UNIVERSITÄT SIEGEN

Computergraphik und Multimediasysteme CG

Efficient SPH-based Simulation and Rendering of Fluid Transport Dynamics

Effiziente SPH basierte Simulation und Darstellung von Fluid Transport Dynamiken

bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen

zur Erlangung des Grades eines
Doktors der Ingenieurswissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

Jens Orthmann
Siegen, 06. Juni 2014

1.Gutachter: Prof. Dr. Andreas Kolb, Universität Siegen
2.Gutachter: Prof. Dr. Matthias Teschner, Universität Freiburg

Verteidigt am 14.November 2014

Printed on non-aging wood- and acid-free paper.
Gedruckt auf alterungsbeständigem holz- und säurefreiem Papier.

Acknowledgments

I am grateful for the support of all those who made this thesis possible.

First and foremost, I would like to thank Prof. Dr. Andreas Kolb for giving me the privilege of writing a dissertation at the Computer Graphics and Multimedia Systems Group at the University of Siegen. I have been very lucky to have such an attentive and supportive mentor who always kept the door open. I appreciate his trust and patience over the years. If it wasn't for his excellent education during my years as a Computer Graphics student and his outstanding supervision during my time as a Ph.D. student, I would not have succeeded to the degree that I have.

I am also thankful to Prof. Dr. Matthias Teschner of the Graphics Group at the University of Freiburg for accepting the co-advisorship of my Ph.D. thesis. I am very lucky to have such a highly renowned researcher in the field of computer animation as an examiner of my work. Furthermore, I would like to thank Dr. Markus Ihmsen of the Freiburg team for giving me the opportunity of being a co-author in his STAR report and for providing the contact with SPH experts including Barbara Solenthaler.

My work has been partially funded by the Siegener Graduate School for the "Development of Integral Heterosensor Architectures for the n-Dimensional (Bio)chemical Analysis," and I would like to knowledge all the partaking groups for giving tutorials and providing me with introductory knowledge, especially Remo Winz for the many fruitful conversations we had.

I also would like to thank my colleagues at the Computer Graphics lab in Siegen.

My sincere thanks go to Prof. Dr. Christof Rezk Salama who showed me the fun in Computer Graphics. With his expertise within the field and with his endless support, he has been a great asset to the group. We have missed him ever since he left to become a professor at Trier University of Applied Sciences.

Many thanks go to Hendrik Hochstetter for being a comrade during our last (oversea) paper deadlines and for reading and commenting on several versions of this thesis. Even in this short but intense time we shared at Siegen University, we have become very close, and I am truly grateful for his open-mindedness and selfless nature.

I especially would like to thank Julian Bader for co-authoring one of my publications and for his friendliness. I appreciate him ensuring that I get out of the office once in a while and the many good times we had during these off hours. I trust him to remember to feed the birds in winter.

A deep gratitude goes to my former office mate Maik Keller, to whom I feel a deep connection. Apart from the papers we wrote, he always was supportive and prevented me from going insane. I have been very fortunate to have such a great character at my side. He truly is a rock star. Also, many thanks to his wife Kathrin who often helped out correcting my English.

Many thanks go to Nicolas Cuntz, who provided me with a kick start and who I have the honour to work with in my new position.

Special thanks goes to our secretary Willi Gräfrath for handling the many administrative tasks and for lending his remarkable voice to so many videos.

I am also grateful to Serkan Bayraktar and Martin Pätzold for helping me out with the editing of my latest videos.

I also would like to thank Martin Lambers, Lisa Brückbauer and Björn Labitzke, with whom I have mentored many lectures and for the many nice chats during the countless coffee breaks.

Of course, this dissertation would simply not exist if it wasn't for my parents Agnes and Anton Orthmann, my brothers Dirk and Peter Orthmann, and the love of my life Danica Brenner. There is simply not enough space to put my gratitude for them into words.

– Jens Orthmann
May 2014

Abstract

Transported substances, such as salt, water colors and detergents, not only change fluid properties and the fluid's natural appearance, but they also interact with surrounding materials. Naturally, physically-based simulation of transport phenomena can be an important tool for visual effects. However, even though Smoothed Particle Hydrodynamics (SPH) provide state-of-the art solutions for fluid motion in unbounded free-surface scenarios, fluid transport phenomena have seldom been addressed. Hence, the main aim of this work is to advance SPH-based fluid animation by developing efficient simulation and rendering mechanisms for fluid transport processes. Since properties of SPH are linked to field interpolations, this thesis proposes three novel sampling techniques that target an efficient reconstruction of quantity fields:

First, this work addresses issues that arise from coupling surface transport and bulk transport. Therefore, the surface is modeled by stably sampling a surface delta function with bulk particles, which results in a consistent representation of quantity fields in bulk and on surfaces. The embedded surface model fully avoids back-and-forth interpolation between bulk and surface, and, additionally, it lays the foundation for stable and symmetric boundary fluxes.

Second, the demand for high surface resolution leads to inefficiencies that are counteracted by an adaptive particle sampling within homogeneous regions in the fluid's bulk. Time coherent sampling, which is achieved via smooth blending of particle levels, reintroduces robustness, which is especially needed for incompressible fluids.

Third, the work features a sampling mechanism for an instant high-quality ray casting of particle fields. A greedy approach reveals strict error bounds for the underlying adaptive sampling mechanism and triggers cell merging in order to ensure cache-coherent particle access.

Error predictions stabilize integration time steps during simulation and they also preserve sharp features during ray casting. In addition to the aforementioned contributions, this thesis features a fully data-parallel implementation on currently-available Graphic Processing Units, but without limiting algorithms to hardware-specific features.

Zusammenfassung

Transportierte Substanzen, wie beispielsweise Salz, Wasserfarben und Waschmittel, ändern nicht nur die Eigenschaften und das Erscheinungsbild von Flüssigkeiten, sondern interagieren auch mit den sie umgebenden Materialien. Besonders die physikalisch-basierte Simulation dieser Transport-Phänomene kann ein wichtiges Werkzeug für visuelle Effekte sein. Obwohl Smoothed Particle Hydrodynamics (SPH) State-of-the-Art Lösungen für die Bewegung von Flüssigkeiten im Zusammenspiel mit offenen Simulationsbieten und freien Oberflächen bieten, stellen Aspekte des Flüssigkeitstransportes bisher ein Desiderat der Forschung dar. Das Hauptziel dieser Promotionsarbeit ist es, mittels effizienten Simulations- und Rendering Mechanismen für Flüssigkeitstransporteprozesse, die bisherige SPH-basierte Forschung im Bereich der Flüssigkeitsanimationen voranzutreiben. Da die Eigenschaften von SPH direkt an Feldinterpolationen gekoppelt sind, erarbeitet die vorliegende Qualifikationsarbeit drei neue Sampling-Techniken für eine effiziente Rekonstruktion von Partikelfeldern:

Erstens werden die bestehenden Transportgleichungen um einen entsprechenden Oberflächentransport erweitert. Um die Kopplung zwischen Bulk und Oberfläche zu erreichen wird die Oberfläche mittel einer impliziten Deltafunktion beschrieben. Das Sampling der impliziten Oberfläche mit Volumenpartikeln führt zu einer konsistenten Beschreibung der quantitativen Felder im Volumen und an der Oberfläche. Das eingebettete Oberflächenmodell vermeidet dabei eine vor-und-zurück Interpolation zwischen Volumen und Oberflächefeldern und legt zudem das Fundament für stabile und symmetrische Randbedingungen.

Zweitens wird durch ein adaptives Sampling in homogenen Bereichen der hohen Partikelzahl an der Oberfläche entgegengewirkt: Ein zeitkohärentes Sampling, welches durch ein kontinuierliches Überblenden von Partikelauflösungen erreicht wird, führt die nötige Robustheit ein, die besonders im Zusammenhang mit inkompressiblen Flüssigkeiten benötigt wird.

Drittens, stellt die Arbeit einen Samplingmechanismus für eine hochqualitätige Strahlenverfolgung von Partikelfelder ohne Vorverarbeitung vor. Der beschriebene Greedy-Ansatz enthüllt strikte Fehlergrenzen für den darunterliegenden adaptiven Samplingmechanismus und stellt einen Cache-kohärenten Partikelzugriff sicher.

Fehlervorhersagen stabilisieren dabei die Integrations-Zeitschritte während der Simulation, und erhalten Konturen während der Strahlenverfolgung. Abgesehen von den technischen Neuerungen bietet die vorliegende Dissertation eine komplette datenparallele Implementierung auf aktuell verfügbaren Graphikkarten (GPUs), ohne dabei die verwendeten Algorithmen auf hardware-spezifische Aspekte zu beschränken.

Contents

Chapter 1

Introduction

This introductory chapter presents the basic objectives and ideas of this thesis and motivates fluid transport dynamics as a valuable asset for particle-based fluid simulation. It further clarifies challenges in the numerical realization of free-surface transport using grid-free Lagrangian methods, and it outlines the major contributions of this thesis.

The field of Computer Graphics ranges from computer-aided design and data-visualization to computer animation including fluid simulation. In general, our major goal in Computer Graphics is to produce high-quality images by employing as few computational resources as possible. Very often, the visual effects and the gaming industry drive development of computer animation techniques with their ever-increasing demand for imitating natural phenomena. Among other phenomena, the simulation of fluids (liquids and gases) produces very appealing effects, which is due to the fluid's characteristic appearance that follows complex rules but is also easily recognizable. Due to the complex nature of these phenomena, fluid motion can seldom be animated by hand, and instead is controlled by means of physical simulation including initial and boundary conditions.

Usually, Computer Graphics literature focuses on the simulation of fluid movement and rendering of its geometric shape, i.e. the fluid's surface. Despite transport of momentum, transport of other quantities is very rare, even though it can provide a tremendous increase of realism and can enable a variety of new effects that can drastically improve the appearance of simulations as shown in Fig. 1.1. This thesis aims to fill this gap while it also provides efficient sampling techniques in the context of particle-based fluid simulations.

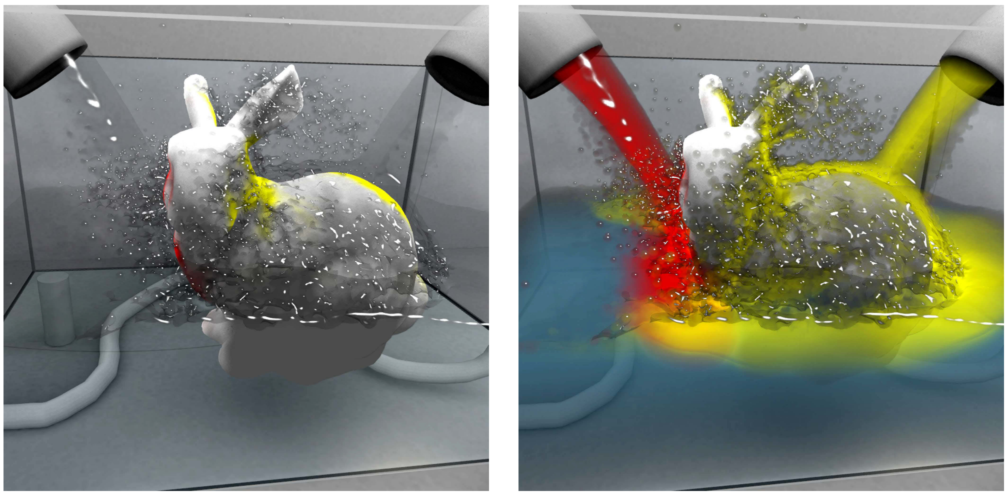


Figure 1.1: Transport of water color drastically improves the quality of the scene from simple showering to painting the Standford bunny.

1.1 Motivation

With any new generation of hardware, the complexity of simulations as well as the number of small-scale details increases in order to ever better capture the natural appearance of fluids. Over the last decade, fluid simulations have been enriched by increasingly more physical phenomena such as capillary waves, turbulence methods, and foam. To cope with these challenges, Computer Graphics (CG) applications presently employ physically-based formulations for fluid dynamics and usually adapt numerical techniques from the field of Computational Fluid Dynamics (CFD) in order to solve the underlying Ordinary Differential Equations (ODEs). One significant advantage of physically-based modelling is, that the complexity of the simulation is hidden behind meaningful and intuitive simulation parameters, such as fluid density and viscosity.

Physically-based Fluid Animation

In practice, however, the outcome of simulations is still hard to predict. Even animation experts often need to run multiple simulations with varying parameter settings in order to achieve the desired effect. Furthermore, interactive applications or small production cycles demand computationally efficient methods in order to provide fast feedback for users and effect artists. Robust algorithms supporting these methodologies are thus mandatory. For these reasons, high performance simulation and rendering of fluids is among the most challenging tasks in Computer Graphics.

Efficiency and Robustness

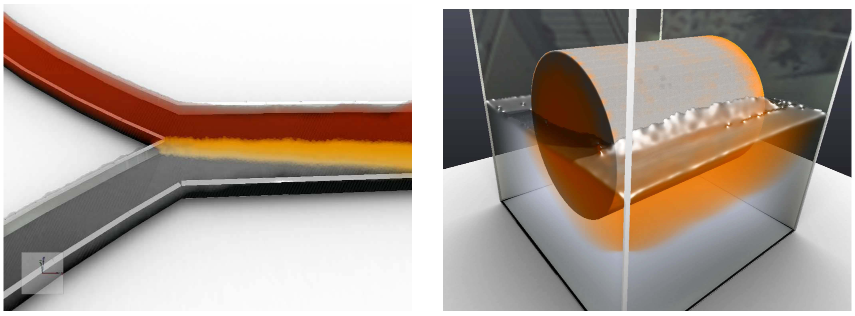


Figure 1.2: Micro mixing devices (left) and wetting effects (right) are two possible applications of fluid transport in biochemical engineering.

Fluid Transport

In transport processes, the fluid acts as a carrier for (soluble) substances [SKS05, BS09]. Motion of these substances occurs on two different length-scales: at macroscopic scales due to convection induced by the velocity field, and at microscopic scales due to diffusion induced by concentration differences. Depending on the type of transported substance, both flows may occur simultaneously, thus leading to total transport as a combination of convective and diffusive flux.

Biochemical Engineering

In many biochemical applications or engineering fields, fluid transport is an well-established and fundamental tool. Engineering examples of fluid transport include the movement of substances within ecosystems, the production of chemicals, cleansing effects, chaotic mixing in micro channels, and drug delivery in biochemical analysis. Often, transport of substances is only the first step and is followed by chemical reactions at phase contact lines. In micro mixing applications, for example, targets are guided toward a sensor where they are immobilized on a functionalized surface [HLS05]. As another example, a class of materials called surface active agents (surfactants) [RK04], which due to their amphiphilic nature, are attracted to the fluid-air interface where they reduce surface tension. In contact with solids, they reduce wetting effects [FAB*11] and increase cleansing effects [BT07a].

Interdisciplinarity

In the future, physically-based fluid animations in Computer Graphics, as shown in Fig. 1.2, may allow for quick parameter analysis before starting costly experiments or running time-consuming CFD simulations. In turn, transport models from engineering fields also drastically improve the simulation quality in Computer Graphics, as shown in Fig. 1.1. However, fluid transport dynamics have been, to date, marginally addressed in the community. The challenges that arise from transport problems therefore provide a strong motivation for this thesis.

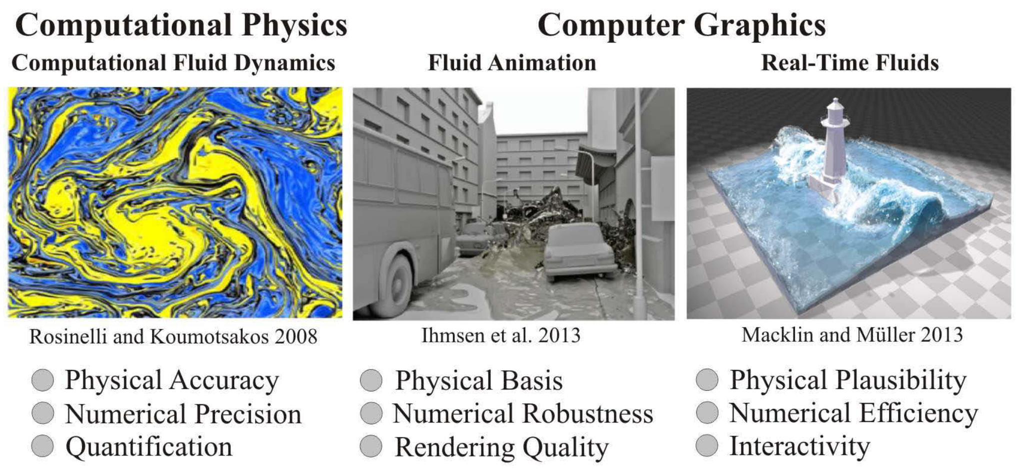


Figure 1.3: Characteristics and goals of particle-based fluid applications.

1.2 Techniques

While the field of particle-based fluid simulation, as shown in Fig. 1.3, comprises a wide range of applications, this thesis follows the major motif of Computer Graphics and prioritizes rendering quality and computation speed. It aims to produce visually attractive results without introducing excessive computational burden. Numerical accuracy, as important as it is in engineering applications, is not as critical as efficiency and robustness. The switch from numerical accuracy and scientific precision to simulation speed and visual appearance requires trade-offs between reliability and computational efficiency. Challenges arise from

From CFD to CG

- large integration time steps in combination with incompressible fluids,
- large number of computational elements, and
- coarse approximations of the underlying physics,
- parallelization on multi-core and data-parallel architectures
- reconstruction of smooth fields for rendering

Although, ideally physical parameters would be as close as possible to real fluids, material properties may be required to be significantly different from measured ones in order to support time step constraints.

For the sake of computational efficiency, it is important not to simulate the air phase. Instead, dynamics simply are described by a single fluid phase that defines a sharp interface, the so-called free surface. During the last two decades, a large variety of methods has been developed in order to

Free Surfaces

solve the governing equations of free-surface fluid motion in the context of physically-based computer animation.

Currently, grid-based or grid-free particle methods, which are compared later in Sec. 2.2, are state-of-the art for simulating fluids in free-surface scenarios. In the later, particles follow the trajectory of the fluid and they carry the physical properties that define the underlying fluid dynamics. Since particles adapt to the flow map, they trivially conserve mass in combination with free surface flows. As such, Smoothed Particle Hydrodynamics (SPH), as simultaneously introduced by Lucy [Luc77], and Gingold and Monaghan [GM77], have gained increasing popularity over the last decade. Given a sufficiently large number of fluid particles, most complex fluid dynamics can be reduced to simple interactions of particle pairs that naturally map to data-parallel architectures as provided by currently available Graphics Processing Units (GPUs). Despite the time-step restrictions of SPH methods, their ability to handle free surfaces in a mass-conserving manner builds a robust foundation for the simulation of convection-dominated transport problems [Cle98, KCR08].

Smoothed Particle Hydrodynamics

1.3 Objectives

The main objectives of this dissertation are to contribute to the field with high performance, physically-based, and easy-to-control simulation and rendering techniques for fluid transport. We anticipate, that this research will make possible a variety of new effects such as evaporation and condensation (cf. Chap. 7.2).

Goals

From a high-level perspective, the motivation behind this thesis is to establish preliminary work for the ultimate future goal of having a seamless interactive transport framework that provides an instant response to changing simulation parameters for large particle numbers. From a detailed perspective, the presented work takes SPH-based fluid solvers one step further into this direction by improving stability of surfaces, and efficiency of the particle sampling, as well as by increasing the quality of instant particle-based rendering systems.

Design Decisions

In addition to these goals, all contributions in this thesis maintain the properties of current state-of-the art SPH-based fluid solvers in order to make the proposed techniques widely applicable. In particular, this requirement raises the following specific design considerations:

– **Soundness:** the physical basis of the simulation is of major importance. Formulations are as close as possible to the underlying continuum mechanics as described in Sec. 2.1.1 in order to remain consistent with the mathematical foundations and in order to minimize the number of design parameters.
– **Simplicity:** physics are formulated on particle pairs only, and global data dependencies are avoided. The simple data-parallel nature of SPH is therefore maintained. In fact, this work utilizes a fully GPU-based system (even if is is not limited to this system) in order to accentuate the efficiency of the proposed techniques.
– **Applicability:** this thesis utilizes a state-of-the art incompressibity solver as its major building block (cf. Sec. 3.3.1), which is capable of handling complex fluid-rigid interactions (cf. Sec. 3.2.4). To further increase usability, pre-processing is completely avoided, and instead a combined simulation and rendering system is proposed. Finally, algorithms are designed to be independent of specific hardware features such as rasterization units or shared memory capacities.

Restrictions

Since fluid simulation comprises a wide range of physical phenomena, the presented scenarios in this thesis are restricted to

– **Convection Dominated Flows:** in scales of meters down to millimeter convection usually exceeds diffusion. Since such scales are common in the field of Graphics, the proposed methods are especially well suited for momentum-dominated flows, as well as flows with complicated material interfaces.

– **Incompressible Fluids:** in general, incompressibility conditions lead to very restrictive integration time steps. To increase applicability, contributions of this thesis are thus solely based on incompressible fluids, even though all presented approaches can be easily applied to compressible fluids like gases as well. Furthermore, it is assumed that the mass of the transported substances is much smaller as the mass of the carrier fluid, which renders the fluid density invariant to soluted substances.
– **Isothermal Conditions:** usually, the fluid's density, the diffusion speed, as well as the speed of kinetic reactions, highly depend on the temperature of the system. However, effects related to temperature changes, including conservation of energy, are not within the scope of this thesis and thus, are neglected by assuming iso-thermal conditions.

Since all proposed techniques target high-performance simulations and rendering quality, they are of interest for real-time fluids as well as for fluid animation.

1.4 Challenges

In Lagrangian continuum descriptions, particles move freely with the underlying velocity field. However, this automatic adaptivity comes at the expense of the regularity of particle distributions [Kou05]. This flexibility of computational elements imposes several challenges when dealing with fluid transport problems:

Compact Neighborhoods: Differentials turn out especially simple in SPH (see Sec. 3.2.1) but also are the main source of instabilities when compact smoothing kernels are employed. SPH requires nearly equi-spaced, full particle neighbourhoods, otherwise interpolations result in over- or underestimations of quantities.

Challenge: Tensile Instability

At free surfaces, the reconstructed field is then dominated by the shape of interpolation kernels instead of the interpolated function [Bro85]. As a consequence, incorrect pressure forces are computed, which results in undesirable pressure variations, known as tensile instability. While the differential form of the continuity equation [Mon92] avoids such deficiencies, it usually requires higher-order time stepping schemes, since otherwise numerical errors creep into simulations over time [BT07b, IOS*14]. To maintain stability in combination with summation approaches, a near-density dependent pressure term [CBP05], an artificial repulsion force [Mon00, AO11], a sampling of ghost particles [SB12], or a constant correction [BK02, BTT09] (cf. Sec. 3.1.4) work considerably well. However, these methods do not fully avoid irregular particle distributions at free surfaces.

Challenge: Turbulent Flow

In practice, errors in the approximation of derivative operators limit integration time steps. Often, artificial viscosity [Mon89, HK89, BT07b, SB12] is mandatory to increases stability, but this comes at the cost of damping turbulent flow dynamics. Such numerical dissipation of energy makes modeling of inviscid fluids via Euler equations a challenging task.

Boundary Handling: Even though SPH does not require any explicit interface tracking, efficient modeling of boundary conditions remains complex [KCR08]. Unsurprisingly, much research addresses stable formulations of boundary conditions in fluid-solid [HKK07b, SSP07, BTT09, IAGT10, AIA*12] or fluid-fluid [MSKG05, HA06, BT07b, SP08, IBAT11, SB12] interactions.

However, effects related to the (missing) air phase are in general limited to modeling of surface tension forces [MCG03, CBP05, BT07b, YWTY12, AAT13]. Effects related to air pressure, evaporation and condensation are especially challenging since they require an explicit representation of the air-phase and have not been addressed in the SPH literature so far. Additionally, for physically-based formulations of reactive-diffusive flows, modelling phase singularities at free surfaces becomes mandatory.

Challenge: Liquid-Air Interactions

Very few methods extend the functionality of the surface to compute mass flows [KAG*05, KBKS09, YWTY12]. These available solutions are typically based on explicit representations introducing re-meshing problems when the surface undergoes topological changes. In general, in order to avoid complicated computations on triangulated surfaces and to preserve the adaptive character of particle-based simulations, it is desirable to model such dynamics with an implicit surface representation [BCOS01, AHA10].

However, computational elements do not necessarily coincide with the fluid's surface. Furthermore, thin fluid sheets and splashes emerge quite frequently. The resulting lack of neighboring particles requires a robust handling of neighborhood singularities to ensure stability of the simulation, especially in combination with compact smoothing kernels. Additionally, contributions between neighboring particles must be symmetric in order to avoid unnatural disappearance and creation of transported materials.

Challenge: Robust Surface Model

Adaptive Sampling: Free-surface flows often require high resolutions in order to resolve important fine-scale surface details. When working with uniform particle sizes, the fluid domain is often discretized using unnecessarily large particle numbers. Since computation speed linearly depends on the number of simulated particles, spatially adaptive sampling, that shifts particles to phase-interfaces can thus lead to a significant increase of computation speed and to a reduction of memory requirements.

Global mappings [CKS00, CPK02, Kou05] can seldom be applied, since they require a priori knowledge of the flow and because they introduce numerical diffusion due to particle-to-grid or particle-to-particle interpolations. Even though local operators maintain the adaptive nature of particle methods, minimization of sampling errors [LQB05] is often too time-consuming. Thus, locally-adaptive methods [KAG*06, APKG07, HHK08] suffer from poor sampling of particles, which results in severe particle distortion, leading to approximation errors of derivative operators [BKOF96]. Particularly, for stiff systems such as incompressible fluids, those sampling errors pose a challenge.

Challenge: Sampling Errors

Challenge: Mass Conservation

Coexisting weakly-coupled particle levels [SG11] on the one hand reduce sampling errors that otherwise are introduced by varying kernel radii [BOT01], but on the other hand lead to divergent particle levels, which introduce local conservation problems for unbounded free-surface flows. Additionally, a mass conservative sampling should work well with other performance mechanisms such as adaptive time-steps and compact smoothing kernels.

Combined Surface and Volume Rendering In contrast to grid-based methods, artifact-free reconstruction of quantity fields from an unordered set of data points is challenging. Related to this subject, computer animation focuses mainly on reconstruction of smooth surfaces for offline [ZB05, SSP07, APKG07, BGB11, OCD11, AIAT12, YT13] or interactive [vdLGS09, GSSP10] renderings. However, fluid transport processes must combine surface reconstruction techniques with volume rendering techniques [HLSR08] in order to render the evolution of transported materials inside the fluid's bulk and in order to convey the fluid's geometric shape.

