Wendland kernel.*

values for the line chart, the resulting meshes are not stored. That is, the whole computation pipeline is executed for each rendered frame in order to keep the memory requirements low. Therefore, the user can select any frame for visualization or view an interactive animation of the simulation without precomputation. As all computations run in real-time, it can also be used to analyze or monitor a running simulation. Figure 3 shows a typical fluid simulation comprising 800k particles. The frame rates are slightly lower than for the largest data set in Table 1, since the volume has a much higher resolution due to the large, unoccupied areas. Modern SPH simulation methods can simulate over 40M particles (e.g. [ICS*13]). Here, not only the computation speed but also the GPU memory becomes a limiting factor. Our method could, however, be trivially parallelized for multiple GPUs (e.g. a compute cluster) by dividing the data set into uniform bricks. Only particles within the cutoff radius h would have to be replicated between bordering instances.

We did not measure the performance of the surface definition based on the distance field [OCD11], since our current implementation runs on the CPU. It can be implemented analogously to the density volume computation, since it also

Table 1: *Performance measurements (all timings in milliseconds). #P is the particle count, #Δ is the number of surface triangles, t_ρ denotes the time to upload all particles P to the GPU, sort them into the acceleration grid, and compute the density ρ, t_{MC} is the calculation time for the marching cubes and its surface area. The overall performance (computation + rendering) is given in averaged frames per second (fps).*

#P	Volume Size	#Δ	t_ρ	t_{MC}	fps
125k	50×50×50	∼50k	10	2	67
	99×99×99	∼170k	16	4	37
250k	63×63×63	∼150k	15	3	39
	125×125×125	∼650k	31	8	18
500k	79×79×79	∼300k	27	4	22
	157×157×157	∼1.3M	59	12	9
1M	100×100×100	∼650k	53	6	11
	199×199×199	∼2.5M	119	18	5

considers only a neighborhood of particles for each voxel. However, we anticipate that it would reach a slightly lower performance due to the higher computational cost.

We compared the surface area based on the Wendland kernel with the analytical function that describes the power law. The growth rate of the area corresponds satisfactorily to the power law. The relative discretization error stemming from the lower grid resolution is negligible ($\sim 10^{-3}$). Figure 4 shows a line chart with the surface areas of the four simulations plotted over time and the power law for reference. We also compared the surface based on the Wendland kernel with the alternative surface based on the distance field. As observable in Figure 4, the distance-based method (dashed red curve) closely matches the corresponding area of the Wendland kernel (black curve). Consequently, it also satisfies the power law. That is, both methods can be used for a qualitative analysis of the SPH simulation data of our project partners. However, the results for the two surfaces differ by about 1% of the total surface area in some cases. Thus, the deviation of the distance-based method is too high for a quantitative analysis in some cases. Furthermore, the distance-based methods constructs surfaces that do not exactly represent the simulation in some cases. Figure 1 shows an example where this is the case, even though the overall surface area differs by less than 1%.

Our visualization application enables the validation of the SPH simulations developed by our collaborators from material science. It also allows them to visually analyze the morphology of the simulated membranes. Our collaborators were very pleased with the results and the possibilities for simulation verification and data analysis.

6. Summary & Future Work

We presented an interactive visualization application for SPH simulations. Our application is tailored to illustrate the

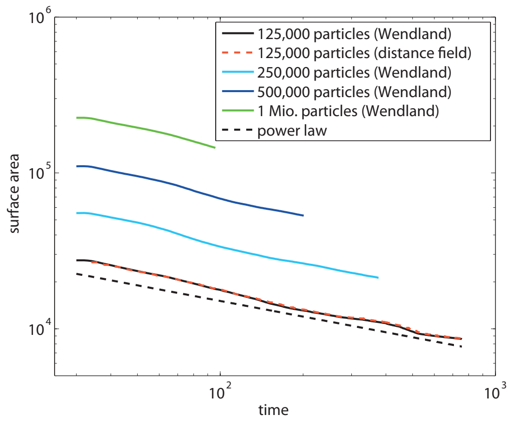

Figure 4: *Surface area over time for different particle numbers using the volume computation based on the Wendland kernel (both axis have logarithmic scale). The decrease of the area adheres to the power law for all simulations (dashed black line). Note that the larger data sets were simulated over shorter periods. For 125,000 particles, the chart also shows the curve for the alternative surface based on the distance field (dashed red curve).*

phase separation in a fluid mixture. The density contributions of the SPH particles are sampled on a grid and an isosurface is extracted. Since the visualization uses the same density function as the simulation, it faithfully depicts the simulation results. A line chart shows the surface area over time. By extracting the surface area, we were able to verify the correctness the phase separation simulation of our collaborators, since the area has to adhere to a given power law.