Challenge: Efficient Sampling

To date, very little research [vdLGS09, FAW10, IAAT12] has addressed volume rendering techniques in the context of particle-based fluids. Physically-based volume rendering of particle fields requires sophisticated sampling mechanisms in order to compensate for the high memory throughput.

In general, for larger particle numbers, either a sophisticated preprocessing [FSW09, FAW10] or a fast neighbor search in combination with space-partitioning data structures [ZHWG08, ZGHG10] is required. Even more challenging, interactive applications require an instant rendering. In feature films, instant rendering, which avoids the pre-processing of simulation data, reduces the complexity of the production workflow and enables seamless parameter studies, which provides fast feedback for effect artists.

Challenge: Instant Rendering

1.5 Contributions

This thesis presents three major contributions to the field of SPH-based fluid simulations. Each of the newly-developed techniques, depicted in Fig. 1.4-1.6, can be viewed as one part of a more comprehensive toolbox for simulating transport dynamics. Together, they address some of the major challenges arising by modelling fluid transport processes, which have been reviewed in the previous section:

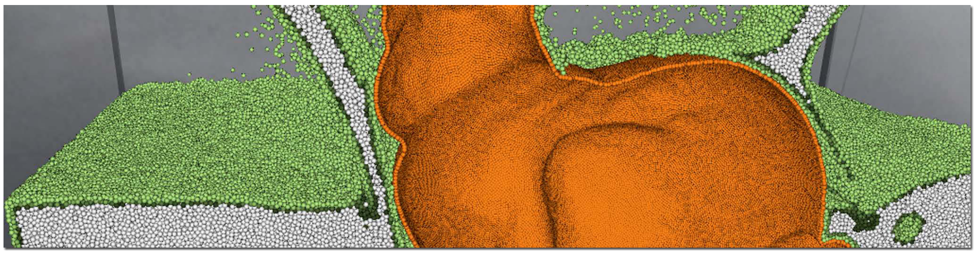


Figure 1.4: Consistent surface model for interfacial flux.

Consistent Surface Model for Interfacial Flux: The stable simulation of reaction-diffusion systems with Neumann boundary conditions relies on a robust surface computation. This thesis proposes a stable surface model that yields a consistent definition of quantity fields for bulk and surface. The key concept of the unified approach is to realize an implicit definition of the fluid's (free) surface by assigning each fluid particle a value estimating its surface area. This area measure is computed by sampling a smeared-out version of the surface delta-function at discrete particle locations in bulk. A correction of area values at highly under-resolved regions or thin fluid sheets enables a stable transition over time. The embedded surface avoids unnecessary back-and-forth interpolation between surface and bulk, enabling conservative transport. Based on this consistent representation, symmetric particle fluxes for surface-surface and surface-bulk interactions are derived, as will be demonstrated for kinetic reactions.

Temporal Blending of Particle Levels: This work introduces a fast and conservative sampling mechanism that allocates computing resources to regions close to the free surface (geometry-driven) and to regions in which concentration gradients arise (dynamics-driven). The idea is to smoothly

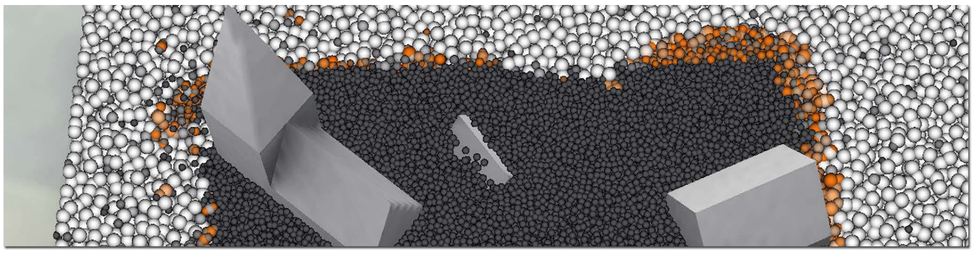


Figure 1.5: Temporal blending of particle levels.

blend in new particles while fading out old ones. The proposed temporal blending yields stable transitions of particle levels for stiff systems. The speed of the adaptive process is therefore driven by an estimation of the introduced sampling error. Predicting sampling errors greatly reduces the noise that occurs in the pressure term when particle configurations change, and it also preserves integration time steps during transition.

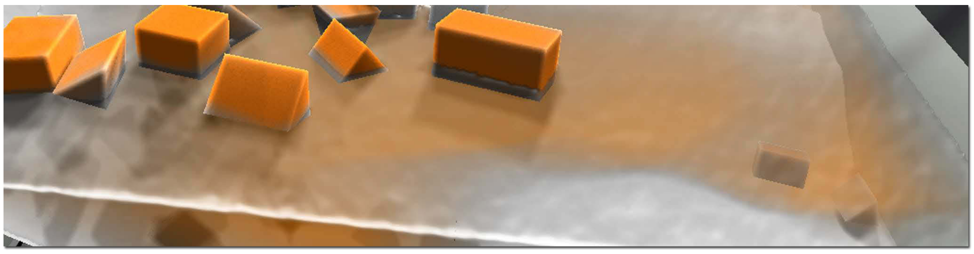


Figure 1.6: Feature preserving ray casting.

Feature-Preserving Ray Casting: This thesis presents an instant and physically-based ray casting for unstructured particle sets. This approach combines state-of-the-art surface renderings with novel volume sampling techniques in order to produce high-quality renderings of the underlying transport dynamics. Two complementary strategies provide the necessary computational efficiency: First, a feature-preserving, adaptive sampling of particle fields reduces the sampling overhead in homogeneous regions by employing an online screen-space error analysis. Secondly, the proposed sparse view-aligned access structure accounts for cache-coherent memory reads of particle data. Large increases in computation speed are achieved without relying on specific hardware features.

1.6 Outlook

Most of the research presented in this thesis has been published as multiple publications during my time at the Computer Graphics and Multimedia Systems Group at the University of Siegen. The main part of this thesis, therefore, is aligned with those published contributions and is structured into six additional chapters:

Chapter 2 introduces the governing equations of fluid transport and relates this work to existing fluid solvers in Computer Graphics.

Chapter 3 begins with our report "SPH Fluids in Computer Graphics" [IOS*14], presented at Eurographics (EG 2014). In contrast, the focus is put on transport-specific aspects. The data-parallel realization of SPH formulations is enabled by our work on "Integrating GPGPU Functionality Into Scene Graphs" [OKK09], which was published in the Proc. of the Vision, Modeling, and Visualization Workshop (VMV 2009).

Chapter 4 describes the "Consistent Surface Model for SPH-based Fluid Transport" [OHB*13] which has been presented in the Symposium on Computer Animation (SCA 2013). The applicability and robustness of the implicit surface model is demonstrated for various effects including adsorption, adhesion, anisotropic diffusion, and cleansing effects.

Chapter 5 outlines our "Temporal Blending for Adaptive SPH" [OK12], published in Computer Graphics Forum (CGF 2012) and invited to Eurographics (EG 2013). This section demonstrates how to smoothly shift computational resources toward regions of interest while ensuring the conservation of transported quantities and integration time steps.

Chapter 6 presents the combined surface and volume rendering technique currently in preparation for publication under the title "Adaptive, Feature Preserving Volume Ray Casting for SPH-based Fluids" [Unpublished]. The underlying cache-coherent sampling mechanisms employed in this work are based on "Topology-Caching for Dynamic Particle Volume Raycasting" [OKK10], which has been published in the Proc. of the Vision, Modeling and Visualization Workshop (VMV 2010).

Chapter 7 concludes this work by summarizing the contributions of this thesis and providing directions for future research.

Chapter 2

Fluid Transport

This section provides the governing equations for fluid transport and gives a brief comparison of related fluid solvers, which embeds the presented work into the much broader field of physically-based fluid animation. For a more general introduction, the interested reader is referred to the textbooks of Shaughnessy et al. [SKS05] and Berthier et al. [BS09] regarding fluid transport problems or to the work of Bridson [Bri08] and Koumoutsakos et al. [KCR08] regarding fluid simulation in Computer Graphics.

In transport problems, the fluid acts as a carrier for specific materials. Even though, fluid transport means transport of any fluid quantity like momentum, in this thesis the focus is on transport of soluted substances. In cases for which no analytical solution can be found, the solution is found by partial differential equations. The continuum is therefore modelled with a finite number of computational elements whose shape is defined by the numerical algorithm under consideration.

Solvers can be classified into Eulerian methods [FM96, Sta99] computing flow quantities in a fixed coordinate system, and Lagrangian approaches [SF95, DC96] employing a moving coordinate system. In Eulerian simulations, computational elements represent stationary points in space while the fluid is flowing past them. In Lagrangian descriptions, quantity fields are described with a finite number of loosely- or strongly-coupled fluid particles.

In Sec. 2.1, we will see how simple the governing transport equations become for Lagrangian continuum descriptions and in Sec. 2.2, we will compare the relative advantages of mesh-based, hybrid, and mesh-free methods.

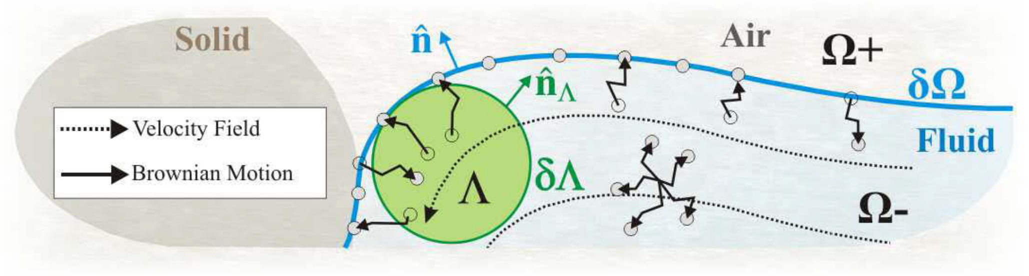


Figure 2.1: Inflow and outflow of molecules (grey) at the fluid boundary $\delta\Omega$ (blue) and through an element's surface $\delta\Lambda$ (green).

2.1 Governing Equations

Convection-Diffusion

Time evolution of quantities occurs simultaneously on two different length-scales: by convection (or macroscopic motion) with the velocity field, and by a microscopic diffusive flux (or Brownian motion) as sketched in Fig. 2.1. Convection can only occur in a moving fluid, while diffusive transport may also occur in a fluid at rest due to spatial variation of a fluid quantity.

Computational Elements

Rather than employing a computationally expensive description based on the actual molecular structure, in continuum descriptions, the fluid domain $\Omega^- = \bigcup_i \Lambda_i$ is divided into a finite number of computational elements. Each element encloses a small fluid region Λ_i, which consists of a finite number of fluid molecules, and thus represents a small volume fraction of size $V_i = \int_{\Lambda_i} dV$. Each element has a concentration Q_i that has dimension $[mol\, m^{-3}]$, and which depends on its fraction $\overline{Q}_i = (V_i Q_i)$ given in $[mol]$ of the total amount of the transported quantity.

Initial Value Problem

Because in continuum mechanics, computational elements are considered to be so small that there is no variation in any fluid quantity on this scale, a solution to fluid transport can be found by solving the following initial value problem:

$$Q_i(t_s) = Q_i(0) + \int_0^{t_s} \frac{D}{Dt} Q_i(t)\, \partial t,$$

which yields the quantity field $Q_i(t) = Q(\mathbf{x}_i, t)$ at time t_s at locations $\mathbf{x}_i$ by starting from initial values at time $t = 0$.

Material Derivative

Here $\frac{D}{Dt}Q_i(t)$ is the so-called material derivative of the quantity field as given in $[mol\, m^{-3} s^{-1}]$. For Eulerian frames, where element positions vary over time, the material derivative $\frac{D}{Dt}Q_i = \frac{d}{dt}Q_i + \mathbf{u}_i \cdot \nabla Q_i$ consists of an unsteady derivative with respect to time and a convective derivative with respect to fluid velocity $\mathbf{u}_i = \frac{d}{dt}\mathbf{x}_i$.

In Lagrangian form, dynamics are formulated with respect to a local reference frame that moves with the fluid for which the convective derivative disappears:

$$Q_i(t_s) = Q_i(0) + \int_0^{t_s} \frac{d}{dt} Q_i(t)\, \partial t, \tag{2.1}$$

For steady fields, the transport problem is fully defined by Eq. (2.1), but for dynamic velocity fields, one also has to couple the reactive-diffusive flux with the fluid motion in order to solve the full transport problem:

$$\mathbf{x}_i(t_s) = \mathbf{x}_i(0) + \int_0^{t_s} \mathbf{u}_i(t)\, \partial t, \tag{2.2}$$

$$\mathbf{u}_i(t_s) = \mathbf{u}_i(0) + \int_0^{t_s} \frac{d}{dt} \mathbf{u}_i(t)\, \partial t. \tag{2.3}$$

According to Reynolds transport theorem, there should be no accumulation of any fluid property. Thus, we will now take a closer look at conservation of quantities (cf. Sec. 2.1.1) and conservation of momentum (cf. Sec. 2.1.2) since they are essential ingredients to physically-based simulations.

2.1.1 Total Transport

Inflow or outflow of quantities occurs only at the element surface $\delta\Lambda$ [1] enclosing the fluid element. Thus, the net flux, given in $[mol\, s^{-1}]$, at which a quantity is crossing depends on the area and outward unit normal $\hat{\mathbf{n}}_\Lambda$ of the element's surface. In the Eulerian viewpoint, the time rate of change of quantities in $[mol\, m^{-2} s^{-1}]$ is then described by the following continuity equation ensuring global conservation:

(Net) Flux

$$\frac{d}{dt} \int_\Lambda Q \; dV = - \int_{\delta\Lambda} Q\, \mathbf{u} \cdot \hat{\mathbf{n}}_\Lambda \; dA + \int_{\delta\Lambda} \sigma \nabla Q \cdot \hat{\mathbf{n}}_\Lambda \; dA + \Theta_{\delta\Omega}, \tag{2.4}$$

where σ is a diffusion coefficient as described by Fick's law which is a scalar or a 3×3 tensor for isotropic or anisotropic diffusion, respectively. This form simplifies if we allow the computational elements to move with the flow field. For fluid particles, the inflow and outflow of quantities at the element's surface as induced by the velocity field is zero. As a result, the corresponding convection term integrating $Q\, \mathbf{u} \cdot \hat{\mathbf{n}}_\Lambda$ over the element's surface is not needed. Thus, for Lagrangian descriptions, the continuity equation reduces to

Eulerian vs. Lagrangian Elements

$$\frac{d}{dt} \int_\Lambda Q \; dV = \int_{\delta\Lambda} \sigma \nabla Q \cdot \hat{\mathbf{n}}_\Lambda \; dA + \Theta_{\delta\Omega}. \tag{2.5}$$

[1] To avoid confusion, $\delta\Lambda$ will be referred to as the element surface, representing the boundary layer between two fluid elements Λ, while $\delta\Omega$ refers to the fluid's surface, i.e. the boundary layer of the fluid Ω^- with other phases Ω^+, such as the fluid-air or the fluid-solid interface.

In both formulations, $\Theta_{\delta\Omega}$ is an additional term representing sources or sinks for fluid quantities. By far the most interesting sources are those that arise at the fluid boundary $\delta\Omega$, which are usually associated with chemical reactions. Inflow and outflow of quantities at phase interfaces is typically described by a Neumann boundary condition $\sigma\nabla Q(\mathbf{x})\cdot\hat{\mathbf{n}}(\mathbf{x}) = \Theta$ for $\mathbf{x} \in \delta\Omega$ as given in $[mol\, m^{-2}s^{-1}]$, where $\hat{\mathbf{n}}$ is the unit normal of the fluid boundary. In continuum mechanics, the phase-singularity is therefore modeled by a surface delta function δ, which has dimensions $[m^{-1}]$ and is defined to be non-zero only at the fluid's surface $\delta\Omega$. The source term can then be described as $\Theta_{\delta\Omega} = \int_\Lambda \Theta\,\delta\;dV$. Applying the Gauss theorem to replace the surface integral over the diffusive flux with a corresponding volume integral then yields

Boundary Flux

Surface Delta Function

$$\frac{d}{dt}\int_\Lambda Q\;dV = \int_\Lambda \nabla\cdot(\sigma\nabla Q)\;dV + \int_\Lambda \Theta\,\delta\;dV, \tag{2.6}$$

By assuming isothermal conditions, bulk diffusion becomes isotropic[2], i.e. $\nabla\cdot(\sigma\nabla Q) = \sigma\nabla^2 Q$. Finally, the differential form of the continuity equation is derived from Eq. (2.6) by taking the limit as the element volume Λ tends to zero and dividing by the differential volume dV:

Isothermal Conditions

Continuity Equation

$$\frac{d}{dt}Q = \underbrace{\sigma\nabla^2 Q}_{\text{diffusion}} + \underbrace{\Theta\,\delta}_{\text{reaction}}\;. \tag{2.7}$$

As a special form, the conservation of constant mass of the carrier fluid is described as $\frac{d}{dt}\rho = 0$, where ρ is the fluid's density. As we will see in Sec. 4, challenges arise in practice from embedding the infinitely thin delta function into the three-dimensional simulation domain.

2.1.2 Velocity Field

Similarly to conservation of quantities, conservation of momentum is described by the well-known Navier-Stokes equations. According to Newton's second law, the rate of change of linear momentum is equal to the surface stress $\mathbf{\Sigma}$ and body forces, like gravity $\mathbf{g}$, acting on the fluid volume. In Lagrangian form conservation of momentum is given as

Conservation of Momentum

$$\frac{d}{dt}\int_\Lambda \rho\mathbf{u}\;dV = \int_{\delta\Lambda} \mathbf{\Sigma}\,\hat{\mathbf{n}}\;dA + \int_\Lambda \rho\,\mathbf{g}\;dV + \int_{\delta\Lambda} \alpha\,\kappa\,\hat{\mathbf{n}} + \nabla_{\hat{\mathbf{n}}}\alpha\;dA$$

[2]Please note that even if diffusion in bulk under isothermal conditions is isotropic, some materials like surfactants show anisotropic behaviour close to the surface (cf. Sec. 4.4).

where α accounts for surface tension at the fluid-air interface. Tension forces [BKZ92] minimize the surface area at the fluid-air interface and act in normal direction $\hat{\mathbf{n}}$ and along the tangential direction as defined by the differential surface operator $\nabla_{\hat{\mathbf{n}}} = \nabla - \hat{\mathbf{n}}(\hat{\mathbf{n}} \cdot \nabla)$. The strength depends on the local curvature κ of the surface and the respective tension coefficient α. Additionally, tangential forces or so-called Marangoni forces may occur due to uneven surface concentration. By converting the surface integrals to volume integrals and then taking the limit as the element volume Λ tends to zero and dividing by the differential volume dV we get

Continuum Surface Tension

$$\frac{d}{dt}\rho\mathbf{u} = \nabla \cdot \mathbf{\Sigma} + \rho\,\mathbf{g} + (\alpha\,\kappa\,\hat{\mathbf{n}} + \nabla_{\hat{\mathbf{n}}}\alpha)\,\delta.$$

Usually, surface stress $\nabla\cdot\mathbf{\Sigma} = -\nabla\cdot\mathbf{I}p + \nabla\cdot\xi$ is divided into shear stress ξ and stress related to pressure p, where $\mathbf{I}$ is a 3×3 identity matrix. Pressure forces counteract compression and therefore act from high to low pressure regions as measured by the negative gradient of pressure $\mathbf{I}p = -\nabla p$. Pressure ensures a divergence free velocity field $\nabla \cdot \mathbf{u} = 0$ for incompressible fluids. For incompressible Newtonian fluids, shear stress then linearly depends on the velocity gradient, i.e. $\nabla \cdot \xi = \nabla\,\mu\,\nabla\mathbf{u} = \mu\nabla^2\mathbf{u}$, where the viscosity constant μ minimizes differences in velocity. Putting all together the well-known Navier-Stokes equations plus an additional surface tension term read

Incompressible Newtonian fluids

Navier-Stokes Equations

$$\frac{d}{dt}\mathbf{u} = \frac{1}{\rho}[\ \underbrace{-\nabla p}_{\text{pressure}} + \underbrace{\mu\,\nabla^2\mathbf{u}}_{\text{viscosity}} + \underbrace{\rho\,\mathbf{g}}_{\text{gravity}} + \underbrace{(\alpha\,\kappa\,\hat{\mathbf{n}} + \nabla_{\hat{\mathbf{n}}}\alpha)\,\delta}_{\text{tension}}\ ]. \tag{2.8}$$

In general, no quantity can be transported across the fluid surface $\delta\Omega$ by means of convection, which corresponds to a no-penetration boundary condition at the solid-fluid interface, i.e. $\mathbf{u}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x}) = 0$ for $\mathbf{x} \in \delta\Omega$.

No-Penetration Condition

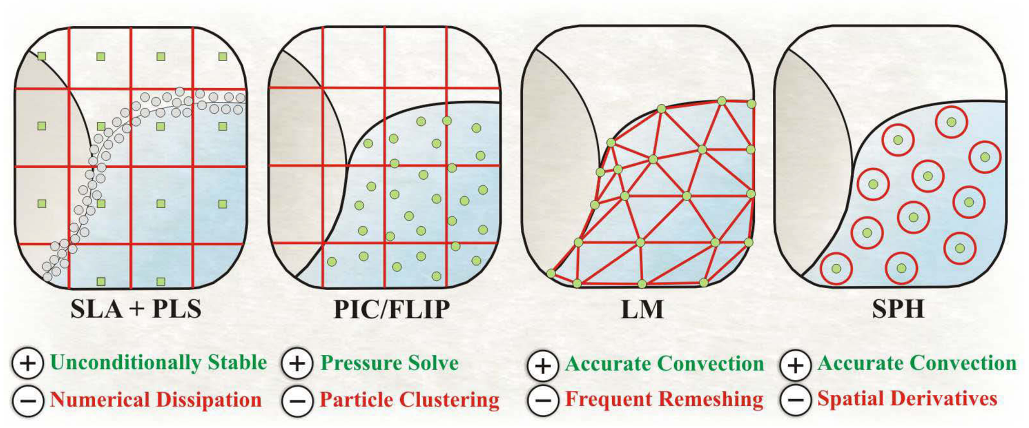


Figure 2.2: Fluid solvers in Computer Graphics mainly differ in their computational elements (red) and data representations (green).

2.2 Fluid Solvers

Even though fluid solvers in Computer Graphics typically utilize the Navier-Stokes equations as described in the last section, their underlying computational elements and data representations can significantly differ. Fig. 2.2 outlines the mainstream methods in Computer Graphics and compares their relative advantages and disadvantages. To further support the high-level perspective, Alg. 1-4 provide pseudo-codes for the basic simulation loops of the non-optimized algorithms using subscripts Q_g and Q_p to differentiate between grid and particle quantities and using $Q_P(\mathbf{x})$, $Q_G(\mathbf{x})$ to indicate interpolation of quantities in a particle field and interpolation of quantities within a grid, respectively. In detail, for convection-dominated transport dynamics, their

Solver Properties

- conservation properties in combination with free-surfaces,
- computational efficiency, depending on the incompressibility solver,
- rendering quality, linked to the smoothness of surface reconstructions,

are the most important characteristics. Note that, a detailed comparison between solvers in the graphics literature is rather difficult because each solver offers a unique solution to the governing equations, and optimized algorithms can even leverage disadvantages of the basic solvers.

Algorithm 1: Combined SLA [Sta99] and PLS [ELF05] simulation step. A tri-linear interpolation (red) of quantity fields $Q_G(\mathbf{x})$ during advection results in numerical diffusion, while surface distances ϕ_g are corrected using PLS.

```
1 update $Q_g \leftarrow Q_g + \Delta t \, \frac{d}{dt} Q_g$
2 advect $Q_g \leftarrow Q_G(\mathbf{x}_g - \Delta t \; \mathbf{u}_g)$ and $\phi_g \leftarrow \phi_G(\mathbf{x}_g - \Delta t \; \mathbf{u}_g)$
3 update $\mathbf{u}_g \leftarrow \mathbf{u}_g + \Delta t \, \frac{d}{dt} \mathbf{u}_g$
4 enforce $\nabla \cdot \mathbf{u}_g = 0$ by solving pressure Poisson equation
5 correct $\phi_g$ using PLS
```

2.2.1 Mesh-based Methods

SLA

The most commonly used grid-based solver in Computer Graphics uses an unconditionally stable Semi-Lagrangian advection (SLA) scheme [Sta99]. In addition to enabling large integration time steps, one of its many strengths is its simplicity of domain discretization as provided by the Marker-And-Cell (MAC) grid [FM96], which stores velocity and pressure values in different locations to increase stability. SLA has a comparably long history and supports a wide range of applications, including interactive simulations [CLT07, CTG10] and transport problems [FOA03, IKC04, KL07, KJI07], providing detailed renderings and smooth surfaces.

Numerical Dissipation

Unfortunately, sampling errors are accumulated over time, leading to smoothing of quantity fields (so-called numerical dissipation) which often results in unnatural volume or mass loss at free surfaces. Much work has reduced this undesirable effect by utilizing higher-order advection schemes [MCPN08, SFK*08], conservative advection schemes [MMTD07, LAF11, CM12], a modified hermite interpolation [FSJ01], or an energy-preserving time integration [MCP*09]. Blurring of sharp features is the primary reason that more Lagrangian components like vortex particles [SRF05], turbulence particles [PTC*10], and marker particle methods [CMVHT02] such as particle level sets (PLS) [OF02, ELF05] have become part of the standard algorithm.

PLS

As summarized in Alg. 1, in an combined SLA and PLS method, massless marker particles sample the zero level set representing the interface and then are passively advected with the velocity-field in order to correct the intermediate level set of the SLA method. PLS has become a common choice to stabilize free-surfaces [EMF02], multi-phase fluids [HK05b, WMT05, LSSF06], and rigid-fluid interactions [CMT04]. However, particles are auxiliary and must be frequently corrected and reinitialized on the underlying fluid rep-

Algorithm 2: In a FLIP step [ZB05], particle quantities are updated by interpolating the change $\Delta Q_G(\mathbf{x}_p)$ back from discrete grid values $\Delta Q_g = Q_g^* - Q_g$. However, noise may develop, leading to irregularly-distributed particles.

```
1 interpolate $Q_g = Q_P(\mathbf{x}_g)$ and $\mathbf{u}_g = \mathbf{u}_P(\mathbf{x}_g)$
2 update $Q_g^* \leftarrow Q_g + \Delta t \, \frac{d}{dt} Q_g$ and $\mathbf{u}_g^* \leftarrow \mathbf{u}_g + \Delta t \, \frac{d}{dt} \mathbf{u}_g$
3 enforce $\nabla \cdot \mathbf{u}_g^* = 0$ by solving pressure Poisson equation
4 update $Q_p \leftarrow Q_p + \Delta t \; \Delta Q_G(\mathbf{x}_p)$ and $\mathbf{u}_p \leftarrow \mathbf{u}_p + \Delta t \; \Delta \mathbf{u}_G(\mathbf{x}_p)$
5 advect $\mathbf{x}_p \leftarrow \mathbf{x}_p + \Delta t \; \mathbf{u}_p$
```

resentation [REN*04, LFO06, CKSW08]. Furthermore, the resolution of the dynamics in PLS-SLA methods remains coupled to the resolution of the MAC grid or tetrahedral grid [FOK05, ETK*07]. An octree data-structure [LGF04], tracking of explicit surface meshes [BGOS06, BBB10, WTGT10, TWGT10], embedded high-resolution grids [EQYF13], adaptive meshes [FOKG05, KFCO06, CGFO06, BXH10] or non-uniform distribution of mesh-lines [ZLC*13] are able to increase resolution locally, but at the cost of frequent remeshing operations.

LBM

Note that in addition to SLA, there exist a variety of alternative mass-conserving grid-based algorithms: For example the Lattice Boltzmann method (LBM) [RT04, KTH*05, TPR*06] can handle complex topologies [TR09] and is easily parallelizeable [KPR*06]. Instead of continuum mechanics, dynamics are described via the Boltzmann equations designed for mesoscopic scales. Volume-of-fluid (VOF) methods [PP04] are usually coupled with level sets [Sus03, AGDJ08, KPyNS10] to maintain a sharply-defined surface.

VOF

2.2.2 Hybrid Methods

PIC/FLIP

Originating from particle-in-cell (PIC) methods [Har64], the fluid implicit particle (FLIP) [BR86, ZB05] method combines advantages of mesh-based and particle methods. FLIP retains the simplicity of the pressure solve from grid solvers and at the same time reduces numerical dissipation by making particles the fundamental data representation for quantity fields. As shown in Alg. 2, particles carry flow quantities that are then sampled on grid elements to increment quantity fields over time. Differences are then interpolated back onto particles to update their quantities accordingly. Applications include solid-fluid coupling [BBB07], viscous free surface flows [BB08], advection of diffuse materials [LTKF08], multi-phase scenarios [BB12], and

Algorithm 3: LM methods [EMB11] treat mesh vertices $\mathbf{x}_p$ as particles. $\frac{d}{dt}Q_p$ and $\frac{d}{dt}\mathbf{u}_p$ are computed by employing linear shape functions describing finite elements which require fixed connectivity between vertices, thus, require a computationally expensive remeshing.

```
1 update $Q_p \leftarrow Q_p + \Delta t \frac{d}{dt}Q_p$ and $\mathbf{u}_p \leftarrow \mathbf{u}_p + \Delta t \frac{d}{dt}\mathbf{u}_p$
2 advect $\mathbf{x}_p \leftarrow \mathbf{x}_p + \Delta t \mathbf{u}_p$ (or semi-Lagrangian)
3 remesh vertices $p \in P$ to vertices $p' \in P$
4 interpolate fields $Q_{p'} = Q_P(\mathbf{x}_{p'})$ and $\mathbf{u}_{p'} = \mathbf{u}_P(\mathbf{x}_{p'})$
```

large bodies [GB13].

The main limitation of FLIP is that particles can clump or generate voids due to the fact that particle-particle interactions are still computed on the grid. In combination with complex surfaces, unwanted noise is introduced. However, due to the loose coupling, particle resolution can be easily adapted [ATT12, ATW13]. Thus, particle distributions are usually improved by reseeding of particles [ATW13] or frequent reinitialization of particle velocities [Bri08] at the cost of increased numerical dissipation. Particle Clustering

Another class of hybrid fluid solvers, the so-called Lagrangian Meshes (LM) [CCM*04, MBE*10, EMB11], have adopted the finite element method usually employed for the simulation of deformable models [NMK*06]. As shown in Alg. 3, vertices of the underlying mesh act as fluid particles and move with the flow field but retain their connectivity. LM methods provide a unified framework for multiphase flows [MEB*12], solids and fluids [CWSO13]. An explicit surface representation is tracked over time. However, topological changes must be explicitly handled by remeshing operations. LM Remeshing

2.2.3 Mesh-free Methods

Lagrangian techniques like SPH [DC96, MCG03] have become increasingly popular due to the fact that convection of quantities is simply defined by the motion of particles with the underlying velocity field and therefore is treated exactly. Mass conservation is trivially ensured. Even though SPH may be combined with grids to solve for incompressibility [RWT11] or to restore regularity [CPK02], a domain grid is generally not needed, except as a bookkeeping device [Gre09]. Computational resources are allocated only to regions of interest and scale with the fluid domain instead of scaling with the (unbounded) simulation domain. Fluid particles ro- SPH

Algorithm 4: SPH methods [SP09b] utilize a neighbor search $N(\mathbf{x}_p)$ to reconstruct quantity fields. Explicit time integration and accumulation of pressure via prediction-correction steps require small integration time-steps due to sensitive field derivatives.

```
1 find neighbours $N(\mathbf{x}_p)$
2 update $Q_p \leftarrow Q_p + \Delta t \frac{d}{dt} Q_p$ and $\mathbf{u}_p \leftarrow \mathbf{u}_p + \Delta t \frac{d}{dt}\mathbf{u}_p$ using $N(\mathbf{x}_p)$
3 enforce $\nabla \cdot \mathbf{u}_p = 0$ using prediction-correction steps
4 advect $\mathbf{x}_p \leftarrow \mathbf{x}_p + \Delta t\ \mathbf{u}_p$
```

bustly handle complicated phase interfaces [MSKG05, SP08, SB12, AAT13] and provide an efficient unification of rigid, soft body, and fluid simulations [SSP07, HKK07b, BIT09, BTT09, IAGT10, AIA*12, ACAT13]. In general, it is not necessary to solve large system matrices, which makes SPH easy parallelizeable [HKK07b, ZSP08, Gre09, GSSP10, IABT11]. Incompressibility conditions are solved either by projecting the velocity field to a divergence free space [CR99, LKO05, SJ06, RWT11], by using a stiff equation of state [Mon94, BT07b], or by iteratively enforcing incompressibility constraints on the velocity field [BLS12], particle positions [MM13], or particle pressures [SP09b], as shown in Alg. 4. However, recent developments in this field [BLS12, MM13, ICS*13] demonstrate that maintaining incompressibility still has room for improvement.

Sensitive Field Derivatives

As already discussed in Sec.1.4, underresolved particle neighborhoods and small support domains can lead to stability issues arising from inaccurate field derivatives [Mon05]. Additionally, efficient neighborhood search mechanisms [Gre09, PH10, GSSP10, IABT11] are mandatory. Furthermore, particle simulations require comparable large particle numbers to resolve surface details. As a result, particle methods have a demand for adaptive sampling mechanisms [KAG*06, APKG07, SG11, CIPT14]. Particle rendering requires special smoothing algorithms [APKG07, SSP07, BGB11, OCD11, YT13] to avoid bumpy surfaces, and requires fast sampling mechanisms for rendering of fluid volumes [FAW10].

VM

In addition to SPH, there have been a number of less frequently used particle methods. For example, Vortex Methods (VM) [AN05, PK05] express velocity in terms of vorticity, and thus naturally avoid compression. However, VMs are rather limited to 2D simulations where vorticity is described by scalar values [RK08, RBCK10], whereas in three dimensions, incompressibility must be solved via non-trivial, vector-valued Poisson equations [KCR08].

Chapter 3

SPH-based Transport

*The following chapter presents the major building blocks of SPH-based fluid solvers, as described in our State-of-the-Art report [IOS*14] presented at Eurographics (EG 2014) in Strasbourg. Compared to standard SPH literature, here the focus is brought on aspects specific to fluid transport. This work combines a dimensionless interpolation (cf. Sec. 3.1.3) with constant corrected kernels (cf. Sec. 3.1.4) on a fully data-parallel system (cf. Sec. 3.3.3). Specialized topics such as surface models, adaptive sampling and rendering are dealt within chapters 4-6. As presented at the workshop Vision, Modelling and Visualization (VMV 2009) in Braunschweig, the integration of GPGPU functionality into scene graphs [OKK09] has rendered possible the seamless integration of our work into a heterogeneous rendering and simulation framework. For a general introduction to particle-based techniques, the interested reader is referred to the SPH literature [Mon05, KCR08].*

Inspired by Monte-Carlo methods, Smoothed Particle Hydrodynamics originally have been developed by Lucy [Luc77], Gingold and Monaghan [GM77] for simulating interstellar gas flows within the field of Astrophysics. Since its introduction to Computer Graphics [MCG03], SPH has been applied to simulation of incompressible fluids [BT07b, SP09b, ICS*13], transport of sediments [KBKS09], thermodynamics [MSKG05] and transport of diffuse materials [IAAT12], only to name a few.

SPH systems in Computer Graphics more or less share the same simulation pipeline as given in Fig.3.1: Reconstruction of quantity fields using Monte-Carlo integration (see Sec. 3.1) allows for straight forward formulation of differentials as required for computation of mass fluxes (see Sec. 3.2). Specialized computation strategies provide SPH with the necessary efficiency (see Sec. 3.3). In the course of this thesis, three new components will be added to efficiently realize transport related effects.

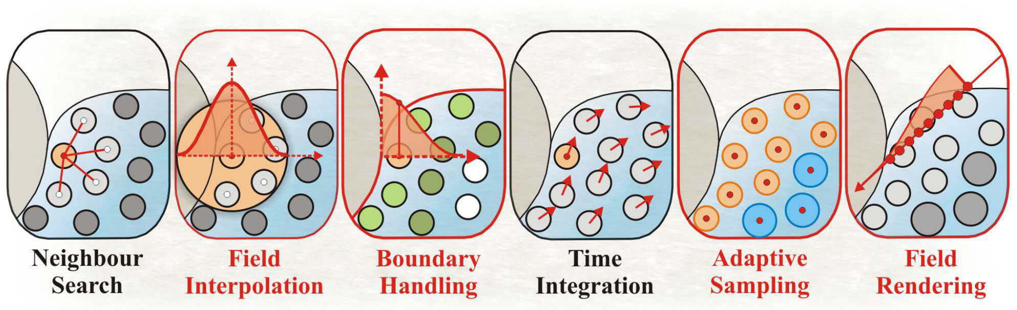


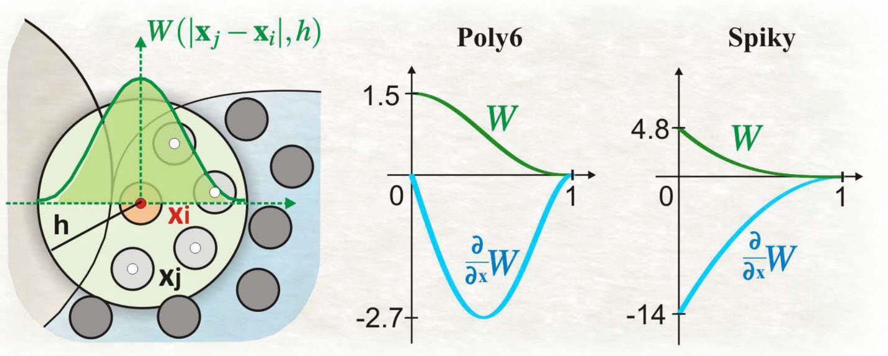
Figure 3.1: Building blocks of SPH-based transport simulations. This thesis contributes to the boundary handling with a consistent surface model (cf. Chap. 4), to the field interpolation and adaptive sampling with an temporal coherent blending (cf. Chap. 5) and proposes an combined surface and volume rendering (cf. Chap. 6).

3.1 Continuum Description

Conceptually, in a Lagrangian reference frame, the carrier fluid is divided into a number of elementary fluid particles moving with the flow. Particles act as interpolation points from which fluid properties can be retrieved. In theory, a continuous function $Q(\mathbf{x})$ is reconstructed by integrating quantities from discrete but unordered set of particles over the fluid domain [KCR08]:

$$Q_I(\mathbf{x}) = \int_\Omega Q(\mathbf{x}')\,\delta(|\mathbf{x}-\mathbf{x}'|)\,d\mathbf{x}', \qquad \delta(x) = \begin{cases} \infty & x = 0 \\ 0 & \text{otherwise,} \end{cases} \tag{3.1}$$

where $\delta(x)$ is a Dirac delta function which is non zero only at sampling position. In practice, quantity fields are reconstructed (cf. Sec. 3.1.1) by replacing the delta function with a piece-wise continuous smoothing function (cf. Sec. 3.1.2). Each particle transports a discrete amount of a quantity Q_i given with respect to its fluid volume V_i (cf. Sec. 3.1.3). In order to increase consistency, smoothing kernels need to be corrected for irregular particle structures or deficient particle neighbourhoods (see Sec. 3.1.4).



$W(|\mathbf{x}_j - \mathbf{x}_i|, h)$
h
Xi
xj
Poly6
1.5
W
0
1
-2.7
$\frac{\partial}{\partial x}W$
Spiky
4.8
W
0
1
-14
$\frac{\partial}{\partial x}W$

Figure 3.2: SPH interpolation scheme and kernel functions.

3.1.1 Field Reconstruction

In SPH, fields are reconstructed by integrating particle quantities over the simulation domain Ω using the following integral interpolant:

$$Q_{I,h}(\mathbf{x}) = \int_\Omega Q(\mathbf{x}')\, W(|\mathbf{x} - \mathbf{x}'|, h)\, d\mathbf{x}' + O(h^2), \tag{3.2}$$

where $W(x, h)$ is a radial symmetric smoothing kernel which is evaluated using the Euclidean distance between sampling position $\mathbf{x}$ and particle positions $\mathbf{x}'$ and which has compact support h and resembles the delta function in Eq.(3.1). Dependent on the choice of the kernel, the integration interpolant gives at least second order accurate results [Mon05]. In practice, Eq. (3.2) is approximated by a Riemann sum over all discrete particle quantities Q_j at positions $\mathbf{x}_j$ in the local neighbourhood (see Fig. 3.2):

Integration Interpolant

Summation Interpolant

$$Q_{I,h}(\mathbf{x}) \approx Q(\mathbf{x}) = \sum_j Q_j(\mathbf{x}) = \sum_j Q_j\, V_j\, W_j(\mathbf{x}), \tag{3.3}$$

where V_j is the discrete volume of a particle and where $W_j(\mathbf{x}) = W(|\mathbf{x} - \mathbf{x}_j|, h)$. Commonly, the short notation $W_{ij} = W(|\mathbf{x}_i - \mathbf{x}_j|, h)$ is used for field reconstruction $Q(\mathbf{x}_i) = Q_i$ at particle locations $\mathbf{x}_i$.

3.1.2 Smoothing Kernels

Compact Kernels

Stability, accuracy and speed of the simulation highly depend on the choice of the smoothing kernel which is of significant importance for the dynamics under consideration. While for accurate simulations or physical interpretation a Gaussian kernel is recommended [Mon05] in practice a cubic spline [Mon92], the poly6-kernel [MCG03] and the Spiky-kernel [DC96] have been regularly employed due to their compactness. The Poly-6 kernel and its derivative in three dimensions, as depicted in Fig. 3.2, read

$$W_{poly6}(x,h) = \frac{315}{64\pi h^9} \begin{cases} (h^2 - x^2)^3 & 0 \le x < h \\ 0 & \text{otherwise,} \end{cases} \tag{3.4}$$

$$\frac{\partial}{\partial x} W_{poly6}(x,h) = -\frac{945}{32\pi h^9} \begin{cases} x\,(h^2 - x^2)^2 & 0 \le x < h \\ 0 & \text{otherwise,} \end{cases} \tag{3.5}$$

in which $x \in \mathbb{R}$ only appears squared which further improves computation speed. The Spiky-kernel and its derivative in three dimensions are

$$W_{spiky}(x,h) = \frac{15}{\pi h^6} \begin{cases} (h - x)^3 & 0 \le x < h \\ 0 & \text{otherwise,} \end{cases} \tag{3.6}$$

$$\frac{\partial}{\partial x} W_{spiky}(x,h) = -\frac{45}{\pi h^6} \begin{cases} (h - x)^2 & 0 \le x < h \\ 0 & \text{otherwise,} \end{cases} \tag{3.7}$$

Kernel Derivatives

which is employed to compute pressure and viscosity forces. By assuming constant support, the outcome of smoothing kernels only depends on the distance between particles and interpolation points which leads to rotational invariant first order spatial and temporal derivatives:

$$\nabla_{\mathbf{x}} W(|\mathbf{x} - \mathbf{x}'|, h) = \frac{\partial}{\partial x} W(|\mathbf{x} - \mathbf{x}'|, h)\, \frac{\mathbf{x} - \mathbf{x}'}{|\mathbf{x} - \mathbf{x}'|} \tag{3.8}$$

$$\frac{d}{dt} W(|\mathbf{x} - \mathbf{x}'|, h) = \frac{\partial}{\partial x} W(|\mathbf{x} - \mathbf{x}'|, h)\, \frac{\mathbf{x} - \mathbf{x}'}{|\mathbf{x} - \mathbf{x}'|} \cdot (\mathbf{u} - \mathbf{u}_j) \tag{3.9}$$

Note that the gradient of the kernel is anti-symmetric, i.e. $\nabla_{\mathbf{x}} W(|\mathbf{x} - \mathbf{x}'|, h) = -\nabla_{\mathbf{x}'} W(|\mathbf{x}' - \mathbf{x}|, h)$. This property is important for spatial derivatives in SPH. For derivatives at locations $\mathbf{x}_j$ we will use the short notation $\nabla_j W_{ij} = \nabla_{\mathbf{x}_j} W(|\mathbf{x}_j - \mathbf{x}_i|, h)$. Even if higher order derivatives $\nabla^{\psi} W(|\mathbf{x}_j - \mathbf{x}_i|, h)$ could be derived similarily, in practice they introduce errors for irregular particle structures and thus are avoided.

Kernel Properties

While smoothing kernel have different characteristics they all have several properties in common. According to Monaghan [Mon92], smoothing kernels

- are even and normalized over their support, i.e. $\int_{-h}^{h} W(x,h)\,dx = 1$,
- satisfy the dirac delta function property, i.e. $\lim_{h\to 0} W(x,h) = \delta$,
- are even functions, i.e. $W(x,h) = W(-x,h)$,
- are compact, i.e. $W(x,h) = 0$ for $x \ge h$, and
- are positive $W(x,h) \ge 0$ for $x < h$.

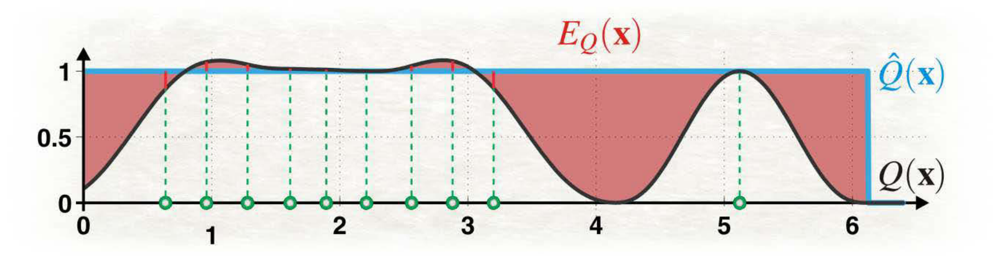


Figure 3.3: Particles (green) sample of a constant field with support radius $h = 1$. Standard SPH (black) yields errors $E_Q(\mathbf{x}) = |1 - Q(\mathbf{x})|$ at boundaries. In contrast, CSPH (blue) yields a consistent reconstruction, i.e. $\hat{Q}(\mathbf{x}) = 1$.

In practice, Eq. (3.3) often cannot reproduce constant functions exactly [Mon92]. If required one has to employ corrective interpolation schemes (See Sec.3.1.4).

3.1.3 Dimensionless Interpolation

In the original work of Ginghold and Monaghan [GM77], discrete particle volumes are defined as $V_i = \frac{m_i}{\rho_i}$.While a particle's mass $m_i = v_i \rho_0$ is an intensive particle property defined per rest volume v_i, particle density $\rho_i = \sum_j m_j W_{ij}$ is an extensive property which is dynamically computed in order to maintain interpolation properties under changing particle configurations. Since a particle mass is constant, V_i is directly related to the rest density of the fluid, e.g. $\rho_0 \approx 1000 \left[\frac{kg}{m^3}\right]$ for water. Particle volumes become more stable by removing this dependency via dimensionless numbers η_i called particle number densities: **Number Density**

$$V_i = \frac{m_i}{\rho_i} = \frac{v_i \rho_0}{\rho_0 \sum_j v_j W_{ij}} = \frac{v_i}{\sum_j v_j W_{ij}} = \frac{v_i}{\eta_i}, \tag{3.10}$$

Note, if $v_i = const$, Eq.(3.10) yields the formulation in the literature [HA06, SP08, AIA*12]. Usually, a particle volume ensures $V_i^{-1} = \sum_j W_{ij}$ for regular and filled particle neighbourhoods.

However, interpolation properties are not satisfied at fluid boundaries as shown in Fig. 3.3: At the fluid-solid interface, neighbouring rigid particles contribute to the density summation. Since homogeneity of the rigid sampling is not guaranteed, rest volumes of rigid particles are pre-computed as $v_b = \frac{1}{\sum_k W_{kb}}$ and recomputed whenever solid objects which are in contact change their relative positions [AIA*12]. While $\eta_i \approx 1$ for the fluid-solid **Rigid Particles** interface, close to the free surface the density is incorrectly approximated due to an insufficient number of particle neighbours (cf. Fig. 3.3). To ease the effect this work avoids negative pressures by ensuring $\eta_i \geq 1$, while for other quantities a constant correction is employed.

Neighbourhood Deficiency

3.1.4 Constant Corrected Interpolation

Due to particle disorder and insufficient number of neighbouring particles [BK02], standard SPH interpolation (cf. Eq. (3.3)) does not satisfy interpolation properties, i.e. $Q(\mathbf{x}_i) \neq Q_i$, as visualized in Fig. 3.3. According to Hieber et al. [HK05a] the dimensionless measure $\sum_k V_k W_k(\mathbf{x})$ is equal to one for regular particle structures, but naturally drops close to fluid boundaries. Taking this measure for the regularity of a particle's neighbourhood as a normalization term enables for a zero order consistent interpolation. According to Bonet et al. [BK02] this can be expressed through a corrected SPH kernel, also known as a Shepard Interpolation [She68]:

Regularity Measure

Corrected SPH

$$\hat{W}_j = \frac{W_j(\mathbf{x})}{V(\mathbf{x})} = \frac{W_j(\mathbf{x})}{\sum_k V_k W_k(\mathbf{x})}, \tag{3.11}$$

The corresponding constant corrected SPH interpolation (CSPH) will be denoted as $\hat{Q}(\mathbf{x}) = \sum_j Q_j V_j \hat{W}_j(\mathbf{x})$.

CSPH Applicability

However, gradients of Eq. (3.11) require computation of inverses which limits their use. In this thesis, CSPH is employed to stabilize computation of surface area (cf. Sec.-4), to interpolate quantities during blending of particle levels (cf. Sec.-5) and to sample particle fields during volume ray casting (cf. Sec.-6). Higher order interpolation methods have not been applied as they increase the computation complexity significantly and C^0 consistency already gave sufficient approximations.

3.2 Transport Formalism

According to Newton's third law, quantity or momentum exchange between neighbouring particles must be opposite but equal in amount in order to ensure conservation. In the following, we derive symmetric pair-wise particle contributions for mass fluxes, i.e. $\Gamma_{i\leftarrow j} = -\Gamma_{j\leftarrow i}$, as well as inter-particle forces, i.e. $\mathbf{F}_{i\leftarrow j} = -\mathbf{F}_{j\leftarrow i}$.

3.2.1 Field Differentials

The beauty of SPH is that field derivatives are easily computed. In continuous form, the spatial derivative of a field is obtained by interpolating the gradient of a quantity field, which yields

$$\nabla_{\mathbf{x}} Q_{I,h}(\mathbf{x}) = \int_\Omega [\nabla_{\mathbf{x}'} Q(\mathbf{x}')]\, W(|\mathbf{x}-\mathbf{x}'|, h)\, d\mathbf{x}'.$$

By applying the product rule one gets

$$\nabla_{\mathbf{x}} Q_{I,h}(\mathbf{x}) = \int_\Omega \nabla_{\mathbf{x}'} \left[Q(\mathbf{x}')\, W(|\mathbf{x}'-\mathbf{x}|, h)\right] d\mathbf{x}' \;-\; \int_\Omega Q(\mathbf{x}')\, \nabla_{\mathbf{x}'} W(|\mathbf{x}'-\mathbf{x}|, h)]\, d\mathbf{x}'.$$

On can then use the gauss theorem to convert the first integral into corresponding surface integral. Since the kernel has compact support, the kernel is zero on the surface, which also zeros out the first integral and renders the gradient of a quantity field as [BKDG98]

Basic Gradient Approximation

$$\nabla_{\mathbf{x}} Q_{I,h}(\mathbf{x}) = \int_\Omega Q(\mathbf{x}')\, \nabla_{\mathbf{x}'} W(|\mathbf{x}-\mathbf{x}'|, h)\, d\mathbf{x}',$$

where the property $\nabla_{\mathbf{x}'} W(|\mathbf{x}-\mathbf{x}'|, h) = -\nabla_{\mathbf{x}'} W(|\mathbf{x}'-\mathbf{x}|, h)$ has been used. The corresponding SPH summation formula approximating the integral interpolant reads

$$\nabla_{\mathbf{x}} Q(\mathbf{x}) = \sum_j Q_j\, V_j\, \nabla_j W_j(\mathbf{x}). \tag{3.12}$$

Similarily, the laplacian, divergence, and time derivative of a quantity field are derived as

$$\nabla^2_{\mathbf{x}} Q(\mathbf{x}) = \sum_j Q_j\, V_j\, \nabla^2_j W_j(\mathbf{x}) \tag{3.13}$$

$$\nabla_{\mathbf{x}} \cdot \mathbf{Q}(\mathbf{x}) = \sum_j \mathbf{Q}_j\, V_j \cdot \nabla_j W_j(\mathbf{x}) \tag{3.14}$$

$$\nabla_{\mathbf{x}} \times \mathbf{Q}(\mathbf{x}) = \sum_j \mathbf{Q}_j\, V_j \times \nabla_j W_j(\mathbf{x}) \tag{3.15}$$

$$\frac{d}{dt} Q(\mathbf{x}) = \sum_j Q_j\, V_j\, (\mathbf{u}(\mathbf{x}) - \mathbf{u}_j) \cdot \nabla_j W_j(\mathbf{x}). \tag{3.16}$$

A big advantage of SPH is that gradients affect smoothing kernels only. However, spatial derivatives do not vanish if Q is constant [Mon05]. Furthermore, they will break-down symmetry laws for pair-wise particle contributions. Various alternatives such as an arithmetic mean [MCG03] have been investigated and currently the following gradient approximations are preferred:

Symmetric Gradient Approximations

For example, substituting Eq. (3.12) for derivatives in $\frac{\nabla Q}{\theta} = \nabla\left(\frac{Q}{\theta}\right) + \frac{Q\nabla\theta}{\theta^2}$ yields the following symmetric formulation [Mon92, CEL06]:

$$\nabla_{\mathbf{x}} Q(\mathbf{x}) = \theta(\mathbf{x}) \sum_j \left[\frac{Q_j}{\theta_j} + \frac{\theta_j\, Q(\mathbf{x})}{\theta(\mathbf{x})^2}\right] V_j \, \nabla_j W_j(\mathbf{x}), \tag{3.17}$$

where θ is any differentiable function and in Sec. 3.2.3 will be replaced by the number density in order to derive symmetric pressure forces.