Furthermore, we compared this density-based surface to a method that was designed to extract a smooth, visually pleasing surface of a SPH simulation [OCD11]. We were specifically interested in the suitability of the this method for data analysis. The alternative method sometimes extracts slightly different surfaces than the density-based method. However, against our expectations, the surface area has a low error rate and adheres to the aforementioned power law. It can, therefore, be used for qualitative analysis. For quantitative analysis, though, the density-based surface is preferable.

In the future, we want to extract and visualize additional characteristics of the simulated material. One example would be to compute the percolation rate of the membrane using a Reeb graph or centerline extraction. We also want to investigate the feasibility of other high-quality SPH surface extraction methods for scientific analysis.

Acknowledgments: This work was partially funded by the German Research Foundation (DFG) as subprojects A.6, D.3, D.4 and D.5 of collaborative research center SFB 716. We want to thank our colleagues F. Beck and P. Eberhard (SFB 716, A.5) for providing the data set shown in Figure 3.

References

[AIAT12] Akinci G., Ihmsen M., Akinci N., Teschner M.: Parallel surface reconstruction for particle-based fluids. *Computer Graphics Forum 31*, 6 (2012), 1797–1809. 2

[CH58] Cahn J. W., Hilliard J. E.: Free energy of a nonuniform system. I. Interfacial free energy. *Journal of Chemical Physics 28 (2)* (1958), 258–267. 1

[DA12] Dehnen W., Aly H.: Improving convergence in smoothed particle hydrodynamics simulations without pairing instability. *Monthly Notices of the Royal Astronomical Society 425*, 2 (2012), 1068–1082. 2

[DCM13] Duffy B., Carr H., Möller T.: Integrating isosurface statistics and histograms. *IEEE Transactions on Visualization and Computer Graphics 19*, 2 (2013), 263–277. 3

[FAW10] Fraedrich R., Auer S., Westermann R.: Efficient high-quality volume rendering of SPH data. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1533–1540. 2

[GM77] Gingold R. A., Monaghan J. J.: Smoothed particle hydrodynamics: theory and application to non-spherical stars. *MNRAS 181* (1977), 375–389. 2

[GSSP10] Goswami P., Schlegel P., Solenthaler B., Pajarola R.: Interactive SPH simulation and rendering on the GPU. In *Proceedings of the 2010 ACM SIGGRAPH/EG Symposium on Computer Animation* (2010), pp. 55–64. 2

[HDS96] Humphrey W., Dalke A., Schulten K.: VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics 14* (1996), 33–38. 3

[HHS*14] Hirschler M., Huber M., Säckel W., Kunz P., Nieken U.: Application of the cahn–hilliard approach to smoothed particle hydrodynamics. *Mathematical Problems in Engineering 2014* (2014). 2

[ICS*13] Ihmsen M., Cornelis J., Solenthaler B., Horvath C., Teschner M.: Implicit incompressible sph. *IEEE Transactions on Visualization and Computer Graphics 99* (2013). 3

[KC05] Kolb A., Cuntz N.: Dynamic particle coupling for GPU-based fluid simulation. In *Proceedings of Symposium on Simulation Technique* (2005), pp. 722–727. 2

[KSES12] Krone M., Stone J. E., Ertl T., Schulten K.: Fast visualization of Gaussian density surfaces for molecular dynamics and particle system trajectories. In *EuroVis - Short Papers* (2012), pp. 67–71. 3

[MCG03] Müller M., Charypar D., Gross M.: Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), pp. 154–159. 2

[MFR*13] Molchanov V., Fofonov A., Rosswog S., Rosenthal P., Linsen L.: SmoothViz: an interactive visual analysis system for SPH data. In *Proceedings of the 8th International SPHERIC Workshop* (2013), pp. 350–356. 2

[Mon12] Monaghan J. J.: Smoothed particle hydrodynamics and its diverse applications. *Annual Review of Fluid Mechanics 44* (2012), 323–346. 2

[OCD11] Onderik J., Chládek M., Durikovic R.: SPH with small scale details and improved surface reconstruction. In *Proceedings of the Spring Conference on Computer Graphics* (2011), pp. 29–36. 1, 2, 3, 4

[SFWP11] Schindler B., Fuchs R., Waser J., Peikert R.: Marching correctors - fast and precise polygonal isosurfaces of SPH data. *in Proceedings of the 6th International SPHERIC Workshop* (2011), 125–132. 2

[SSP07] Solenthaler B., Schläfli J., Pajarola R.: A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds 18*, 1 (2007), 69–82. 2, 3

[YT13] Yu J., Turk G.: Reconstructing surfaces of particle-based fluids using anisotropic kernels. *ACM Transactions on Graphics 32*, 1 (2013), 5:1–5:12. 2

[ZB05] Zhu Y., Bridson R.: Animating sand as a fluid. *ACM Transactions on Graphics 24*, 3 (2005), 965–972. 2, 3