Since second derivatives are quiet sensitive to particle disorder [Mon05, BT07b], they are better approximated by using an integral approximation as derived by expanding $Q(\mathbf{x}_j)$ in a first order Taylor series about $\mathbf{x}$:

$$Q(\mathbf{x}_j) - Q(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_j) \cdot \nabla_{\mathbf{x}} Q(\mathbf{x}) + \frac{1}{2}|\mathbf{x} - \mathbf{x}_j|^2 \, \nabla^2_{\mathbf{x}} Q(\mathbf{x}) + O(h^2)$$

By multiplying both sides with $\frac{(\mathbf{x}-\mathbf{x}_j)^T}{|\mathbf{x}-\mathbf{x}_j|^2} \nabla_j W_j(\mathbf{x})$, integrating both sides over the simulation domain $\int_\Omega d\mathbf{x}_j$ one gets [Bro85]

$$\frac{1}{2}\nabla^2_{\mathbf{x}} Q(\mathbf{x}) = \int_\Omega (Q(\mathbf{x}_j) - Q(\mathbf{x}))\, \frac{\mathbf{x} - \mathbf{x}_j}{|\mathbf{x} - \mathbf{x}_j|^2} \cdot \nabla_j W_j(\mathbf{x})\, d\mathbf{x}_j,$$

by using $\int_\Omega \nabla_j W_j(\mathbf{x})\, d\mathbf{x}_j = 0$ and $\int_\Omega (\mathbf{x} - \mathbf{x}_j)^T \, \nabla_j W_j(\mathbf{x})\, d\mathbf{x}_j = 1$. The SPH summation then results in a stable second order approximation of the Laplacian by using only first order derivatives [Bro85, CM99]:

Integral Approximation

$$\nabla^2_{\mathbf{x}} Q(\mathbf{x}) = 2 \sum_j (Q_j - Q(\mathbf{x}))\, V_j \, \frac{\mathbf{x} - \mathbf{x}_j}{|\mathbf{x} - \mathbf{x}_j|^2} \cdot \nabla_j W_j(\mathbf{x}). \tag{3.18}$$

Since $\frac{\mathbf{x}-\mathbf{x}_j}{|\mathbf{x}-\mathbf{x}_j|^2} \cdot \nabla_j W_j(\mathbf{x}) = \frac{1}{|\mathbf{x}-\mathbf{x}_j|}|\nabla_j W_j(\mathbf{x})|$ the result is radial symmetric. An integral approximation is used to derive artificial viscosity (cf. Sec. 3.2.3) and to compute a stable second order accurate diffusive flux (cf. Sec. 3.2.2).

3.2.2 Diffusive Flux

In SPH, a diffusive quantity flux is defined via net fluxes $\Gamma_{i\leftarrow j}$ from neighbouring particles:

$$\frac{d}{dt} Q_i = \frac{1}{V_i} \sum_j \Gamma_{i\leftarrow j}. \tag{3.19}$$

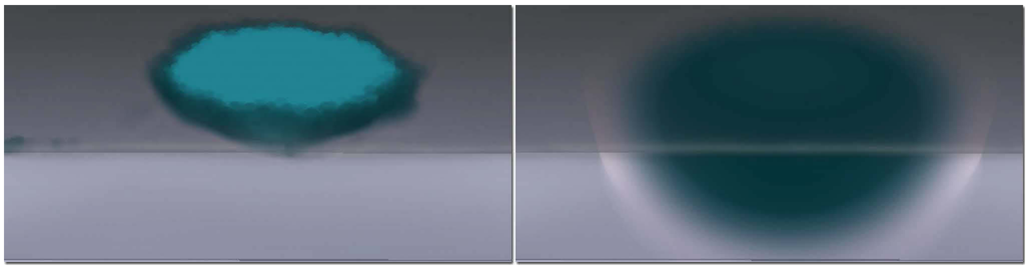


Figure 3.4: Drop of dye. Propagation speed strongly depends on the choice of the smoothing radius, which has been doubled from left to right.

Unfortunately, a direct formulation using Eq. (3.13) is not conservative, sensitive to particle disorder and gets negative for close particles, which unnaturally would invert quantity fluxes. An integral approximation Eq. (3.18) on the other hand yields a symmetric flux between neighbouring particles:

Symmetric Flux

$$\Gamma_{i\leftarrow j} = 2\sigma\, V_i V_j\, (Q_j - Q_i)\, \frac{1}{|\mathbf{x}_i - \mathbf{x}_j| + 0.01h^2}\, |\nabla_j W_{ij}|, \tag{3.20}$$

where $0.01h^2$ is added to avoid division by zero. However, aside from the choice of the diffusion constant σ, the propagation speed also highly depends on the support radius of smoothing kernels, as shown in Fig. 3.4. Even though, small support radii are common for convection dominated transport [SAC*99, MSKG05, KBKS09], they reduce the effectiveness of diffusion constants in diffusion dominated flows.

Kernel-Dependent Propagation

3.2.3 Force Symmetry

Convection of fluid quantities is defined by the following sum of body as well as surface forces acting on fluid particles (cf. Eq. (2.8)):

Inter-Particle Forces

$$\begin{aligned} m_i \frac{d}{dt}\mathbf{u}_i &= \mathbf{F}_i^{\mathbf{g}} + \mathbf{F}_i^{p} + \mathbf{F}_i^{\mu} + \mathbf{F}_i^{\alpha} \\ &= \mathbf{F}_i^{\mathbf{g}} + \sum_j [\mathbf{F}_{i\leftarrow j}^{p} + \mathbf{F}_{i\leftarrow j}^{\mu} + \mathbf{F}_{i\leftarrow j}^{\alpha}], \end{aligned} \tag{3.21}$$

where $\mathbf{F}_i^{\mathbf{g}} = m_i \mathbf{g}_i$ is an external body forces like gravity, and $\mathbf{F}_i^{p}, \mathbf{F}_i^{\mu}, \mathbf{F}_i^{\alpha}$ are surface forces for pressure, viscosity and surface tension respectively.

Symmetric pressure forces as defined by $\mathbf{F}_i^{p} = V_i \nabla p_i = v_i(\frac{1}{\eta_i}\nabla p_i)$ are derived in SPH by substituting $\theta = \eta$ in Eq. (3.17):

Pressure Force

$$\mathbf{F}_{i\leftarrow j}^{p} = -v_i v_j \left(\frac{p_i}{\eta_i^2} + \frac{p_j}{\eta_j^2}\right) \nabla_j W_{ij}, \tag{3.22}$$

Artificial Viscosity

where p_i are particle pressures as explained in Sec. 3.3.1. Similarly, viscosity forces $\mathbf{F}^{\mu} = V_i(\mu \nabla^2 \mathbf{u})$ are derived by using an integral approximation [Mon05] as given in Eq. (3.18):

$$\mathbf{F}^{\mu}_{i\leftarrow j} = -2\mu\, V_i V_j \,\frac{(\mathbf{u}_i - \mathbf{u}_j)\cdot(\mathbf{x}_i - \mathbf{x}_j)}{|\mathbf{x}_i - \mathbf{x}_j|^2 + 0.01h^2}\, \nabla_j W_{ij}, \tag{3.23}$$

Surface Tension

where $0.01\,h^2$ is added to avoid singularities. Even if some fluids, like water usually are not viscous, the stability of the simulation profits from such an artificial viscosity term [Mon05]. To reduce sensitivity to irregular particle structures, the continuum surface tension (CSF) (cf. Eq. (2.8)) is often replaced by using inter-particle interactions forces (IIF) [TM05, BT07b], which read

$$\mathbf{F}^{\alpha}_{i\leftarrow j} = -\alpha \frac{v_j}{v_i}\,\frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|} W_{ij}, \tag{3.24}$$

where α defines the strength of attraction. Alternatively, by modelling color fields, the CSF model [MCG03, KAG*05, HA06] or combined microscopic and macroscopic tension forces [AAT13] can be employed.

3.2.4 Boundary Handling

Rigid-Fluid Forces

Smooth particle distributions at rigid-fluid interfaces are usually achieved by consistently sampling solids with particles (cf. Sec. 3.1.3). Pressure contributions from adjacent rigid particles k to fluid particles i ensure the necessary no-penetration condition, which are modelled by reflecting fluid properties onto the rigid phase, i.e. by setting $p_i = p_k$ and $\eta_i = \eta_k$ [AIA*12]:

$$\mathbf{F}^{p}_{i\leftarrow k} = -v_i v_k\, 2\frac{p_i}{\eta_i^2} \nabla_k W_{ik},$$

Similarly, slip conditions are modelled by altering viscosity at the contact line:

$$\mathbf{F}^{\mu}_{i\leftarrow k} = -2\mu_r\, V_i V_k \,\frac{(\mathbf{u}_i - \mathbf{u}_k)\cdot(\mathbf{x}_i - \mathbf{x}_k)}{|\mathbf{x}_i - \mathbf{x}_k|^2 + 0.01h^2}\, \nabla_k W_{ik},$$

where coefficients μ_r control the damping effect. All in all this yields symmetric contributions between rigid and fluid particles, e.g. by setting $\mathbf{F}^{p}_{k\leftarrow i} = -\mathbf{F}^{p}_{i\leftarrow k}$ and $\mathbf{F}^{\mu}_{k\leftarrow i} = -\mathbf{F}^{\mu}_{i\leftarrow k}$.

Boundary Flux

The interested reader is referred to the literature for details regarding the two-way coupling [AIA*12] or rigid body dynamics [Bar97]. Aside from convection, a robust model of phase singularities is required in order to model Neumann boundary conditions for the diffusive flux.

3.3 Computation Strategies

In this section we are going to investigate how fluid transport problems are efficiently solved in current state-of-the art algorithms. This mainly includes the following three tasks: maintaining incompressibility conditions to avoid volume loss, dynamically adjusting integration time steps to reduce simulation times, and building efficient particle access structures to support fast neighbour lookups in unbounded simulation domains.

3.3.1 Enforcing Incompressibility

Splitting Concept

Maintaining low compressibility is essential for the perception of fluids but is also one of the most computationally expensive parts. The most prominent solutions employ the splitting concept [Bri08], where an intermediate velocity $\mathbf{u}_i^*$ is used to split the velocity update $\frac{d}{dt}\mathbf{u}_i = \frac{1}{\Delta t}(\mathbf{u}_i(t+\Delta t) - \mathbf{u}_i^* + \mathbf{u}_i^* - \mathbf{u}_i(t)) + O(\Delta t^2)$ into two subsequent steps:

$$\begin{aligned} \mathbf{u}_i^* &= \mathbf{u}_i(t) + \frac{\Delta t}{m_i}(\mathbf{F}_i^{\mathbf{g}} + \mathbf{F}_i^{\mu} + \mathbf{F}_i^{\alpha}) \qquad (3.25) \\ \mathbf{u}_i(t+\Delta t) &= \mathbf{u}_i^* + \frac{\Delta t}{m_i}\mathbf{F}_i^p \end{aligned}$$

which effectively separates computation of all non-pressure forces and from computation of pressure forces. That way both can be solved using separate numerical methods.

State Equation

Often, particle pressures are computed from density values using a state equation [DC96, MCG03, BT07b], such as $p_i = \beta\,(\eta_i - 1)$ which relates density to the rest density by using a stiffness constant β. However, density is underestimated when using small values for β. To maintain incompressibility, large values are required which result in stiff systems that dominate the simulation time step.

PCISPH

In Predictive Corrective Incompressible SPH (PCISPH) [SP09b], instead, particle pressures are iteratively refined to avoid stiff state equations. Alg. 6 depicts the corresponding algorithm in combination with a particle blending as described later in Sec. 5. Pressure is recomputed in a Jacobi-style fashion until the density of all particles converges to the rest density, i.e. $\eta^*_{err} = \eta^*_{max} - 1 < \epsilon$ where $\eta^*_{max} = \arg\max_i\ \eta^*_i$. Recently, a number of alternative solvers [MM13, ICS*13] have been developed which further reduce the overall simulation time. The reader is referred to the report [IOS*14] for a detailed overview of incompressibility solvers.

3.3.2 Time Integration

A critical step for each fluid solver is to integrate the governing equations, Eq. (2.1) and Eq. (2.2), over time. In contrast to implicit integrators which acquire a solution to a system of equations, explicit integrators directly calculate the state of the system using finite difference approximations. In the context of SPH, explicit symplectic integrators [LS94], such as the first order Euler-Chromer integration [IAGT10, AIAT12], are commonly used to obtain a numerical solution.

Euler-Chromer

To maintain robustness comparable to implicit integrators, integration time steps are dynamically adapted [IAGT10] employing Courandt Friedrich Lewy (CFL) conditions to ensure convergence. Since the speed of numerical propagation must be higher than the speed of physical propagation, regarding convection, the time step must satisfy [Mon92]

CFL Conditions

$$\Delta t \leq \lambda_F \sqrt{\frac{h}{F_{\max}}} \quad \text{and} \quad \Delta t \leq \lambda_u \frac{h}{u_{\max}},$$

where $F_{\max}, u_{\max}$ are the maximum magnitude of force and of velocity for all particles and $\lambda_u = 0.4, \lambda_F = 0.25$ according to Monaghan [Mon92]. CFL conditions for the diffusive flux are derived similarly [CM99], but in general are not the limiting factor for convection dominated flows.

3.3.3 Particle Neighbourhoods

Lagrangian simulations in general do not have to solve global linear systems of equations but instead pose the challenge of changing particle neighbourhoods. Fortunately, smoothing kernels have only local influence. Thus, computation speed is drastically improved by particle access structures. Fig. 3.5 compares existing data-parallel neighbourhood computations.

Particle Access Structures While kd-trees are commonly employed in combination with non-uniformly support radii [SBH09, KAG*06, APKG07], regular lattices of cubic cells of size h are common for uniform particle sizes [THM*03, Gre09, ZGHG10, GSSP10, IABT11]. In general, a space-filling z-curve, which enforces spacial compactness, defines the mapping $C(\mathbf{x}) : \mathbb{R}^3 \to \mathbb{N}_0$ from particle positions to cell keys. Conceptually, z-indices can be seen as the node path through an octree, from the root node at depth $l = 0$ to leaf nodes or cells at depth $l = D$. The (shuffled) keys of such a path are efficiently computed by compositing the 3-bit child codes

Z-Indexing

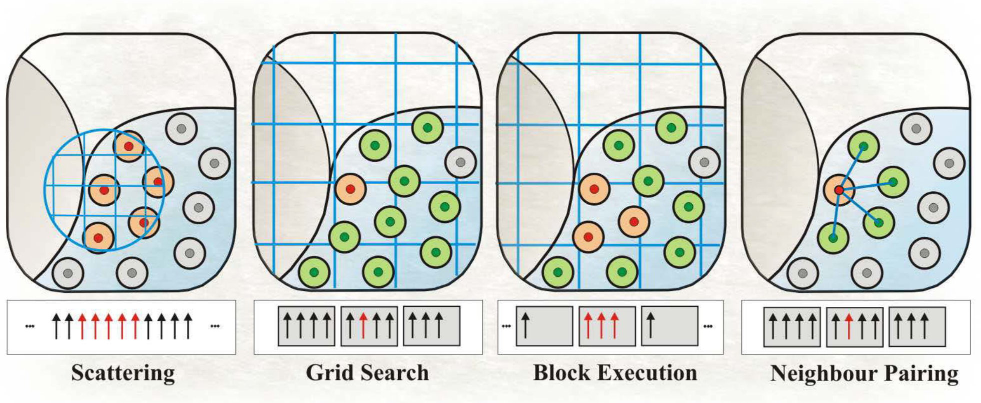


Figure 3.5: Comparison of neighbourhood computations regarding the particle-to-thread assignment (bottom), the access structure (blue), and the particle read (green) and write (orange) operations. Particle states are updated either by scattering contributions onto neighbouring particles, by gathering of particles in neighbouring cells, by block execution of all particles in a cell, and by explicitly holding references to neighbouring particles.

$(C_0, C_1, C_2) \in \{0,1\}^3$ of all nodes visited during path traversal [ZGHG10]:

$$C(\mathbf{x}) = \sum_{l=1}^{D} (\, C_2^l\, 2^2 \;+\; C_1^l\, 2^1 \;+\; C_0^l \,)\, 2^{3(D-l)}, \quad \text{where} \quad C_k^l = \begin{cases} 1 & x_k < c_k^l \\ 0 & \text{otherwise}, \end{cases} \tag{3.26}$$

where $(c_0^l, c_1^l, c_2^l)^T \in \mathbb{R}^3$ is the centroid of the cell that contains $\mathbf{x} = (x_0, x_1, x_2)^T$ at depth $l-1$. During simulation, potentially contributing particles are then found by means of a lookup in the $3 \times 3 \times 3 = 27$ neighbouring cells.

However, uniform grids also reserve memory for empty cells, which unnecessarily consumes available resources. Compact hashing [IABT11], a binary search [GSSP10] or sparse hierarchical data structures [ZGHG10] overcome these size restrictions. On the one hand, hierarchical access structures increase construction cost from $O(N)$ to $O(N \log N)$ and increase the cost to access neighbouring particles from $O(1)$ to $O(\log N)$, but on the other hand they easily build upon z-indexing and support non-uniform particle sizes as used in Sec. 5. **Hierachical Structures**

Access speed can be reintroduced by pre-computing neighbourhood sets [PH10]. Each particle at position $\mathbf{x}_i$ explicitly holds references to all particles j within its support h, resulting in neighbour lists $N_i := \{j \mid |\mathbf{x}_i - \mathbf{x}_j| <$ **Neighbour Pairing**

$h\}$. Once neighbour lists have been created, the access grid is not required anymore. On the one hand, since each particle has ~ 30 neighbours on average such neighbour references increase memory consumptions, but on the other hand, SPH operations compute much faster since memory reads for non-contributing particles are avoided.

SIMD Architectures

Data-Parallelism Since their are almost no data dependencies, SPH methods in general map well to single-instruction, multiple data (SIMD) architectures [KmWH10], like Multicore CPUs or Graphics Processing Units (GPUs). GPUs increase simulation speed due to their high memory bandwidth and due to their large number of compute cores, so-called streaming multi-processors (SM), each of which executes a block of threads in parallel [NVI11]. On SIMD architectures, each thread updates the state of a single particle.

Scattering vs. Gathering

In general, neighbourhood computations can be carried out via two dual mapping operations [SDG08]: either by scattering particle contributions onto points of evaluation [KLRS04, KSW04, AIY*04, KC05, HCM06, HKK07b, HKK07a, ZSP08], at the cost of write collisions, or by gathering of particle data [MCG03, HKK07b, Gre09, IABT11, MM13], at the cost of neighbourhood queries. While scattering is efficiently realized on the rasterization pipeline by blending particle contributions into several texture slices using render-to-texture mechanisms [KC05], the efficiency of gathering approaches mainly is determined by two parameters: thread coherence and thread occupancy, i.e. the number of parallel running threads.

Z-index Sorting

Spatial sorting of particle data with respect to cell keys (Eq. (3.26)) via a data-parallel radixsort [SHG09] drastically increases coalesced data access [Gre09, GSSP10, IABT11]. Additionally, neighbour lists [PH10] reduce branch divergences and the register usage at the same time. Alternatively, execution of thread blocks can be aligned with cells of the access structure [GSSP10] at the cost of introducing architectural dependencies. That way particle data is copied from global memory to a block's local memory once for all particles in a thread block. Memory throughput can further be improved by using a structure of attribute arrays, instead of designing a single array of particle structures [How07]. Last but not least, lazy memory synchronization avoids unnecessary copy operations between main memory and global memory [OKK09].

Block Execution

Structure of Arrays

3.4 The New Components

This chapter has shown that SPH provides a simple solution to the governing equations of bulk transport. However, regarding surface related transport phenomena some questions remain open: Firstly, there is yet no formalism for the Neumann boundary condition for the diffusive flux. Sec. 4 defines an consistent surface model which supports robust formulation of dynamics on surfaces in combination with small support radii. Secondly, since the computation speed linearly depends on the particle count, SPH-based transport profits from an adaptive sampling in order to increase the resolution close to the fluid's surface. A conservative sampling is given in Sec. 5. Thirdly, even though most SPH literature focus on surface rendering, transport simulations additionally require fast rendering mechanisms of concentration fields in bulk. A combined surface and volume rendering for SPH-based transport simulations is presented in Sec. 6.

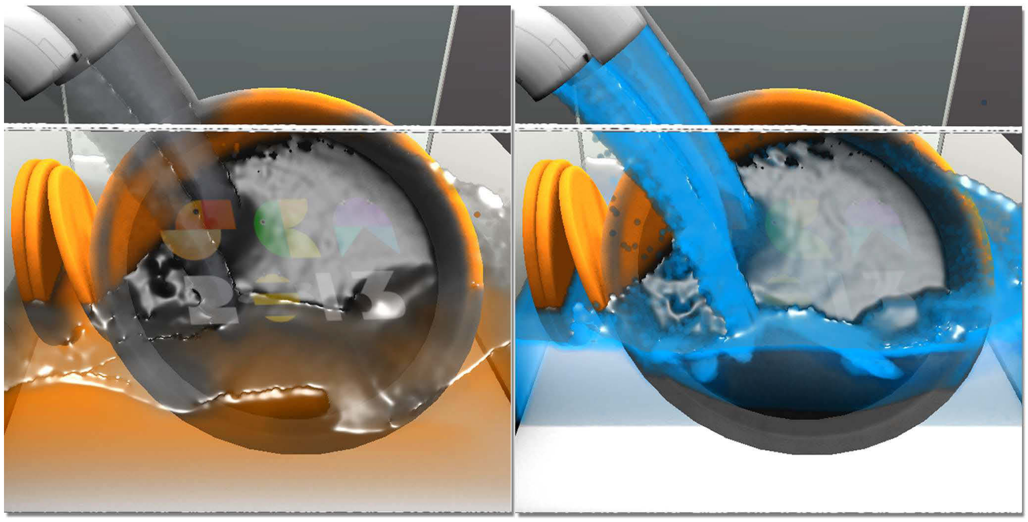


Figure 4.1: A pan's surface is cleansed from grease (orange) due to detergent concentration (blue) at the fluid's surface (visualized on the right).

Chapter 4
Consistent Surface Model

*This chapter presents an embedded surface model that is essential for robust formulations of quantity fluxes at fluid boundaries, as described in Chap. 2.1.1. Thus, SPH-based bulk transport, as presented in the previous chapter, is coupled with an efficient interfacial flux in order to stably model kinetic reactions. The sampling mechanisms presented in the following chapters profit from an stable detection of phase singularities and have been published in [OHB*13] at the Symposium of Computer Animation (SCA 2013) in Los Angeles.*

Surface effects play an essential role in fluid simulations. A vast number of dynamics including wetting of surfaces, cleansing, and foam dynamics are based on surface-surface and surface-bulk interactions, which in turn rely on a robust surface computation. Numerous SPH publications deal with fluid surfaces and interfaces and surface-related effects, inter alia to realize erosion of terrains [KBKS09], density contrasts [SP08], diffuse materials [LTKF08], porous flow [LAD08], and dynamics of foam [IAAT12], temperature modulated viscosity [SAC*99], melting and solidification [SSP07], changing solid conditions of ice [IUYTN10], simulations of trapped air [MSKG05], bubbles [HLYK08, IBAT11], diffuse materials [LTKF08, IAAT12], and wetting effects for granular materials [RSKN08, LD09], only to name a few. However, the simulation of fluid transport processes on free surfaces has not been addressed yet.

Surface Modelling

The reconstruction of surfaces and interfaces is the core component of any physically-based, surface-related effect simulation. In Lagrangian simulations, the options to choose when modeling surfaces are either mesh-based [YWTY12, CWSO13, WTGT10] or integrated particle-based approaches like color-fields [ZB05, SSP07, APKG07, AHA10, OCD11]. The major requirement to the surface model is to provide robust formulation of reaction-diffusion processes at free surfaces. A mandatory step in order to achieve this goal is to incorporate a conservative fluid transport of reactive quantities along surfaces and between surface and bulk. This allows for example for complex dynamics of detergents as shown in Fig. 4.1. However, there is no consistent field representation for free-surface SPH yet.

Goals

The surface model presented in this chapter fills this gap and provides a consistent field reconstruction which supports conservative transport within the fluid's surface and between surface and fluid bulk. After introducing challenges to existing surface models in Sec. 4.1, Sec. 4.2-4.4 outline the major contributions of this chapter which can be summarized as follows:

Detailed Contributions

- In Sec. 4.2 a novel and robust computation of surface area is proposed which correctly handles thin fluid sheets and other singularities frequent to free surface models and which enables
- a embedded surface model (Sec. 4.3) leading to conservative and SPH-consistent mechanism (two-way-coupling) for quantity transport within the surface and between surface and bulk, and based on the coupling,
- a straightforward formulation of surface effects, including adsorption, diffusion and reaction kinetics is realized (Sec. 4.4).

Implementation aspects are described (cf. Sec. 4.5), while in Sec. 4.6, robustness and validity is demonstrated for several types of surface-relevant effects. Sec. 4.7 then draws some final conclusions.

4.1 Foundations and Challenges

Surfaces are either represented explicitly using parametric descriptions or they are represented by the zero level set of a scalar function. However, coupling of surface and bulk raises challenges for both representations.

4.1.1 Explicit Surface Representations

Polygonal Meshes

Aside from surfels [WH94, KAG*05], polygonal meshes are the most common explicit surface representation in Computer Graphics [BKP*10]. Polygonal surfaces are described by a set of vertices which are connected to faces. As such, triangle meshes are especially advantageous for rendering smooth surfaces. Therefore, they are either advected with the flow [KAG*05, WTGT10, YWTY12, CWSO13] or otherwise created from implicit models by using marching cubes algorithms [LC87]. When tracked, mesh-based surfaces require frequent re-meshing operations [KAG*05, YWTY12, CWSO13] in order to handle topological changes. To avoid drifting from the fluid, vertices also need to be projected back onto the fluid's surface [AA04] which is often followed by an additional fairing [SS11] using differential surface operators (cf.Sec. 2.1.2) to improve vertex distribution. Even if tracking can be feasible, representation inconsistencies lead to error-prone interactions between bulk and triangulated surfaces [BCOS01]. Since in our case this coupling is mandatory, an implicit representation has been preferred.

Surface Tracking

4.1.2 Implicit Surface Representations

Implicit surface models usually sample a surface delta function (cf.Sec. 2.1.1) within a three dimensional volumetric representation which in case of SPH is given by the fluid particles. The surface can be represented either by using gradients of color fields [MCG03] or by the zero level set of a scalar function [ZB05]. Color gradients [MCG03, KAG*06, AIAT12] require an adjacent second phase [SP08, AHA10] to accurately model the zero drop at phase boundaries, otherwise they become sensitive to irregular particle distributions [BT07b]. In general, scalar functions are thus employed [ZB05, SSP07, SZP07, APKG07, OCD11, YT13] for free surface scenarios. The free fluid-air interface $\delta\Omega := \{\mathbf{x} \mid \phi(\mathbf{x}) = 0\}$ is then defined as the zero iso-contour of the following scalar field function [ZB05] (cf. Fig. 4.3(a)):

Color Gradient

Scalar Field Function

$$\phi(\mathbf{x}) = |\mathbf{x} - \mathbf{m}(\mathbf{x})| - r, \tag{4.1}$$

where r is the distance between surface and particles, which either is correctly initialized and then tracked over time [APKG07] or is a design parameter usually chosen equal to half the particle's rest distance. Here, $\mathbf{m}(\mathbf{x})$ is the center of the local iso-density distribution [OCD11], which is computed using the corrected kernel given in Eq. (3.11):

Iso-Density Distribution

$$\mathbf{m}(\mathbf{x}) = \sum_j \mathbf{x}_j \, V_j \, \hat{W}_j(\mathbf{x}). \tag{4.2}$$

Nevertheless, one needs a robust representation of the phase singularity in order to simulate quantity transport on free surfaces. As we will see, a reliable estimation of the surface area plays a key role for computing robust surface fields.

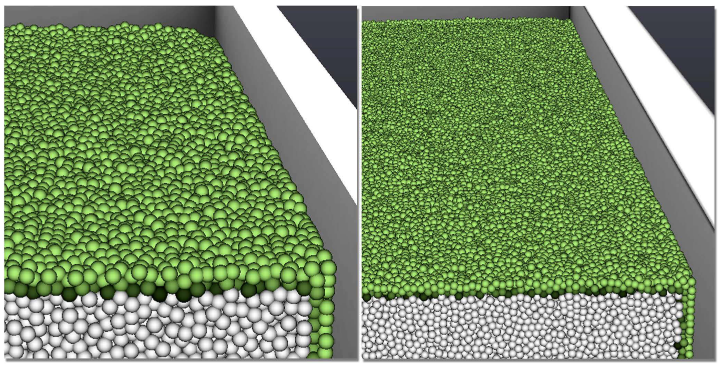


Figure 4.2: A discretization of the surface delta function Eq. (4.3) results in a narrow band of surface particles (green) which thickness depends on the particle resolution (doubled from left to right), but stays constant regarding the number of particles.

4.2 Computation of Surface Area

According to Bertalmio et al. [BCOS01], it is advantageous to represent a lower dimensional surface as the level-set of a higher dimensional function. Hence, the area of the fluid's surface is better defined by integrating a surface delta function over the fluid volume (cf. Osher and Fedkiw [OF02]): Surface Integral

$$\int_\Omega \delta(\phi(\mathbf{x}))\,|\nabla_\mathbf{x}\phi(\mathbf{x})|\,d\mathbf{x} = \int_\Omega \delta(\phi(\mathbf{x}))\,d\mathbf{x}$$

where it is assumed that ϕ is a signed distance function for which $|\nabla\phi| = 1$ holds. With this property of signed distance functions, it is possible to approximate the three-variate delta function depending on the sampling position by a one-variate function depending only on the distance to the surface, i.e. $\delta(\phi)\,|\nabla\phi| = \delta(\phi)$. Note that, for the scalar field function given in Eq. (4.1) the assumption $|\nabla\phi| = 1$ is only true by assuming that particles locally approximate a flat surface. This approximation is still sufficient, since the surface resolution is directly linked to the particle resolution. Thus, surface structures can only emerge up to the size of a particle's support radius as also discussed later in Sec. 4.6.

The idea is now to sample the iso-contour implicitly with volumetric particles as shown in Fig. 4.2. The delta function is then reconstructed by employing an integral interpolant over discrete delta values in the local particle neighbourhood: Surface Discretization

$$\int_\Omega \delta_{I,h}(\mathbf{x}_i)\,d\mathbf{x}_i = \int_\Omega \int_\Omega \delta(\phi(\mathbf{x}_j))\,W(|\mathbf{x}_i - \mathbf{x}_j|, h)\,d\mathbf{x}_j\;\; d\mathbf{x}_i, \tag{4.3}$$

where $d\mathbf{x}_i$ is the differential volume of particle. From Eq.(4.3) we can extract a particle's contributions to the surface as

$$dA_i = d\mathbf{x}_i \int_\Omega \delta(\phi(\mathbf{x}_j))\, W(|\mathbf{x}_i - \mathbf{x}_j|, h)\, d\mathbf{x}_j. \tag{4.4}$$

An integration of differential particle areas dA_i then yields the area of the fluid's surface. In practice, as summarized in Fig. 4.3, the integral interpolant $\delta_{I,h}(\mathbf{x})$ is approximated by a summation interpolant $\delta(\mathbf{x})$ using discrete particle volumes (see Sec. 4.2.2) in combination with a smeared-out version of the delta function (see Sec. 4.2.1).

As a result, the surface extends into the fluid's bulk and leads to a narrow band of contributing particles. Each particle within this band represents a fraction of surface area A_i, depending on its distance to the surface. However, unlike particle volumes V_i, a particle's fraction of surface area is not constant due to internal and external forces changing the surface topology. Thus, in order to stably handle thin fluid sheets delta values are corrected in singular regions as described in Sec. 4.2.3.

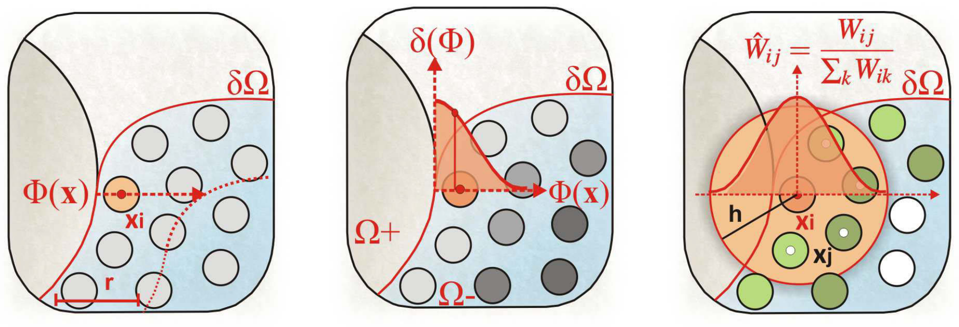
4.2.1 Smeared-Out Delta Function

Due to discretization the delta function would be zero almost everywhere. Therefore it is necessary to smear out the iso-contour $\delta\Omega$ over the local fluid volume using an SPH-based smoothing kernel as shown in Fig. 4.3(b). However, in combination with small support radii, the distance function may return slightly fluctuating distance values ϕ_i over time, since the computation of the local mass center is prone to changing particle neighbourhoods. Thus, in practice it is better to approximate the surface delta function by the following half-sided linear tent-kernel:

Half-Sided Tent-Kernel

$$\delta(\phi) = \frac{1}{r} \begin{cases} (1 + \frac{\phi}{r}) & \text{if } \phi < 0 \\ 0 & \text{otherwise,} \end{cases} \tag{4.5}$$

which only depends on the distance to the iso-contour at $\phi(\mathbf{x}) = 0$ for which by definition $\phi(\mathbf{x}) \geq -r$ holds. During simulation discrete values $\delta_i = \delta(\phi(\mathbf{x}_i))$, which are evaluated at particle locations $\mathbf{x}_i \in \Omega^-$, are integrated in order to compute a surface area. Due to the normalization property, i.e. $\int_{-r}^{r} \delta(\phi)\, \partial\phi = 1$, the surface area is then linearly distributed over fluid particles and the smoothing of the area is directly related to the size of r.



(a) Scalar Field Function (b) Delta Function (c) Area Interpolation

Figure 4.3: Interpolation of particle areas employs a corrected SPH reconstruction 4.3(c) of a half-sided delta function 4.3(b) which smears out the zero iso-contour of the scalar field function 4.3(a).

4.2.2 Interpolation of Particle Areas

In order to determine how much of the surface area is represented by a particle, one has to integrate the surface delta function as derived in the last section over a particle's support h (see Fig. 4.3(c)). In detail, the corresponding SPH approximation for Eq. (4.4) yields

Surface Interpolant

$$A_i = V_i\,\delta(\mathbf{x}_i) = V_i \sum_j \delta_j\, V_j\, \hat{W}_{ij}. \tag{4.6}$$

However, the surface $\delta\Omega$ is only sampled for particles in the fluid domain $\Omega-$. One therefore has to correct the resulting volume deficiency due to the under-resolved particle neighbourhood in Ω^+. It is therefore preferable to use the adapted weighting kernel [BK02] introduced in Sec. 3.1.4 which leads to an consistent reconstruction of constant functions at fluid boundaries.

State Transitions

The closer a particle is to the surface, the more it contributes to the surface. Due to the definition of the distance function, a disturbance of regular particle structures, as caused by the flow-field, leads to small area values in the fluid bulk. Thus, a particle is considered not to be part of the surface as long as its area is below or equal $A_{\min}$. However, values δ_i need to be corrected in highly under-resolved regions.

4.2.3 Stable Handling of Thin Fluid Sheets

During simulation regular but deficient particle neighbourhoods can emerge which might lead to erroneous area values as shown in Fig. 4.4(a). Singular

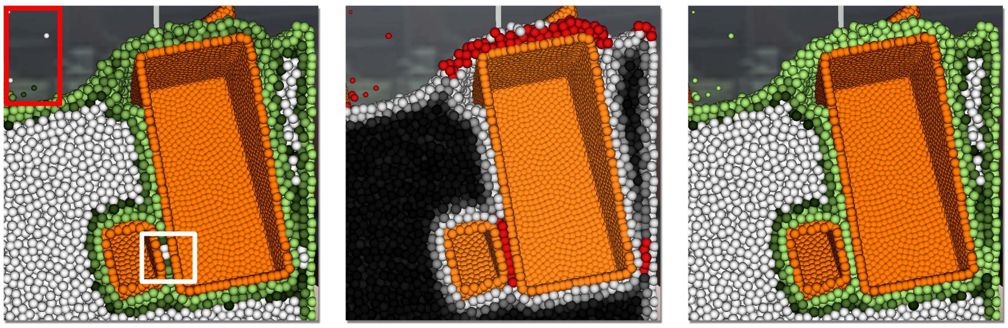


Figure 4.4: Singularities, e.g. as induced by rigid objects (orange), result in erroneous areas A_i (highlighted in 4.4(a)). Detection of singular regions (red in 4.4(b)) enables a correction of delta values (increasing from black to grey), which leads to robust areas A_i (cf. 4.4(c)).

cases might be identified by setting a lower threshold for the number of neighbouring particles, but dense and strongly anisotropically distributions still would remain unidentified. A better description for local particle distributions is the weighted covariance matrix of a particle i, given as [YT13]

Covariance Matrix

$$\mathbf{C}_i = \sum_j (\mathbf{x}_j - \mathbf{m}_i)\,(\mathbf{x}_j - \mathbf{m}_i)^T\, V_j\, \hat{W}_{ij} \quad \in \mathbb{R}^{3x3}. \tag{4.7}$$

In the following the determinant of this symmetric positive-definite matrix is used as an indicator for flat or planar particle neighbourhoods. If $|\mathbf{C}_i| < \epsilon_c$ or if a particle i has less than 30 neighbours (red particles in Fig. 4.4(b)) it is set to $\delta_i = \delta_{\max}$ (assuming a flat surface) leading to a corrected approximation of delta values. Similarly, since rigid particles k are directly sampled on the explicit rigid surfaces, area values A_k of rigid particles are pre-computed by assuming $\delta_k = \delta_{\max}$.

Note, in general the determinant does not necessarily reveal singular structures and one has to use the condition number of the matrix instead. However, in the case of incompressible fluids, neighbouring particles in bulk are regularly distributed so that the determinant stays in a certain range while for planar regions it changes exponentially. Thus, in this case a computation of the condition number involving estimation of singular values is avoided.

Temporal Blending

In order to avoid fluctuation of quantity fields, time-coherence of area values is important. However, thin fluid structures may be under rapid progress which could lead to a flickering of area values. Temporal coherence of particle systems however makes it possible to apply a temporal smoothing of particle areas in singular regions, which reads

$$\delta_i(t) = (1-\varphi)\delta_i(t-\Delta t) + \varphi\delta_i(t)^*, \tag{4.8}$$

where $\delta_i(t)^*$ is the approximated surface delta function according to Eq. (4.5) and where φ defines the rate of adjustment to new values $\delta_i(t)^*$. In the presented examples $\varphi \in [0.1, 0.5]$ for thin sheets and $\varphi = 1$ for all other regions.

Temporal Coherence

In contrast, it is safe to compute $\delta_i(t)^*$ only every n-th frame in regions of small divergence, i.e. where $\nabla \cdot \mathbf{u}_i < \epsilon_{\mathbf{u}}$, since in these regions $\frac{d}{dt}\delta_i \approx 0$. To proof this fact lets assume a temporal coherent particle neighbourhood, i.e. $\mathbf{u}_i - \mathbf{u}_j = \mathbf{0}$, and thus $\frac{d}{dt}W_{ij} = (\mathbf{u}_i - \mathbf{u}_j) \cdot \frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|}\frac{d}{dx}\nabla_j W_{ij} = 0$. In combination with the quotient rule we then get

$$\frac{d}{dt}\mathbf{m}_i = \frac{[\sum_j V_j \frac{d}{dt}(\mathbf{x}_j W_{ij})][\sum_j V_j W_{ij}] - [\sum_j V_j \frac{d}{dt}W_{ij}][\sum_j \mathbf{x}_j V_j W_{ij}]}{[\sum_j V_j W_{ij}]^2}$$

where $\frac{d}{dt}(\mathbf{x}_j W_{ij}) = \mathbf{u}_j W_{ij} + \mathbf{x}_j \frac{d}{dt}W_{ij} = \mathbf{u}_j W_{ij}$, which leads to

$$\frac{d}{dt}\mathbf{m}_i = \frac{[\sum_j \mathbf{u}_j V_j W_{ij}][\sum_j V_j W_{ij}]}{[\sum_j V_j W_{ij}]^2} = \mathbf{u}_i \frac{[\sum_j V_j W_{ij}][\sum_j V_j W_{ij}]}{[\sum_j V_j W_{ij}]^2} = \mathbf{u}_i.$$

for which directly follows that $\frac{d}{dt}\phi_i = \frac{\mathbf{x}_i - \mathbf{m}_i}{\|\mathbf{x}_i - \mathbf{m}_i\|}(\mathbf{u}_i - \frac{d}{dt}\mathbf{m}_i) = 0$. Consequently, $\frac{d}{dt}\delta_i = \frac{d}{d\phi}\delta(\phi_i)\,\frac{d}{dt}\phi_i = 0$ as long as $\nabla \cdot \mathbf{u}_i = 0$.

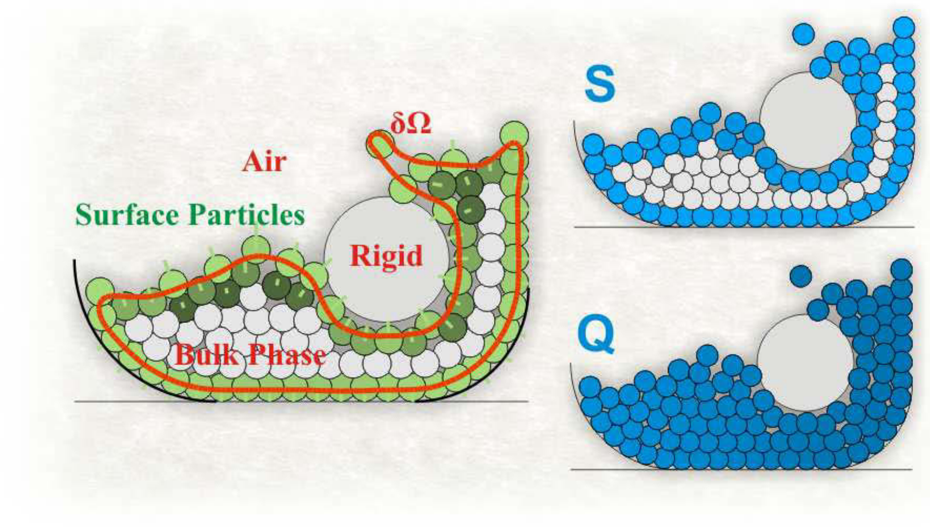


Figure 4.5: Particles sample the implicitly defined surface $\delta\Omega$ (left). Surface particles (green) constitute quantity fields S on the surface, and together with bulk particles (white) also constitute quantity fields Q in bulk (right).

4.3 The Embedded Surface Model

The strength of our surface area computation is that quantity fields on a surface can now be reconstructed by interpolating from a discrete set of volumetric particles (cf. Eq. (3.3)) leading to a consistent representation of bulk and surface. The underlying Lagrangian SPH simulation then acts as a "particle-pool" for quantity fields defined on surfaces, as shown in Fig. 4.5. **Surface Particles** Particles are divided into two groups: surface particles (green) within a small layer around the surface, i.e. where $A_i > A_{\min}$, and bulk particles (white). Surface particles have a double role during simulation: Like bulk particles, they contribute to a quantity field Q and at the same time, they constitute a quantity field S on the surface which is then responsible for surface effects.

Surface Fields Even if our method is not restricted to scalar quantities, we consider Q_i given in $[mol\, m^{-3}]$ and S_i given in $[mol\, m^{-2}]$ as concentrations of a substance defined with respect to a particle's volume or area respectively. Depending on the transport properties of quantities, particles may have an attribute $\overline{S}_i = A_i S$ representing molar mass adhered to a particle's surface and an attribute $\overline{Q}_i = Q_i V_i$ representing the mass transported in bulk. Substances require both attributes if they exhibit different behaviour for surface and bulk (cf. Fig. 4.7). Total mass of such anisotropic distributed substances is $\sum_i \overline{Q}_i + \overline{S}_i$. If formulation of transport in bulk and on surface are equal, only a single attribute representing the total number of molecules in bulk $\overline{Q}_i$ is required (cf. Fig. 1.1). Naturally, for isotropically distributed substances the concentration per area is proportional to the concentration per volume with respect to the particle's delta value, i.e. $\delta_i S_i = Q_i$ (cf. Eq. (4.6)).

Surface Flux

Since surface area is a property of particles, it becomes very easy to formulate the surface flux as given in Eq. (2.7):

$$\Theta_i = \frac{d}{dt} S_i = \frac{1}{A_i} \sum_j \Gamma_{i \leftarrow j},$$

where $\Gamma_{i \leftarrow j}$ represents the net flux of quantities from particle j to particle i, thus is defined with respect to absolute quantities and the chemical process under consideration. While mass is trivially conserved for a flux computed in bulk (cf. Eq. (3.20)), due to changing particle areas, a surface flux requires symmetrization mechanisms to enforce $\Gamma_{i \leftarrow j} = -\Gamma_{j \leftarrow i}$. In the following conservative formulations for surface transport are developed which will be applied in Sec. 4.4.

Coupling of Bulk and Surface Dynamics of a fluid's bulk and surface are strongly interconnected. The bulk acts as a source and as a sink for the surface. Thus, mass is transferred between two different kinds of particles, surface particles i and bulk particles j (which might be one and the same particle, i.e. $i = j$). Consequently, a transport mechanism between bulk and surface is not symmetric per se. To ensure conservation, a quantity flux is computed with focus on surface particles i and the opposite flux is then assumed for bulk particles j:

Surface-Bulk Flux

$$\frac{d}{dt} S_i = \frac{1}{A_i} \sum_j \Gamma_{i \leftarrow j}, \text{ and } \frac{d}{dt} Q_j = \frac{1}{V_j} \sum_i -\Gamma_{i \leftarrow j}. \tag{4.9}$$

That way the the surface receives as much of a quantity as has been taken from bulk and vice versa.

Symmetric Flux on Surfaces Due to non-constant particle areas, standard symmetrization techniques, i.e. gradient approximations [CEL06] or integral approximations [CM99], do not lead to symmetric formulations of fluxes. Therefore, conservation is explicitly enforced by averaging pairwise contributions between surface particles, i.e. using $\Gamma_{i \leftarrow j} = -\Gamma_{j \leftarrow i}$:

Surface-Surface Flux

$$\frac{d}{dt} S_i = \frac{1}{A_i} \sum_j \Gamma_{i \leftarrow j} = \frac{1}{A_i} \sum_j \frac{1}{2} [\Gamma_{i \leftarrow j} - \Gamma_{j \leftarrow i}], \tag{4.10}$$

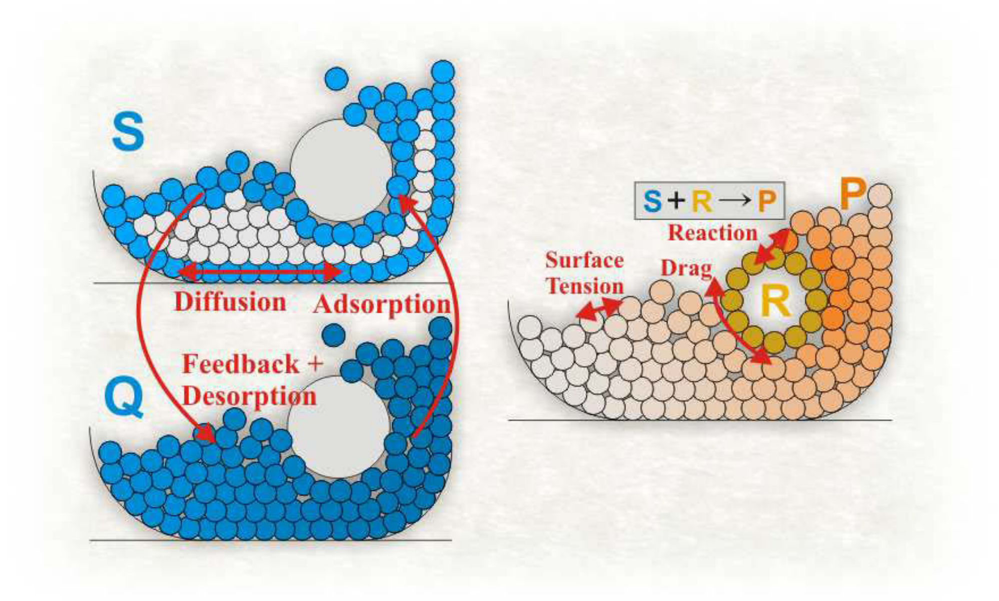


Figure 4.6: Dependening on the application, field S undergoes diffusion along the surface and interacts with field Q in a fluid's bulk due to adsorption. Surface particles feedback quantities when they are drawn under the surface. S may reduce surface tension and fluid drag or reacts with (yellow) substances R on rigid surfaces to create (orange) products P.

where i and j might also be part of different phases. In contrast to the averaging of particle attributes as proposed by Müller et al. [MCG03] here Newton's action = reaction principle is used. Please note that for adjacent phases iso-contours do in general not match exactly. However, pairwise averaging of contributions naturally corrects such discontinuities.

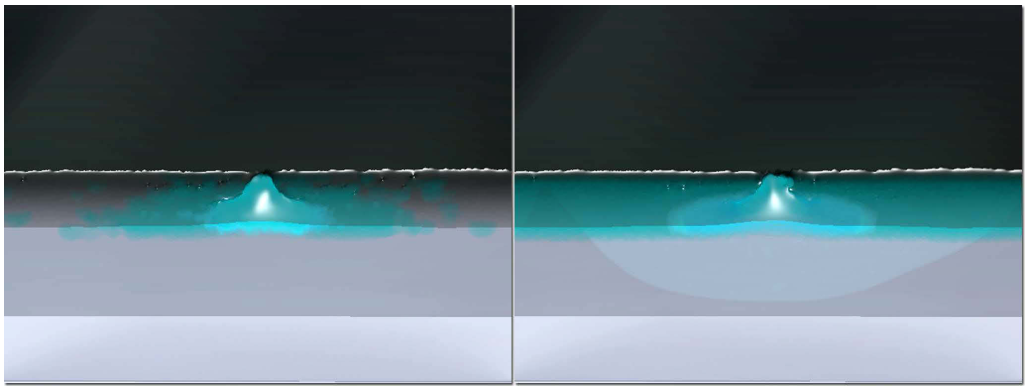


Figure 4.7: For $\sigma_a > 0 \wedge \sigma_d = 0$ detergents stay local on the surface, while $\sigma_a > 0 \wedge \sigma_d > 0$ results in homogeneous distribution along the surface.

4.4 Application to Reaction Kinetics

To demonstrate the applicability of the surface model it has been applied to various reaction kinetics [SKS05, BS09] as depicted in Fig. 4.6. The surface model is especially well suited to simulate dynamics of detergents. Detergents as other surfactants show very complex behaviour on fluid surfaces, including adsorption at surfaces and strong surface diffusion. Beside dynamics of detergents, chemical reactions like coating and cleansing are modelled. Surface concentration is used to alter surface tension [BT07b] and fluid drag [AIA*12] as shown in Fig.4.10.

Adsorption The driving mechanism for simulation of detergents and other surfactants is the well known Langmuir-Hinshelwood mechanism [BS09] which describes how much of a so-called "free" substrate or target $\overline{Q}_j$ in a carrier fluid is adsorbed on a surface. Fig. 4.7 shows the importance of adsorption for a falling drop of detergent. The net rate of adsorbed substrates at surface particle i mainly depends on the presence of targets at neighbouring bulk particles j, which by applying Eq. (4.9) is computed as

Langmuir-Hinshelwood

$$\Gamma^a_{i \leftarrow j} = \left[\sigma_a \overline{Q}_j \left(\overline{S}_0 - \overline{S}_i\right) \; - \; \sigma_d \overline{S}_i\right] V_j W_{ij},$$

where σ_a defines the speed of adsorption and where the total number of available capture sites $\overline{S}_0$ limits the adsorption process. Over time, adsorbed targets constantly dissociate from the surface as controlled by σ_d. Particles which are drawn under the surface, transfer mass back to nearby bulk particles by setting $\sigma_d > 0$ and $\sigma_a = 0$.

Surface Diffusion

Diffusion Since S is non-zero only at the surface, the formulation of surface diffusion as shown in Fig. 4.7 is similar to bulk diffusion given in Eq. (3.20). However, since S is given relative to the surface area, the net mass flux with Eq. (3.18) reads

$$\Gamma_{i\leftarrow j} = A_i \, \sigma_s \, (S_i - S_j) \, V_j \, |\nabla_j W_{ij}|,$$

where σ_s is the corresponding diffusion constant. However, this formulation is not symmetric. We therefore compute surface diffusion by averaging the particle contributions using Eq. (4.10):

$$\Gamma^s_{i\leftarrow j} = \sigma_s \, (S_i - S_j) \; (A_i V_j + A_j V_i) \; |\nabla_j W_{ij}|.$$

Note that i, j are restricted to surface particles in order to avoid diffusion into the bulk.

Micelles Formation

Cleansing With detergent concentration adhered to the surface it is used to cleanse rigid surfaces as shown in Fig. 4.1 and 4.10. Detergent molecules $\overline{S}$ solve and enclose grease molecules on rigid surfaces $\overline{R}$ thereby forming products or micelles $\overline{P}$ which dissolve into the bulk [RK04]. The speed of this bi-molecular reaction is then defined via a simple rate law which linearly depends on the rate constant σ_r and the total number of molecules of both reactants, i.e. $\sigma_r \, \overline{S} \, \overline{R}$, which in SPH reads

$$\Gamma^r_{i\leftarrow k} = \sigma_r \overline{S}_i \overline{R}_k \, \frac{1}{2}(V_i + V_k) \, W_{ij}.$$

By coupling of bulk and surface, the reaction rate is then defined as

Micelles Transport

$$V_i \frac{\partial}{\partial t} P_i = -A_i \frac{d}{dt} S_i = -A_k \frac{d}{dt} R_k = \sum_k \Gamma^r_{i\leftarrow k},$$

where i are fluid particles and k are rigid particles. Note that both Eq. (4.9) and Eq. (4.10) have been used to formulate a conservative reaction.

Coating A coating of rigid objects as shown in Fig. 1.1 is modelled as a uni-molecular reaction where one molecule of paint on the fluid's surface $\overline{S}$ is transformed into one molecule of paint sticking to rigid surfaces $\overline{R}$. The coating process depends on the total number of molecules $\overline{S}$ close to the rigid object's surface, for which the reaction reads:

Uni-Molecuar Reaction

$$\Gamma^r_{i\leftarrow k} = -\sigma_r \overline{S}_i \, \frac{1}{2}(V_i + V_k) \, W_{ij},$$

Algorithm 5: Data-parallel computation of surface area.

```
(a) Distance Function

foreach particle $i$ inparallel do
    compute $\nabla \cdot \mathbf{u}_i(t)$
foreach fluid particle $i$ where $\nabla \cdot \mathbf{u}_i(t) > \epsilon_{\mathbf{u}}$ inparallel do
    compute $\phi_i(t)$ using support $2.5h$ (Eq. (4.1))
(b) Delta Function

foreach fluid particle $i$ where $\nabla \cdot \mathbf{u}_i(t) > \epsilon_{\mathbf{u}}$ inparallel do
    estimate $\delta_i(t)^*$ (Eq. (4.5))
    compute $\mathbf{C}_i(t)$ using support $2.5h$ (Eq. (4.7))
    if $(|\mathbf{C}_i(t)| < \epsilon_c)$ $\delta_i(t)^* = \delta_{\max}$
foreach fluid particle $i$ inparallel do
    update $\delta_i(t)$ (Eq. (4.8))
(c) Area Interpolation


foreach fluid particle $i$ inparallel do
    compute $A_i(t)$ (Eq. (4.6))
    if( $A_i(t) > A_{\min}$ ) mark i as surface particle
```

where σ_r is a rate constant defining the reaction speed. Molecular mass of both substances then changes by

$$V_j \frac{d}{dt} Q_j = -A_k \frac{d}{dt} R_k = \sum_k \Gamma^r_{i \leftarrow k},$$

where Eq. (4.9) has been used. Deposition of materials is stopped once rigid particles reach a maximum concentration of $R_{\max} = 1$.

4.5 Implementation Details

In Alg. 5 the pseudo-code for the computation of surface area is outlined. For field reconstruction $h = 2x$ has been set, which results in a rest distance of $0.92h$ between particles and around 30 neighbours per particle in combination with compact smoothing kernels (cf. Sec. 3.1.2). For the computation of distance values and anisotropy the smoothing radius has been increased to $2.5h$ in order to obtain reliable results. Note that all constants

Precomputed Constants

Scene	**"Pan"**	**"Flooding"**	**"Drop"**	**"Kinect®"**	**"Bunny"**
σ_r/σ_s	10/10	70/10	−/10	40/10	40/−
σ_a/σ_d	10/0.1	10/0.1	10/0.0	10/0.1	−/−
Snapshot t_{sim}	23 sec.	30 sec	3 sec	20 sec	11 sec
Avg.δt [ms]	2	1.5	2	1.5	2
# fluid ptcls	2.1M	1.1M	1.1M	0.75M	4.7M
# surface ptcls	350k	200k	141k	300k	506k
Dist+Delta Func. [ms]	35.9	30.2	28.3	60.0	65.4
Surface Area [ms]	14.0	14.8	11.5	9.2	61.5
Adsorption [ms]	22.1	19.0	12.2	14.9	26.7
Diffusion [ms]	3.4	2.8	1.5	3.6	33.2
Reactions [ms]	15.3	17.5	0	12.0	0.65
Time per Frame	**312**	**261**	**203**	**252**	**693**

Table 4.1: Per frame timings (msec) for steps outlined in Alg. 5.

necessary to compute a surface area (cf. Sec. 4.2.3), are pre-computed from pre-defined incompressible particle configurations as shown in Fig. 4.8

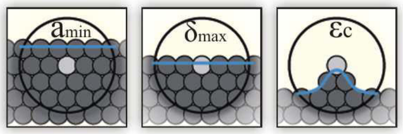


Figure 4.8: Constants are pre-computed for reference configurations.

4.6 Results and Discussion

Scenarios

The proposed method have been tested for various scenarios in order to show its versatility, including a washing of dishes (Fig. 4.1), the flooding of a valley (Fig. 4.10), a complex Kinect®scene (Fig. 4.12), coating of a rigid surface and the evolution of a single drop of detergent (Fig. 4.7). All results were obtained on an Intel Dual-Core 3.3 GHz with an NVidia GTX 580 with 1.5 GB VRAM. Simulation achieves interactive frame-rates even for a few million particles as shown in Tab. 4.1 and it is expected that computation cost will linearly scale for higher particle counts. For visualization of concentrations the SPH-based volume rendering approach [OKK10] as presented in Sec. 6 has been used.

Stability

Fig. 4.9 compares the proposed surfaces to the color gradient based interfaces used by Adami et al. [AHA10] (note, that 2.5h has been applied

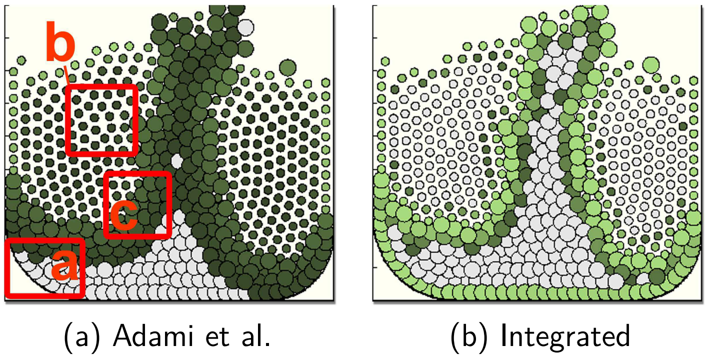


Figure 4.9: Comparison to interface approximation of Adami et al. : The proposed integrated surface approach handles free-surfaces (a), irregular particle structures (b), and yields thin interfaces (c).

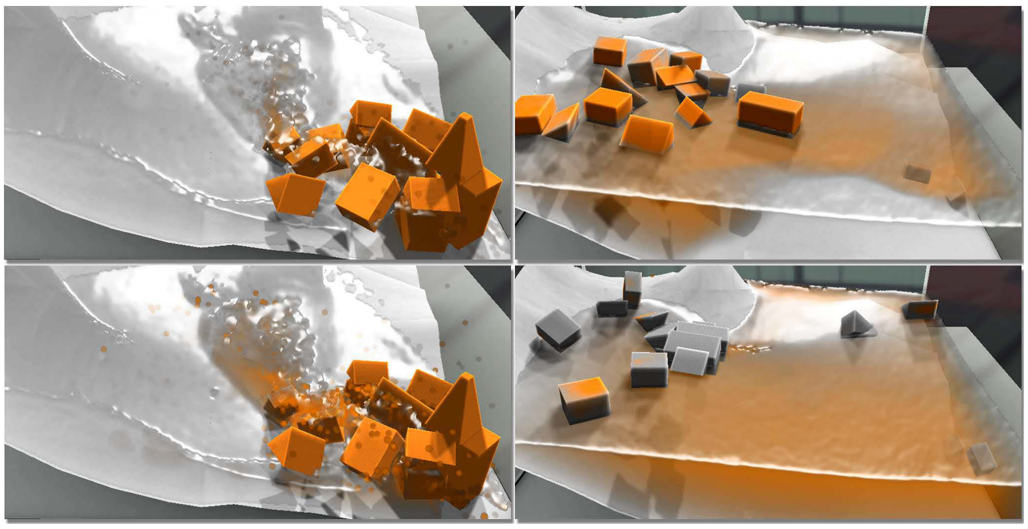


Figure 4.10: Flooding of a valley shown for two time-steps from left to right and two different detergent concentrations of $Q = 0.1$(top) and $Q = 1.0$ (bottom). Emitting less detergent results in less cleansing and in a different outcome of the convection due to stronger surface tension and fluid drag.

for the sampling radius in contrast to the proposed $6h$, which would lead to even thicker surface layers). Most importantly, our method supports free surfaces and avoids error-prone normal projection for surface dynamics; In fact the proposed approach does not require any normal computation. Due to the robust formulation, even complex scenes with highly irregular point distribution, e.g. as originating from a Kinect®sensor (cf. Fig. 4.12), are stably handled without reducing integration time-step. Furthermore, our integrated surface model is able to conserve mass within highly turbulent free-surface flows and for complex rigid-fluid interactions as shown in Fig. 4.10.

Limitations

The main limitation of our current implementation is that the surface resolution is directly linked to the bulk resolution, since uniform particle sizes are used. As can be seen in Fig. 4.2, the particle resolution directly relates to the thickness of the surface and influences its appearance and the transport dynamics (cf. Fig. 4.7). Due to the underlying implicit distance function (cf. Eq. (4.1)), particles can only capture details equal to the particle resolution.

Convergence

In Fig. 4.11 particles sample an analytically defined signed-distance function to illustrate this relation. The higher the particle resolution the more complex surface structures can emerge and the more accurately surface details can be captured. Beside visual appearance, the accuracy of the approximation also depends on the particle resolution. Note, with increasing resolution, the surface area converges towards the ground truth area of the underlying sine function. Surface resolution can be easily decoupled from the bulk resolution as discussed in the next section.

Future Work

Thus, the effect simulation would benefit from more complex reaction kinetics, which account for temperature dependencies or non-linear reactions like explosions and can easily be enriched with other effects like foam dynamics.

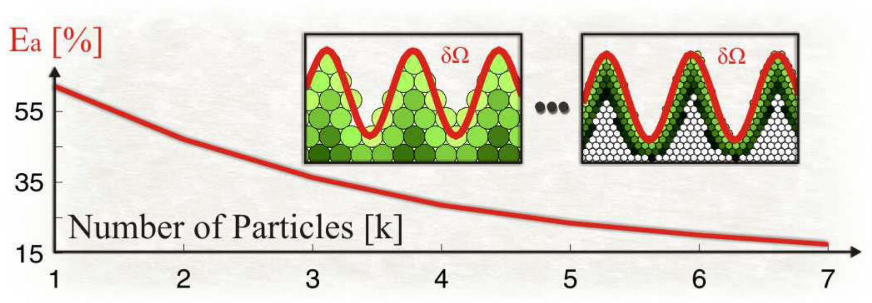


Figure 4.11: Approximation error E_a. With increasing resolution, the surface area converges towards the ground truth area of the analytic surface $\delta\Omega$ and particles are able to reconstruct surface structures more accurately.

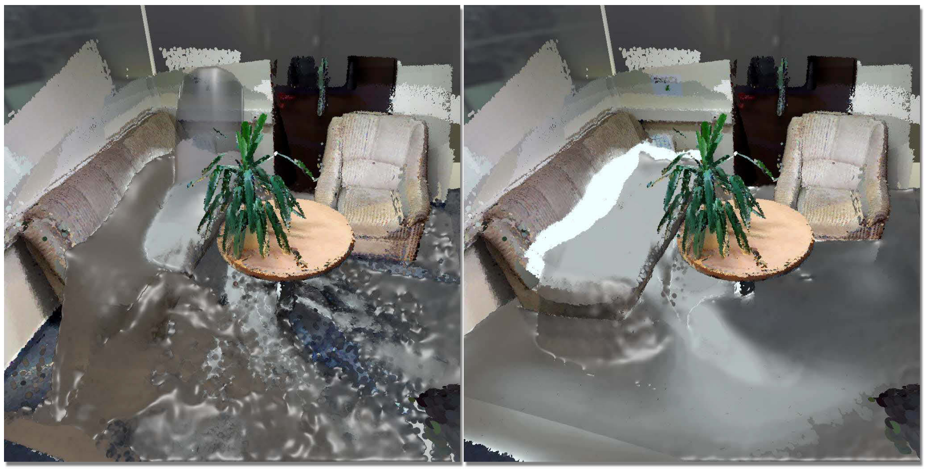


Figure 4.12: Complex room-scene acquired with a Kinect®camera. Multiple material colors are washed out over time resulting in dirty water.

4.7 Conclusions

In this chapter a novel approach for robust computation of a fluid's (free) surface has been presented, which is the core concept for any physically-based simulation of surface transport. For a conservative coupling between both fields it is advantageous to utilize a consistent formulation for a fluid's surface and bulk. Based on this formulation, one can then easily realize a transport model which incorporates a separate diffusion for surface and bulk, as well as a model for reactions at fluid-rigid interfaces which has been evaluated using various application scenarios. Due to its robustness it seems promising that the proposed surface model has a high potential to improve related research areas such as surface reconstruction.

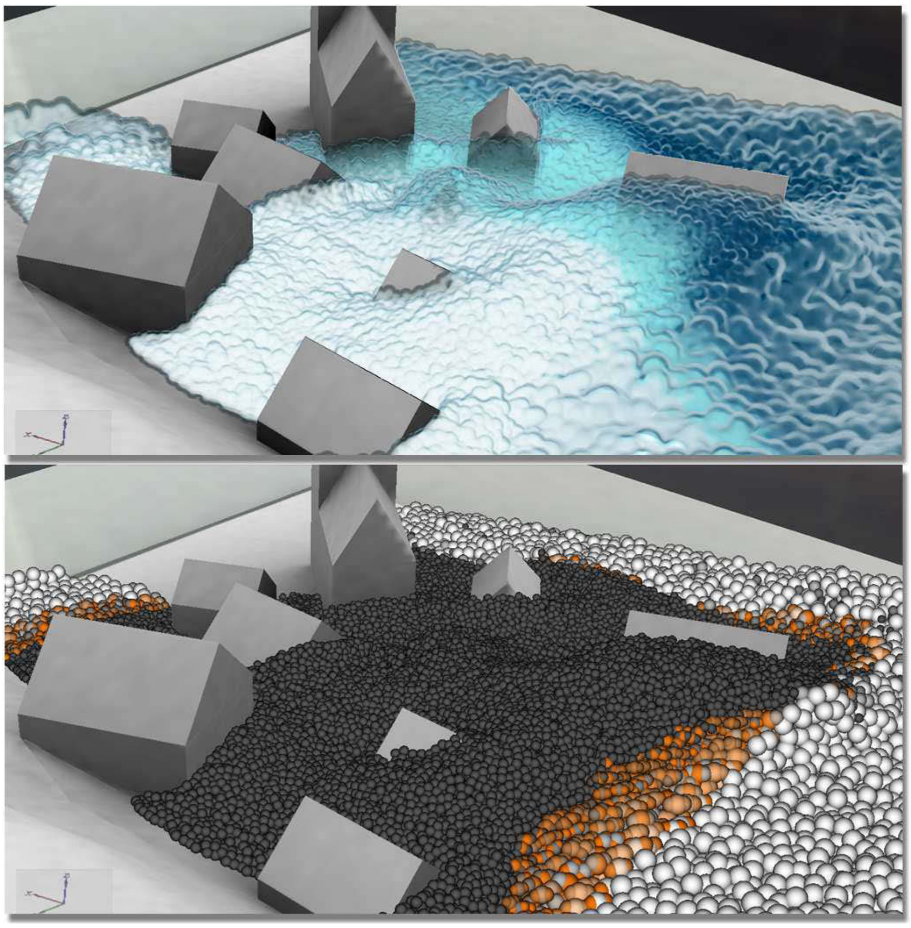


Figure 5.1: Flooding a valley with a salt diffusion (from white to blue). The particle resolution doubles around the village using blend-sets (orange).

Chapter 5

Temporally Coherent Sampling

In this chapter, a local conservative particle sampling is introduced which enables smooth transition between particle configurations even for stiff systems such as those arising from dealing with incompressible fluids. The sampling is designed to shift particle resolution from the fluid's bulk to its *surface as has been described in Sec. 4. With convection driven flows, the pressure term usually is the main source of instabilities, as described in Chap. 3.3.1. Thus, the following sections focus especially on the very error which is introduced into the density field. The proposed temporal blending mechanism [OK12] dealing with this aspect has been published at Computer Graphics Forum (CGF 2012) and has been presented at Eurographics (EG 2013) in Girona.*

The resolution of a fluid has a large impact on the resulting visual quality. On the one hand, surface complexity is drastically increased when employing large particle numbers. On the other hand, reducing the overall particle count either globally or locally is very appealing in order to improve simulation efficiency. In particle-based simulations, the overall computational cost usually increases linearly with the number of particles. In the context of SPH, several schemes have been introduced to achieve this goal, which can be classified into dynamic particle refinement methods and multi-scale techniques.

Spacial Adaptivity

The idea of a particle refinement is to dynamically exchange particle sets, either globally [CKS00, CPK02] or locally [DC99, LQB05, KAG*06, APKG07, FB07, ZSP08], in order to increase the resolution in regions of complex flow. Since globally adaptive methods distract from the adaptive character of Lagrangian systems, local sampling operators usually are preferred. Unfortunately, in Computer Graphics, simple but faster replacement schemes cause high approximation errors when instantly applied. As an alternative to dynamically merging and splitting particles, level-of-detail can be achieved by using multiple resolution levels which are simulated in separate but coupled simulations [SG11, HS13]. However, such multi-scale approaches lead to divergent resolution levels which especially in case of unbounded free surface scenarios, such as shown in Fig.5.1, may lead to conservation problems.

Performance Parameters

In contrast to an adaptive space discretization, the time domain can be adaptively sampled as well, either by using per-particle time-steps [DC99, GP11] or by employing a single but dynamic integration time step [DC96, IAGT10]. Beside spatially or temporally adaptive sampling, other SPH parameters are crucial for the performance of an SPH simulation: for example, using small particle support radii [MCG03] minimizes the neighbourhood complexity. Additionally, compared to weakly compressible SPH [BT07b], PCISPH [SP09b] enables large integration time-steps for incompressible fluids. Last but not least, graphics processing units (GPUs) are able to exploit the highly parallel nature of SPH simulations [KC05].

The adaptive sampling mechanism should work closely with theses performance mechanisms.

Goals

The overall goal of this chapter is to provide a consistent and an adaptive SPH simulation which allows for large integration time-steps. For this purpose, a feature-specific adaptation of the number of particles in bulk is introduced, which, in combination with globally adaptive time-steps, small support radii and prediction-correction steps, leads to fast simulations.

Detailed Contributions

However, an instantaneous replacement of particle configurations introduces a significant change in the pressure term, which leads to small time-steps when CFL conditions are enforced. Therefore, a temporal blending scheme is applied to smooth the error in the pressure term. In detail, the approach incorporates the following contributions:

- a novel approach for a consistent adaptive SPH using a temporal blending of quantities, which enables large time-steps, and
- a scheme to estimate the blending step size based on a predicted error in the pressure term which enables an error-dependent blending time, and,
- a solely GPU-based implementation for incompressible SPH fluids with support for non-uniform particle sizes and in combination with adaptive time-steps.

Compared to prior approaches, a temporal blending proves to be very robust in terms of the pressure error. As such, it is expected that an integration of the proposed method into any other SPH scenario, where a smooth and consistent transition between particle sets is required[LD09, SB12], is possible.

Overview

In the remainder of the chapter, all necessary components for a temporal blending as shown in Fig. 5.2 are described: In Sec. 5.1 the relevant aspects of an adaptive SPH are introduced, including a motivation for the temporal blending approach which is presented throughout Sec. 5.2-5.3. The error estimation is then described in Sec. 5.4. Implementation details are given in Sec. 5.5. Finally, in Sec. 5.6, advantages and limitations of the proposed sampling are discussed, and the chapter is concluded in Sec. 5.7.

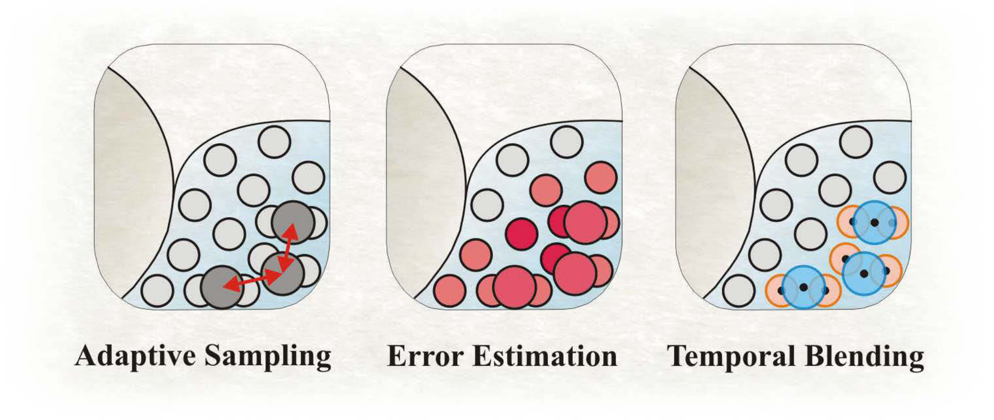


Figure 5.2: Components for the proposed adaptive SPH system. Particle levels are smoothly blended (right) by estimating sampling errors (middle). The adaptive sampling employs an relaxation before blending (left).

5.1 Foundations and Challenges

This section gives a brief introduction into the concepts of an adaptive sampling including the identification of high resolution regions (Sec. 5.1.1) and the particle refinement (Sec. 5.1.2). Sec. 5.1.3 discusses the major challenges in realizing an adaptive sampling while giving reasons for the proposed blending of quantities, as described in Sec. 5.2.

5.1.1 High-Resolution Regions

An adaptive mechanism shifts computational resources to regions of interest. High-resolution regions are either predefined (see Fig. 5.1) or are dictated by the flow dynamics. Dynamic regions are defined via the fluid's surface in combination with areas of high diffusive flux, as visible at the contact-line between the two fluids shown in Fig. 5.3. Thus, particles $\mathbf{x}_i$ belong to a high-resolution area as long as one of the following conditions holds:

Sampling Criteria

$$A_i > A_{\min} \quad \text{or} \quad |\sum_j V_j(Q_i - Q_j)\nabla_j W_{ij}| > \epsilon_Q,$$

where $A_{\min}, \epsilon_Q$ are user defined thresholds for a particle's surface area as defined in Sec.4.2.2 and the concentration gradient, respectively. Similar to Adams et al. [APKG07] an intermediate layer of particles is left unchanged

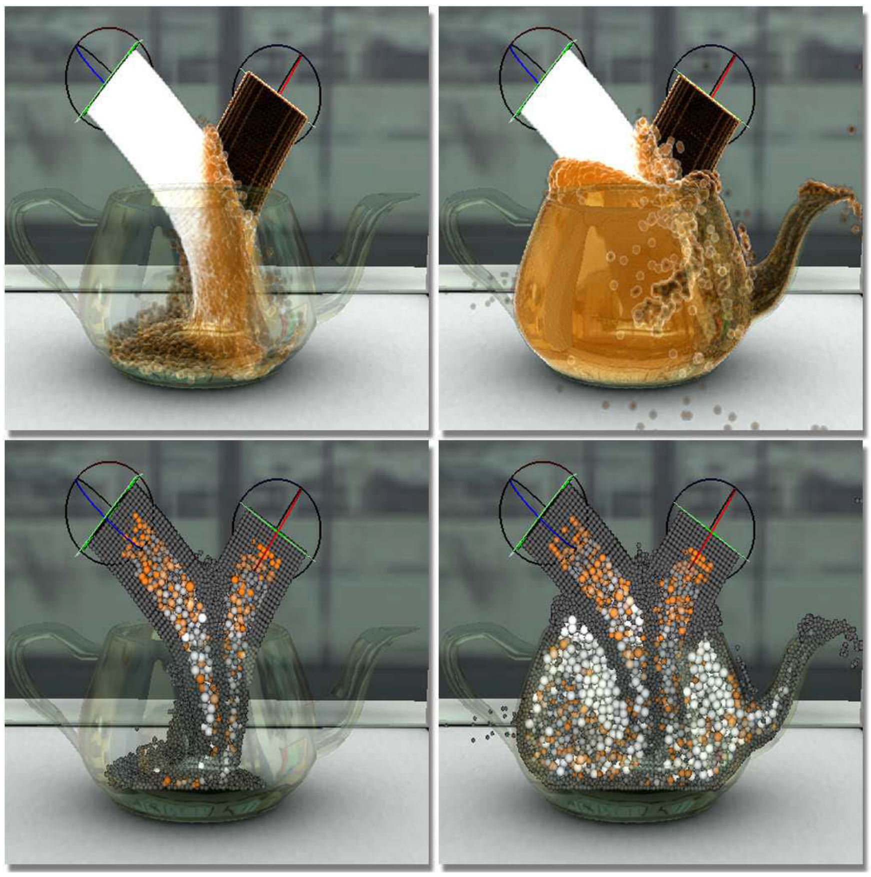


Figure 5.3: Mixing of coffee and cream in a Utah Teapot, shown for t=2 and t=6. Blend-sets (bottom) dynamically adapt to the convective and diffusive flux (top).

in order to avoid split-merge fluctuations. Note that, throughout the literature other re-sampling criteria exist, like interface regions, distance to the camera [SZP07] and inactive regions [GSSP10].

5.1.2 Non-Uniform Support Radii

Sampling Operator

Once high-resolution regions are identified, simple sampling operators, such as shown in Fig. 5.6, are used to refine a local particle set, aiming for a good performance. An adaptive sampling uses particles with non-uniform support radii h_i and rest volumes v_i depending on their current level $l_i = 0, 1, 2, ..., l_{\max}$:

$$h_i = 1.3\ (v_i)^{\frac{1}{3}}\ , \quad v_i = 2^{l_i} v_0, \tag{5.1}$$

where v_0 is the reference volume for level zero particles and 1.3 is used in order to conserve a fluid's volume [Mon05].

Averaged Smoothing Radii

However, in non-uniform particle systems, particles may either contribute to particles of their level only [KAG*06, SG11], or may exchange information with all neighbour particles directly [APKG07]. In the latter case, which is used due to its efficiency, the influence of neighbouring particles needs to be averaged in order to symmetrize contributions [Mon92]:

$$W_{ij} = W(|\mathbf{x}_i - \mathbf{x}_j|, \frac{h_i + h_j}{2}).$$

Alternative averaging operators [DC99] may be applied as well.

Adaptive Time-Stepping

As an adaptive spatial discretization directly implies adaptive temporal discretization, high resolution regions require smaller integration time-steps in order to secure simulation stability. According to the Courant-Friedrich-Levy (CFL) condition, the maximum time-step for a single particle i is defined by

$$\Delta t_i = \min(\lambda_{\mathbf{u}} \frac{h_i}{|\mathbf{u}_i|}, \ \ \lambda_{\mathbf{F}} \sqrt{\frac{h_i}{|\mathbf{F}_i|}}), \tag{5.2}$$

where $\lambda_{\mathbf{u}} = 0.4$ and $\lambda_{\mathbf{F}} = 0.25$ according to Monaghan [Mon92]. The overall simulation speed then depends on the minimum over all individual time-steps, as described for example by Ihmsen et al. [IAGT10], with the result the integration time step Δt is then adapted globally for all particles.

5.1.3 Challenges of an Adaptive Sampling

Adaptive SPH systems refine particles globally [CKS00] or locally [FB07]. In case of free surface flows, local sampling operations have to minimize a local error function in order to preserve a quantity field $Q(\mathbf{x})$ as good as possible:

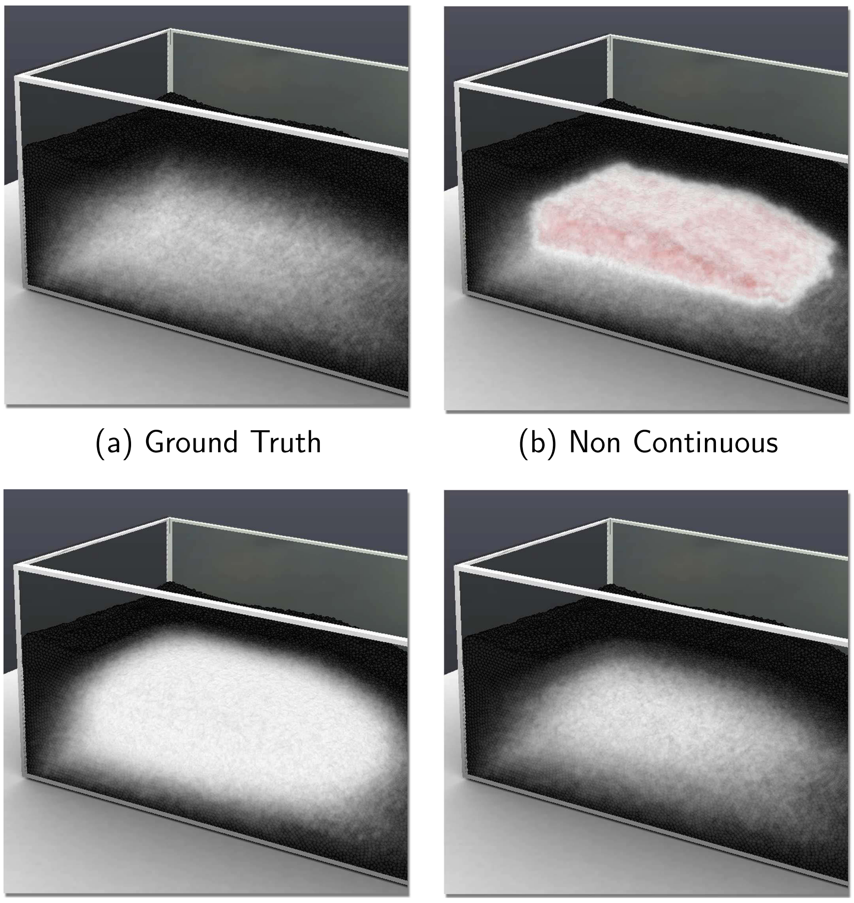
Sampling Error

$$E_Q(\mathbf{x}) = |Q(\mathbf{x}) - \hat{Q}(\mathbf{x})|, \tag{5.3}$$

where $\hat{Q}(\mathbf{x})$ is the result of the approximation. Lagrange multipliers [FB07] and iterative solvers are required in order to conserve the total amount of quantities and to account for non-negativity constraints [LB95].

Local Operators

Instead, in Computer Graphics, simple and fast sampling patterns are used. For example, in high-resolution regions, particles of level l split to N child particles of level $l - \log_2(N)$, or vice versa merge to a single particle of level $l+1$ in low resolution regions. In general, such operators do not include any neighbouring particles to minimize $E_Q(\mathbf{x})$, do not preserve the overall regular particle structure and consequently, approximate the old quantity



(a) Ground Truth (b) Non Continuous

(c) Error-Independent (d) Error-Dependent

Figure 5.4: Visualization of the pressure from red to grey for the scene shown in Fig. 5.9. Fig. 5.4(a) shows the pressure after 1.5 seconds of simulation time. In Fig. 5.4(b) high pressure is introduced due to an abrupt merging of 60k particles as utilized by Adams et al. [APKG07] in the context of PCISPH, leading to an unstable simulation in case the number of merge operations is not reduced. In Fig. 5.4(c) a linear temporal blending function is applied which gives a result close to the original pressure field. However, an error estimation in combination with an piecewise linear temporal blending preserves the overall pressure distribution much better, as shown in Fig. 5.4(d).

field with larger errors as shown in Fig. 5.4. Differentials in SPH are quite sensitive to such irregular particle structures and field discontinuities. Solenthaler and Gross [SG11] have utilized an impulse-based transition for high-level boundary particles turning into real particles. However, their method is applicable only if two resolution levels are allowed to coexist, possibly leading to divergence problems. In consistent systems, communication between particles of different smoothing radii increases the error as well [BOT01]. In general, this error can be reduced by avoiding a direct communication between levels as proposed by Keiser et al. [KAG*06] or by using a 1 : 2 replacement structure as has been proposed by Adams et al. [APKG07]. Still, large pressure forces are introduced due to a non-optimized sampling, strong particle overlaps, and small compact smoothing kernels. By applying the CFL condition in each step, such forces dramatically decrease the integration time in order to preserve simulation stability. Even worse, in the context of PCISPH such forces may trigger a shock handling mechanism as described by Ihmsen et al. [IAGT10]. Instead, a temporal blending (Sec. 5.2) smooths out these errors over time by using an error-dependent transition (Sec. 5.4) via a local quantity blending (Sec. 5.3). Thus, the proposed adaptive SPH system allows to use large time steps in combination with small smoothing kernels which are required for a fast and consistent simulations.

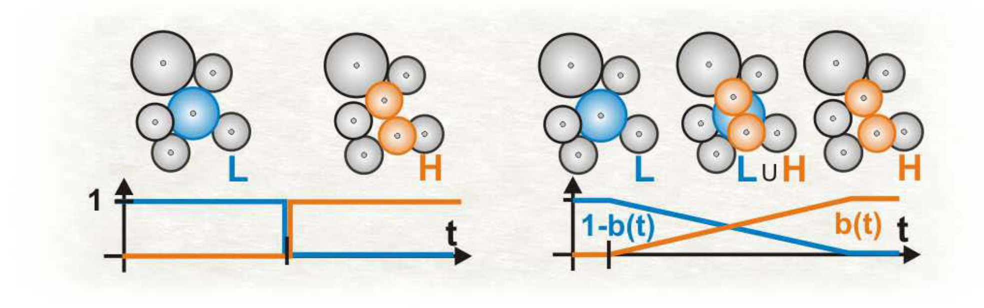


Figure 5.5: Instead of a non-continuous sampling (left) a blending mechanism (right) smoothly replaces old particles (blue) by a new particle set (orange) with respect to a blend-weight $b(t)$.

5.2 Temporal Blending

In order to smooth sampling errors as introduced by sampling operators, the system should smoothly interpolate between two interchangeable fluid representations over time as shown in Fig. 5.5. Even if a blending between multiple representations of an object is a well known concept in computer-graphics, the idea is transferred into the context of SPH-based fluid simulation by starting with the SPH interpolation. With a blending between two global particle sets, the SPH-based summation interpolant changes to

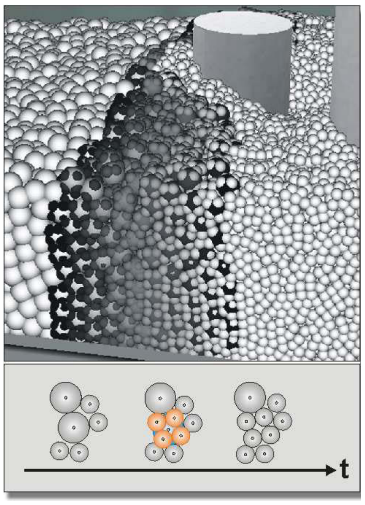
Blending Concept

$$\begin{aligned} Q(\mathbf{x}) =& b \sum_{j\in H} Q_j(\mathbf{x}) + (1-b) \sum_{j\in L} Q_j(\mathbf{x}) \\ =& b\, Q_H(\mathbf{x}) \quad + \quad (1-b)\, Q_L(\mathbf{x}). \end{aligned} \tag{5.4}$$

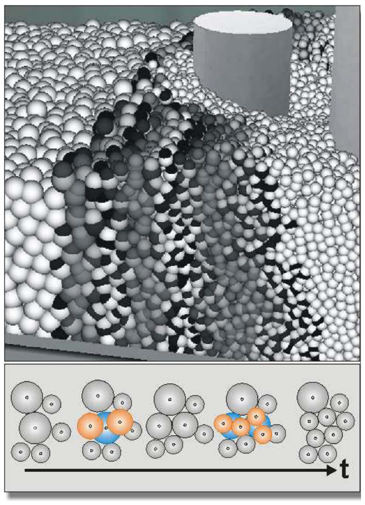
Here, a low-resolution particle set L and a high-resolution particle set H represent two interchangeable *blend-domains*. Both corresponding quantity fields $Q_H(\mathbf{x})$ and $Q_L(\mathbf{x})$ are smoothly blended with respect to a *blend-weight* $b \in [0,1]$. Over time, b increases from zero to one or decreases from one to zero which depends on the required resolution. As a result, particles of one blend-domain smoothly replace the particles in their complementary blend-domain, their so-called *blend-partners*. Instead of just refining one global fluid volume, the system blends between many *local blend-sets* (i.e. sub-volumes of the fluid) simultaneously in order to be of practical use.

Blend-Sets

The concept of blend-sets is required in order to make a local sampling possible. Conceptually, blend-sets are defined as local fluid volumes which are represented by transitioning particles. Each blend-set s therefore consists of two particle sets, a low resolution particle set L_s (blue) and a high



(a)



(b)

Figure 5.6: In 5.6(a) particles are split to 8 child particles in order to double resolution close to the pillars. With blend-sets, new (black) particles are smoothly blended in over time until they fully contribute (white) to neighbouring particles while the contribution from old particles is reduced. In 5.6(b) particles are replaced by two child particles three times in order to double resolution. Due to blending and proper initialization the result is independent from the used pattern.

resolution particle set H_s (orange). The transition between these local particle-sets is controlled by a blend-weight $b_s \in [0, 1]$. Due to multiple co-existing blend-sets, the SPH system needs to handle several individually blending particles together with non blending particles over time as shown in Fig. 5.6. Accordingly, the SPH summation with support for blend-sets is defined by

$$Q(\mathbf{x}) = \sum_{j \notin B} Q_j(\mathbf{x}) + \sum_{s} (b_s \sum_{j \in H_s} Q_j(\mathbf{x}) + (1 - b_s) \sum_{j \in L_s} Q_j(\mathbf{x})), \tag{5.5}$$

where $B = \bigcup_s \{H_s \cup L_s\}$ includes all blending particles. However, similarly to the global blending function as defined by Eq. (5.4), it is advantageous to compute flow quantities separately in both blend-domains of a blend-set. Consequently, Eq. (5.5) has to be rearranged with respect to a single blend-set r. Therefore, the contribution from particles which do not belong to r (grey particles in Fig. 5.7), the neighbouring particles of r, are accumulated by

Blend-Set Boundary

$$\Phi_r(\mathbf{x}) = \sum_{j \notin B} Q_j(\mathbf{x}) + \sum_{s \neq r} (b_s \sum_{j \in H_s} Q_j(\mathbf{x}) + (1 - b_s) \sum_{j \in L_s} Q_j(\mathbf{x})) \, .$$

By adding Φ_r to the contribution from particles in r we can reformulate the SPH summation to

$$\begin{aligned} Q(\mathbf{x}) =& \Phi_r(\mathbf{x}) \;+\; b_r \sum_{j \in H_r} Q_j(\mathbf{x}) + (1-b_r) \sum_{j \in L_r} Q_j(\mathbf{x}) \\ =& b_r \left(\Phi_r(\mathbf{x}) + \sum_{j \in H_r} Q_j(\mathbf{x})\right) \;+\; (1-b_r) \left(\Phi_r(\mathbf{x}) + \sum_{j \in L_r} Q_j(\mathbf{x})\right) \\ =& b_r \, Q_{H_r}(\mathbf{x}) \;+\; (1-b_r) \, Q_{L_r}(\mathbf{x}), \end{aligned} \tag{5.6}$$

where $Q_{H_r}(\mathbf{x})$ and $Q_{L_r}(\mathbf{x})$ represent the quantity fields of the high-resolution blend-domain and low-resolution blend-domain with respect to a single blend-set r. However, Eq. (5.6) cannot be directly applied in practice for reasons described next.

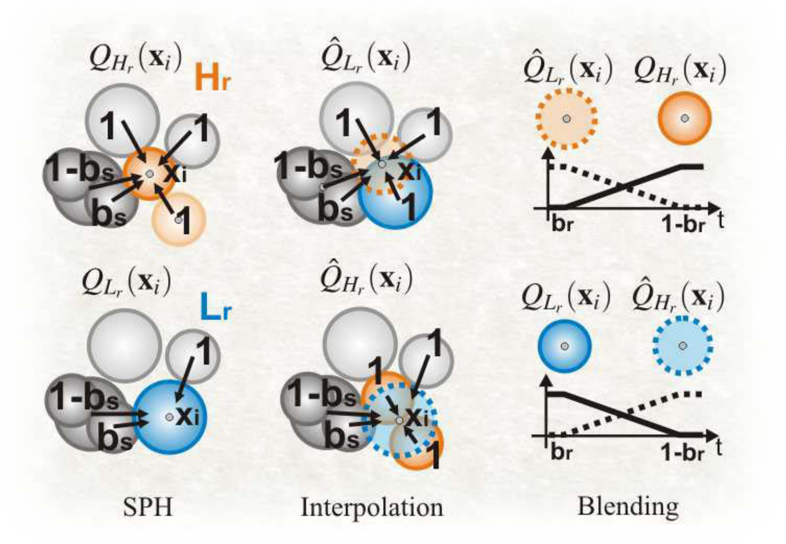

$Q_{H_r}(\mathbf{x}_i)$
Hr
$\hat{Q}_{L_r}(\mathbf{x}_i)$
$\hat{Q}_{L_r}(\mathbf{x}_i)$ $Q_{H_r}(\mathbf{x}_i)$
1
1-bs
bs
xi
br
1-br
t
$Q_{L_r}(\mathbf{x}_i)$
Lr
$\hat{Q}_{H_r}(\mathbf{x}_i)$
$Q_{L_r}(\mathbf{x}_i)$ $\hat{Q}_{H_r}(\mathbf{x}_i)$
SPH
Interpolation
Blending

Figure 5.7: Particles i in a blend-set r blend (right) between a quantity which they evaluate via SPH in their blend-domain (left) and a quantity which they interpolate in their complementary blend-domain (middle), as shown for $i \in H_r$ (top row) and $i \in L_r$ (bottom row). In both steps, neighbouring blend-sets s (dark grey) contribute to the particle with respect to their blend-weights b_s, as indicated by the black arrows.

5.3 Application of Blend-Sets

The separation of resolution levels as described in the last section becomes useful when evaluating flow quantities for blend-sets. However, because blend-partners strongly overlap, a force computation in a particle's complementary blend-domain would lead to strong repulsion forces. That is why in practice, a particle i in a blend-set r utilizes three sequential steps to resemble Eq. (5.6), as shown in Fig. 5.7:

Blending Steps

a) **SPH**: At first, particle i computes its flow quantities via SPH in its blend-domain only, i.e. $Q_i = Q_{H_r}(\mathbf{x}_i)$ for $i \in H_r$ and $Q_i = Q_{L_r}(\mathbf{x}_i)$ for $i \in L_r$. During this step, blend-domains are treated independently (see Sec. 5.3.1).

b) **Interpolation**: Subsequently, particle i interpolates flow quantities in its complementary blend-domain (see Sec. 5.3.2), resulting in $\hat{Q}_i = \hat{Q}_{H_r}(\mathbf{x}_i)$ for $i \in L_r$ or in $\hat{Q}_i = \hat{Q}_{L_r}(\mathbf{x}_i)$ for $i \in H_r$. However, an SPH summation would result in an underestimation of flow fields as the interpolation point $\mathbf{x}_i$ does not coincide with any particle position $\mathbf{x}_j$ in i's complementary blend-domain. Instead, a more consistent interpolation [BK02] is applied using the corrected kernel function as given in Eq. (3.11).

c) **Blending**: Finally, both quantities are blended with respect to the blend-weights b_r in order to synchronize flow dynamics between blend-domains:

$$Q_i \leftarrow \begin{cases} b_r\, Q_i \;+\; (1-b_r)\, \hat{Q}_i & i \in H_r \\ b_r\, \hat{Q}_i \;+\; (1-b_r)\, Q_i & i \in L_r. \end{cases} \tag{5.7}$$

With such a blending of quantities, the system enables a smooth transition between blend-partners over time.

Instead of grouping particles according to their blend-sets it is better to employ pair-wise conditions in order to evaluate flow quantities.

5.3.1 Flux Computation

As a first step, a particle i in blend-set r evaluates flow quantities in its blend-domain. According to Eq. (5.6) neighbouring particles j, which belong to the same blend-domain, contribute to i with respect to their blend-weights. Instead of gathering contributions from all blend-sets separately, conditional pair-wise contributions $b_{i\leftarrow j}$ are introduced into the SPH summation:

Separation of Blend-Domains

$$Q_i = \sum_j b_{i\leftarrow j}\, Q_j\, V_j\, W_{ij}. \tag{5.8}$$

As shown in Fig. 5.7, a contribution from a neighbouring particle j to particle i under consideration is then defined as

$$b_{i\leftarrow j} = \begin{cases} 0 & j \in H_s \wedge i \in L_s \;\vee\; j \in L_s \wedge i \in H_s \\ b_s & j \in H_s \wedge r \neq s \\ 1-b_s & j \in L_s \wedge r \neq s \\ 1 & \text{otherwise.} \end{cases}$$

Please note that pair-wise contributions $b_{i\leftarrow j}$ are also applied during flux computation for particles i which do not belong to any blend-set. For such non-blending particles simply $r \neq s$ holds, since both blend-domains are empty $H_r \cup L_r = \emptyset$. Thus, neighbouring particles fully contribute to the blend-set and itself receive partial contributions from each particle level

Spatial Derivatives

of the blend-set. Note that such blend weights do not change the way spatial derivatives or symmetrized gradient approximations are computed since blend weights are an inherent particle property which can be pre-multiplied with particle volumes. That way they are fully transparent for SPH formulations and at the same time the SPH system avoids instantaneous changes in the number density (see Eq. (3.10)) which otherwise would lead to strong pressure forces. With blending, neighbouring particles smoothly adapt to new particle configurations as new particles become visible. However, particles belonging to a blend-set require an subsequent step in order to exchange information with the particles of their complementary blend-domain. Blend-partners need to synchronize their quantities by utilizing an interpolation in their complementary blend-domain.

5.3.2 Flux Synchronization

In each step of the simulation, divergence between blend-domains is avoided due to synchronization of flow quantities utilizing an cross-interpolation between blend domains. However, standard SPH interpolation does not even preserve constant functions which would result in larger errors when reconstructing quantity fields close to surfaces. In order to interpolate quantities in the complementary blend-domain a constant correction (cf. Sec. 3.1.4) is utilized which minimizes the error as defined by Eq. (5.3) in moving least square sense:

Error Minimization

$$\arg\min_{\hat{Q}_i} \sum_j \hat{b}_{i\leftarrow j} V_j W_{ij} \left(Q_j - \hat{Q}(\mathbf{x}_i)\right)^2,$$

where $\hat{b}_{i\leftarrow j}$ again are pair-wise contributions as examined shortly. By assuming piecewise constant polynomials as an approximating function, i.e. $\hat{Q}(\mathbf{x}_i) = \hat{Q}_i$, one receives the Shepard interpolation [She68] for particle i in blend-set r:

Interpolation

$$\begin{aligned}
0 &= \frac{d}{d\hat{Q}_i}\left[\sum_j \hat{b}_{i\leftarrow j}\, V_j\, W_{ij}(Q_j - \hat{Q}_i)^2\right] \\
0 &= \sum_j \hat{b}_{i\leftarrow j}\, V_j\, W_{ij} \frac{d}{d\hat{Q}_i}\left[\hat{Q}_i^2 - 2\hat{Q}_i Q_j + Q_j^2\right] \\
0 &= 2\hat{Q}_i \sum_j \hat{b}_{i\leftarrow j}\, V_j\, W_{ij} - 2\sum_j \hat{b}_{i\leftarrow j}\, Q_j\, V_j\, W_{ij} \\
\hat{Q}_i &= \frac{\sum_j \hat{b}_{i\leftarrow j}\, Q_j\, V_j\, W_{ij}}{\sum_j \hat{b}_{i\leftarrow j}\, V_j\, W_{ij}}, \qquad (5.9)
\end{aligned}$$

which correctly interpolates constant functions in an efficient manner (See also Lewis et al. [LPA10] for a derivation from moving least squares). As shown in Fig. 5.7, the contributions from neighbouring particles j to particle i under consideration is now defined as

$$\hat{b}_{i \leftarrow j} = \begin{cases} 0 & i,j \in L_s \;\vee\; i,j \in H_s \\ b_s & j \in H_s \;\wedge\; r \neq s \\ 1 - b_s & j \in L_s \;\wedge\; r \neq s \\ 1 & \text{otherwise.} \end{cases}$$

Please note that Eq. (5.9) is used for pure interpolation only, e.g. to interpolate velocities $\hat{\mathbf{u}}_i$ or densities $\hat{\rho}_i$. In practice the interpolation radius is slighty increased in order to get a good trade-off between the smoothing of quantity fields and the divergence between blend-partners $h_{ij} = 1.25\,\frac{h_i+h_j}{2}$. However, in rare cases, blend-partners may still diverge, e.g. due to contact with sharp boundaries. In cases where $x_{ij} > h_{ij}$, an averaging of the velocity values is employed in order to let blend-partners stick together. On the one hand, this effectively smooths sharp features, but on the other hand, an averaging operation avoids non-physically motivated bonding mechanisms.

Divergence

Even if Eq. (5.9) results in a better approximation of quantity fields the Shepard interpolation neither preserve linear nor preserve angular momentum. As the introduced damping of flow dynamics is not noticeable a normalization of the interpolated density and velocity values is not applied. However, the total amount for transported quantities should be exactly preserved. Fortunately, due to isotropic diffusion, the concentration profile is rather homogeneous. As concentrations are given with respect to a particle's volume, one can utilize a nearest neighbour interpolation of concentrations as given by

Conservation

$$\hat{Q}_i = \sum_j Q_j, \tag{5.10}$$

where in this case the contributing particle set j is restricted to blend-partners only, i.e. $j \in H_s \wedge i \in L_s \vee j \in L_s \wedge i \in H_s$. With the proposed blending of quantities, the system is able to smoothly exchange particle sets over time.

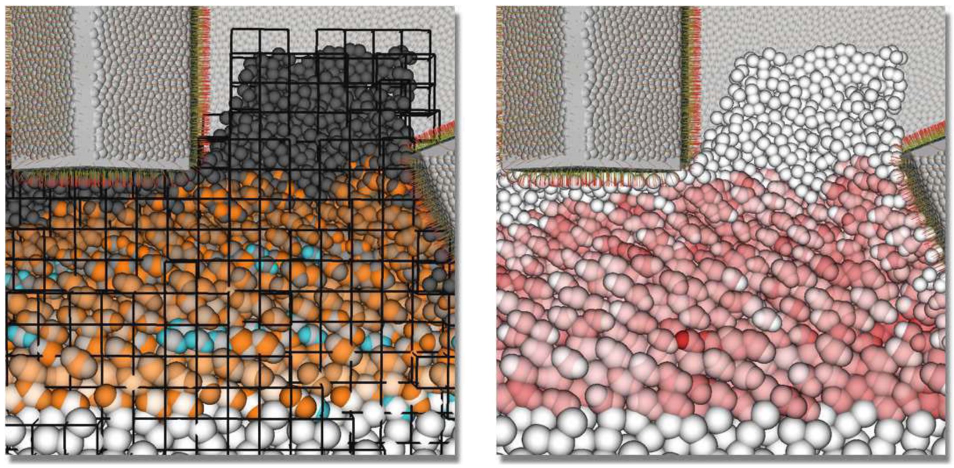


Figure 5.8: Top-view (left) of the "Valley"-scene and the estimated sampling-errors (right). Newly created (blue) particles may introduce large blend-errors (red), due to strong overlaps with neighbouring particles.

5.4 Blending Duration

Blending Time

After a blend-set has been created via a split or a merge operation, it goes through two different phases: in the initialization phase, newly created particles are passively advected while their initial position is improved, as described in Sec.5.4.2. In the subsequent transition phase, blend-sets smoothly update their blend-weights in order to enable a stable transition from one time-step to the next. The influence of new particles increases from zero to one, and simultaneously, decreases from one to zero for old particles as modelled by the following piecewise linear blending function:

$$b_r(t+\Delta t) = b_r(t) + \begin{cases} \Delta b_r(t) & L_r \text{ splitted} \\ -\Delta b_r(t) & H_r \text{ merged}, \end{cases} \tag{5.11}$$

where Δb_r depends on the local sampling error (see Sec.5.4.1). As soon as old particles do not contribute to their neighbouring particles anymore, i.e. $b_r = 1$ in case of a split or $b_r = 0$ in case of a merge, they are removed from the system.

5.4.1 Error Estimation

Blend Error

In order to make the transition as smooth as possible, blend-weights are incremented with respect to local sampling errors. For this purpose, the system measures the error which is introduced into a quantity field by updating all blend-weights by a global blend-increment $\Delta b \in]0,1]$, which according to Eq. (5.3) and Eq. (5.5) yields

$$\begin{aligned} E_Q(\mathbf{x}) &= \mid Q(\mathbf{x}) - Q^*(\mathbf{x}) \mid \\ &= \mid \sum_s (b_s \sum_{j \in H_s} Q_j(\mathbf{x}) + (1-b_s) \sum_{j \in L_s} Q_j(\mathbf{x})) \\ &\quad - \sum_s ((b_s + \Delta b) \sum_{j \in H_s} Q_j(\mathbf{x}) + (1-(b_s+\Delta b)) \sum_{j \in L_s} Q_j(\mathbf{x})) \mid \\ &= \mid \sum_s (\Delta b \sum_{j \in H_s} Q_j(\mathbf{x}) - \Delta b \sum_{j \in L_s} Q_j(\mathbf{x})) \mid \end{aligned}$$

where $Q^*(\mathbf{x})$ is the estimated quantity field when incrementing all blend-weigths at once. By assuming a static particle neighbourhood it is then possible to measure the sensitivity of a quantity field with respect to a change of blend-weights. Even if one can compute sampling errors for all quantity fields, for convection dominated flows in general it is only necessary to measure the error which is introduced into the density field, which is given by (cf. Eq. (3.10)) **Density Error**

$$E_\eta(\mathbf{x}) = \mid \sum_s (\Delta b \sum_{j \in H_s} v_j W_j(\mathbf{x}) - \Delta b \sum_{j \in L_s} v_j W_j(\mathbf{x}) \mid, \tag{5.12}$$

which in the examples is most important to ensure stability. Instead of iteratively adjusting Δb to stay below $E_{\eta,\max}$, a single estimation step was used by assuming a linear dependency between blend-weights and sampling errors. First, all particles estimate the number density difference $E_{\eta,j}$ (see Fig.5.8) setting $\Delta b = \Delta b_{\max} = \Delta t / \Delta t_{\min}$, where the minimum blending time $\Delta t_{\min}$ is defined by the user. Second, blend-sets compute weight increments by using the maximum of all individual errors: **Blend-Step Size**

$$\Delta b_r = \Delta b_{\max} - (\Delta b_{\max} - \Delta b_{\min}) \max_j \frac{E_{\eta,j}}{E_{\eta,\max}}, \tag{5.13}$$

where $\Delta b_{\min} = \Delta t / \Delta t_{\max}$ and the maximum user defined density error $E_{\eta,\max} = 0.06$. Please note that particles j include all non-blending neighbouring particles of blend-set r as well. $\Delta t_{\max}$ and $\Delta t_{\min}$ allow users to steer the blending either towards performance or towards accuracy. For all the presented examples $\Delta t_{\min} = 40$ms and $\Delta t_{\max} = 200$ms has resulted in smooth transitions.

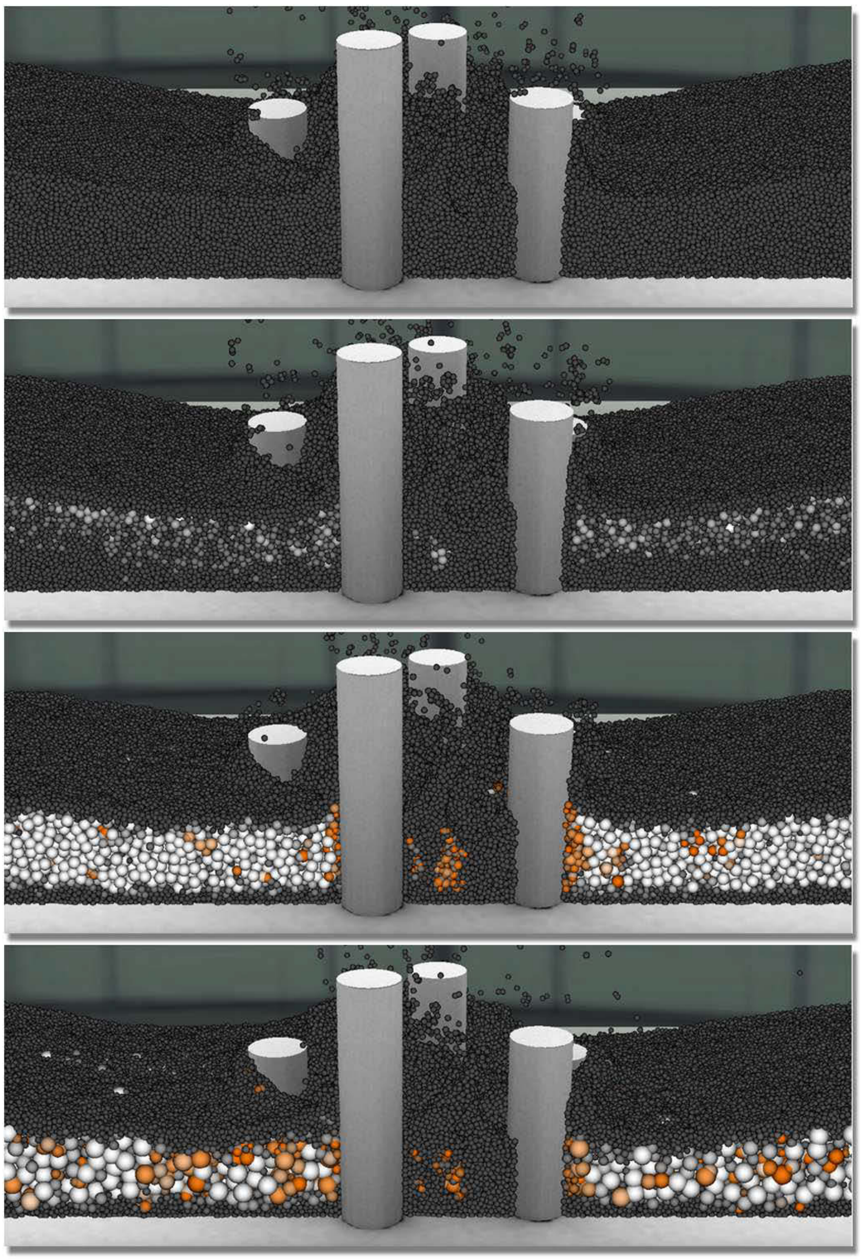


Figure 5.9: PCISPH simulation of two colliding fluid fronts (top row). Only 160 k of the 960 k particles can be stably merged with [APKG07] (second row). In contrast, a temporal blending (third row) achieves an 1.6x speed-up by halving the particle count. Decreasing the particle resolution further to level $l_{\max} = 6$ (last row) however introduces sampling errors.

5.4.2 Particle Initialization

The blending duration can be greatly reduced if particles are initialized properly. As shown in Fig.5.8, newly created particles may introduce larger errors than $E_{\eta,\max}$ due to excessive overlaps with neighbouring particles or due to a collision with rigid objects. Accordingly, initial positions of new particles are improved over the time of a few simulation steps by using their pressure force and keeping $\Delta b_r = 0$:

Corrective Impulses

$$\mathbf{x}_i \leftarrow \mathbf{x}_i + \alpha \frac{1}{m_i} \mathbf{F}_i^p, \tag{5.14}$$

where α is a user defined parameter. Passive particles are restricted not to move more than h_i away from the mass center of their (active) blend-partners. Please note that such impulses can only improve positions to a certain extent. A merge or a split is therefore postponed if the error is still too high after relaxation.

5.5 Implementation Details

Alg. 6 highlights the modifications to PCISPH in order to introduce blend-sets (orange). Transitioning particles have to interpolate and blend quantities after each SPH update as shown in Fig.5.7. In each simulation step, particles of a blend-set synchronize their density, velocity and concentrations as described in Sec. 5.3. Please note that forces are computed separately for each blend-domain and are not synchronized between blend-partners. Instead, the convective flux is synchronized by blending particle velocities. At the end of each simulation step, blend-sets then update their blend-weights by predicting a sampling error as described in Sec. 5.4. Note that according to the measurements, it is sufficient to apply a synchronization of particle densities within the correction loop. Since non-uniform particle sizes are used, multiple constants $\beta_l, \;\; l = 0, 1, .., l_{\max}$ need to be computed. β_l dependent on particle levels are required to in order to account for the particle size when correcting particle pressures. Each is pre-computed independently for a prototype particle of level l with a filled neighbourhood of level-l particles and is updated each time the integration step-size changes. During iteration, each particle then chooses its constant β_l corresponding to its level l_i. Neighbours lists as described in Sec. 3.3.3 are set up using the averaged support radius, i.e. particles are neighbours if $x_{ij} < \frac{1}{2}(h_i + h_j)$, which results in a symmetric visibility during flux computations.

Synchronization Kernels

Blended Quantities

Non-Uniform Particle Sizes

Neighbour Lists

Algorithm 6: Data-parallel PCISPH with quantity blending (orange). Without blending the convection depicts the original PCISPH [SP09b].

```
        Diffusive Flux

foreach particle $i$ in parallel do
    update concentration $Q_i(t+\Delta t)$ (Eq. (3.19))
foreach blending particle $i$ in parallel do
    interpolate concentration $\hat{Q}_i(t+\Delta t)$ (Eq. (5.10))
    blend concentration $Q_i(t+\Delta t)$ (Eq. (5.7))
        Convective Flux

foreach particle $i$ in parallel do
    compute non-pressure forces $\mathbf{F}_i^{\mu+g+\alpha}(t)$ (Eq. (3.23)-(3.24))
    initialize pressure $p_i(t) = 0$
    initialize pressure force $\mathbf{F}_i^p(t) = \mathbf{0}$
while $\eta^*_{err} < \epsilon$ do
    foreach particle $i$ in parallel do
        predict velocity $\mathbf{u}_i^*(t+\Delta t)$ (Eq. (3.25))
        predict position $\mathbf{x}_i^*(t+\Delta t)$
    foreach particle $i$ in parallel do
        predict density $\eta_i^*(t+\Delta t)$ (Eq. (3.10))
    foreach blending particle $i$ in parallel do
        interpolate density $\hat{\eta}_i^*(t+\Delta t)$ (Eq. (5.9))
        blend density $\eta_i^*(t+\Delta t)$ (Eq. (5.7))
    foreach particle $i$ in parallel do
        compute pressure force $\mathbf{F}_i^p(t)$ (Eq. (3.22))
        correct pressure $p_i(t) \mathrel{+}= \beta_l(\eta_i^*(t+\Delta t) - 1)$
foreach particle $i$ in parallel do
    update velocity $\mathbf{u}_i(t+\Delta t)$
foreach blending particle $i$ in parallel do
    interpolate velocity $\hat{\mathbf{u}}_i(t+\Delta t)$ (Eq. (5.9))
    blend velocity $\mathbf{u}_i(t+\Delta t)$ (Eq. (5.7))
```

5.6 Results and Discussion

Scenarios

All presented scenes had been tested on an Intel Dual-Core 2.66 GHz with a NVidia GTX 580 Graphics Card with 1.5 GB VRAM. In order to demonstrate the applicability the temporal blending method is compared with the

Scene	**"Valley"**	**"Pillars"**	**"Teapot"**
Sim. Time [s]	75	21	35
Avg.Δt [ms]	2	2.5	2
Min #ptcls [k]	0-500 / 0-1000 0-270*	210,500,700 / 500,1000,1500 380*,820*,1100*	0-470 / 0-1000 0-700*
Comp. Time [min]	**34 / 57** **32.3***	**5.9,13.7,18.1 /** **7.6,17.3,26.7** **9.9*,23.9*,31.3***	**15.8 / 19.8** **18.68***
Snapshot	**Fig.5.1 at 30s**	**Fig.5.9 at 5s**	**Fig.5.3 at 4.5s**
#ptcls [k]	310[20]/ 635	500[40]/ 960	100[40]/ 128
Neighbours [ms]	10.1[0.7]/ 18.2	14.6[1.4]/ 26.5	5.2[2]/ 4.1
Diff. Flux [ms]	6.9[0.3]/ 12.4	9.67[1.1]/ 16.9	1.7[0.6]/ 2.1
Conv. Flux [ms]	27.1[3.3]/ 47.7	41.1[3.8]/ 65.7	11[2.5]/ 9.5
Blend. Trans. [ms]	5.1[5.1]/ 0	9[9]/ 0	1.7[1.7]/ 0
Time Int. [ms]	1.1[0]/ 2.1	1.9[0]/ 4	1.9[0]/ 0
Split/Merge [ms]	0.7[0]/ 0	1.7[0]/ 0	0.2[0]/ 0
Total [ms]	**51[9.4]/ 80.4**	**78.1[15.3]/ 113.1**	**20.5[6.8]/ 16.6**

Table 5.1: Timings of the adaptive / non-adaptive PCISPH. The overhead due to blend-sets is highlighted in orange. Overall timings for [APKG07] are marked with *.

techniques from Solenthaler and Pajarola [SP09b], Müller et al. [MCG03] and Adams et al. [APKG07] with particle numbers varying from 500 k particles to 1,5 M particles. Table 5.1 gives an overview over the simulation times for all presented scenes and shows timings for the operations as presented in Alg.6. Results are visualized by using an interactive volume ray-casting as described in Sec. 6.

Computation Speed

In the "Village" scene, Fig.5.1, the particle count increases to one million particles over time in case of a non-adaptive simulation and 500 k in case of a comparable adaptive SPH within a fixed predefined high-resolution region around the village. As shown in Fig. 5.10, the workload scales linearly with the number of particles in the predictive-corrective loop. As expected, an adaptive method outperforms the PCISPH [SP09b] method by a factor of 1.6 and speeds-up a compressible SPH simulation [MCG03], by a factor of 1.4 when the particle count is halved. As shown in the Teapot-example (see Fig.5.3), in very complex flow scenarios, the number of blend-sets might be very high compared to the overall particle count which results in no speed-up at all. However, this will be the case for most adaptive simulations due to fast changing resolution regions (which is recomputed every 20 frames). Still, resolution regions are in good agreement with flow dynamics.

Stability

By inserting particles abruptly [APKG07], high artificial pressure forces

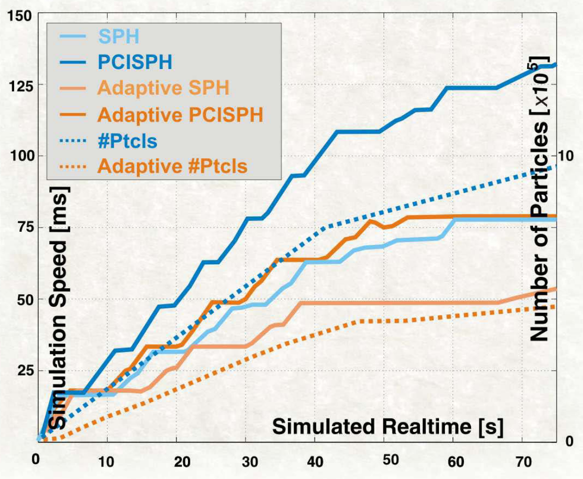


Figure 5.10: The temporal blending speeds-up the simulation given in Fig.5.1 by a factor 1.6 for PCISPH [SP09b] and by a factor of 1.3 for SPH [MCG03].

are introduced, as visualized in Fig.5.4. In general, very few particles can be stably merged, as shown in Fig. 5.9. Even with a reduced number of refinement operations, occasional high pressure forces occure and lead to a flickering of the integration time-step, as shown in Fig. 5.11. The system is not able to stably merge particles up to level $l = 6$. In contrast, with temporal blending the method preserves the integration time-step while the number of particles can be halved, even for $l_{\max} = 6$ as shown in the last row of Fig.5.9.

Limitations

However, non-uniform particle radii in combination with kernel averages [Mon92] is only stable for mass differences up to a factor of 10 [SP08] and non-uniform smoothing radii introduce larger errors if they differ by more than a factor 2 between particle neighbours [BOT01]. Thus, in case of $l_{\max} = 6$, larger particles introduce sampling errors and damp flow dynamics notably and the SPH system cannot gain much speed-up, since the neighbour search then becomes a new bottleneck.

Future Work

In the future, it seems promising to combine a blending of quantities with virtual boundary particles as proposed by Keiser et al. [KAG*06]. However, the presented temporal blending technique is independent from the smoothing kernels and refinement patterns and may even stabilize related

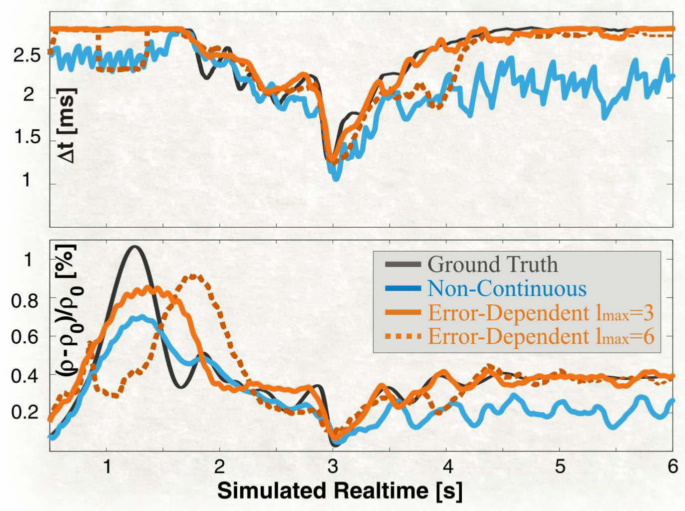


Figure 5.11: Time-step size and average density (in percentage of the rest density) for Fig. 5.9. Even if the density profile is preserved a stable non-continuous replacement (blue) results in a flickering of the integration time-step (top) due to occasional high pressure forces. In contrast, the proposed sampling still preserves the integration time-step, even for $l_{\max} = 6$.

sampling mechanisms which insert or remove particles at the fluid boundary [LAD08, SB12].

5.7 Conclusions

This chapter has presented a novel temporal blending approach which is capable of exchanging particle sets while maintaining a consistent convection-diffusion simulation by using standard SPH rules only. The temporal blending mechanism significantly reduces the influence of sampling errors while preserving the integration time-step. Additionally, a scheme to control the blending time according to a predicted error in the pressure term has been proposed.As such, it is easily applicable to a larger range of SPH applications not only including performance-related scenarios. In order to evaluate the flexibility of the system, it has been integrated into the latest approaches presented in the field of SPH-based fluid simulations. In combination with the new blending approach, the fully GPU-based implementation achieves interactive frame-rates for up to a million of particles.

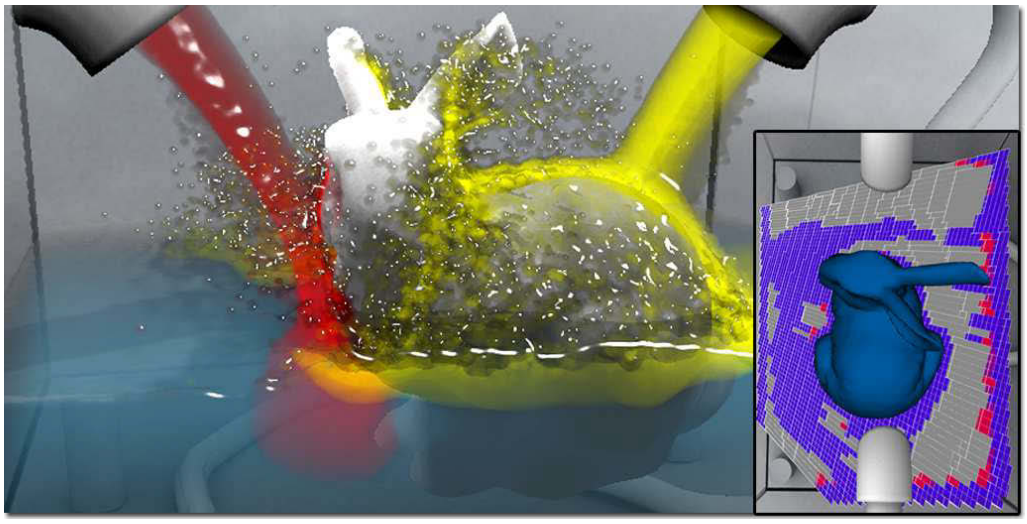


Figure 6.1: Combined surface and volume rendering which uses a feature preserving sampling in a sparse perspective particle access structure (right).

Chapter 6

Feature-Preserving Ray Casting

*This chapter proposes a combined surface and volume ray casting technique of SPH data for rendering bulk dynamics (cf. Chap. 3) and surface dynamics (cf. Chap. 4). The underlying sparse data-structure is especially designed to enable an instant, cache-coherent particle access as well as an adaptive sampling of SPH-fields. While the underlying caching mechanisms are based on [OKK10], the presented sampling is currently unpublished. The reader is referred to the literature for a general introduction to volume rendering [EHK*06, HLSR08] and point-based rendering [KB04].*

Lagrangian simulations like SPH pose high spatial flexibility which is advantageous for convection-driven flow effects, free surfaces, and dynamic fluid-object interactions. However, rendering of unstructured particle data sets is a challenging task, especially for convection-diffusion scenarios.

Several techniques exist to reconstruct smooth surfaces [SSP07, vdLGS09, OCD11, AIAT12, YT13]. However, sole surface rendering can only convey the fluid's geometric shape, but does not provide any information about the internal fluid structures. Thus, volume rendering techniques [EHK*06, HLSR08] have to be combined with surface reconstruction techniques in order to render the full effect. In the context of unstructured particle sets, mainly hybrid splatting-slicing approaches have been proposed so far [FGE10, FAW10]. Particle contributions [Wes90] are scattered (cf. Sec. 3.3.3) onto view-aligned [NMM*06, FGE10, FAW10] or axis-aligned texture slices [SP09a]. These slices are then composited front-to-back to yield the final image. Most importantly, texture-slicing is limited to the rasterization pipeline, thus making it very hard to incorporate advanced rendering techniques as required to render smooth surfaces or to apply adaptive sampling mechanisms. In general, ray casting is the more generic approach to volume rendering [EHK*06]. In this chapter a fast, high quality ray casting for unstructured particle data sets is proposed. It combines volume and surface raycasting and uses an adaptive sampling allowing for efficient rendering of large data sets as shown in Fig. 6.1. In detail, the proposed rendering approach incorporates the following contributions:

Surface vs. Volume Rendering

Splatting vs. Raycasting

Goals

Detailed Contributions

– Inspired by the idea of a perspective grid used for sweeping slicing planes in texture-based approaches [FAW10], a sparse and view-aligned access data structure has been developed which removes rasterization restrictions and leads to very little memory overhead.
– The proposed rendering does not rely on any pre-computations, thus, large dynamic particle data sets can be visualized without any latency. It comprises a built-in empty space skipping and completely releases the ray casting algorithm from any traversal logic.
– An on-the-fly sampling error analysis is applied for each view-aligned grid cell, revealing strict screen space error bounds. Computing and applying these bounds is achieved using a greedy algorithm which practically results in a very reliable error control.
– For each cell the local sampling rate of the volume rendering equation is adapted in order to dynamically shift computational resources to salient regions of the fluid volume.

Overview

The remainder of this chapter is structured as follows: Sec. 6.1 discusses the foundations in surface rendering and volume raycasting. Sec. 6.2 gives an overview of our ray casting pipeline and Sec. 6.3 describes the perspective access structure. Sec. 6.4 explains the sampling error analysis applied to determine proper sampling rates. Implementation details for the resulting particle ray casting algorithm are given in Sec. 6.5. Sec. 6.6 then discusses the results before final conclusions are drawn in Sec. 6.7.

6.1 Foundations and Challenges

In this section we will describe existing surface rendering approaches and investigate challenges in combination with volume rendering techniques.

6.1.1 Surface Reconstruction

Even though particles accurately capture surfaces, in general some work is required to reconstruct smooth surfaces. Scalar field function as described in Sec. 4.1 are in general preferred. Implicit functions require an extra smoothing pass which is either achieved by redistancing of surface particles [APKG07], by distance [SSP07] or density [OCD11] decay functions, by computing anisotropic smoothing kernels [YT13] or by minimizing the thin plate energy via constraint optimization [BGB11]. Polygonalization of the resulting iso surfaces [MCG03, SSP07, APKG07, AIAT12, AAIT12, AAOT13, YT13] using the marching cubes algorithm [LC87] then yields controllable smooth surfaces, but usually is restricted to offline applications. For an instant rendering either screen space techniques [ZPvBG01, ALD06, MSD07, vdLGS09, Gre10, MM13] or direct rendering of the iso-surface [GSSP10, FAW10, OCD11] are employed. Anisotropic kernels [YT13] are mandatory for image-based surface reconstruction [vdLGS09], which otherwise lead to smoothing artefacts when viewed from flat angles (cf. Fig. 4.7). Direct ray casting of the implicit surface [OCD11], which samples $\nabla\phi(\mathbf{x})$ (cf. Eq. (4.1)), can be combined with the proposed particle access structure and produces smooth surfaces independent from view points (cf. Fig. 6.5).

Smoothing

Marching Cubes

Image Space Techniques

Direct Rendering

6.1.2 Volume Rendering

The idea of volume ray casting is to evaluate a physically-based model for light transport by treating quantity fields as a participating medium. Conceptually, for each viewing ray, emission-adsorption values are integrated from the camera towards the fluid volume. Considering ray samples at integer coordinates $i = 0, \dots, N-1$ with associated constant-corrected quantities $\hat{Q}_0, \dots, \hat{Q}_{N-1}$, the discretized version of the volume rendering integral defines a ray's composited intensity as [EHK*06]

Volume Rendering Integral

$$I = \sum_{i=0}^{N-1} I_i \prod_{j=0}^{i-1} T_j^{\Delta s}, \tag{6.1}$$

where irradiance values $I_i = \tau_I(\hat{Q}_i)$ and transparency values $T_i = \tau_T(\hat{Q}_i)$ are defined via material-dependent transfer functions $\tau \in [0, 1]$. Discrete samples $\hat{Q}_i = \hat{Q}(\mathbf{x}(s_i))$ are distributed at distances $s_i \in [s_{\text{near}}, s_{\text{far}}]$ along viewing rays, where s_{near} and s_{far} are the distances to the near and far clipping planes of the viewing frustum. Note, transparency values given for a unit reference length are corrected to match the actual sampling step size Δs. To increase readability, opacity correction terms will be omitted in the following, but in practice need to be applied appropriately.

Opacity Correction

Challenge: Adaptive Sampling A uniform sampling ($\Delta s = const$) results in a large number of sampling operations. Sampling operators [APKG07, ZSP08, HHK08] as dicussed in Sec. 5.1.3 approximate particle subsets by fewer larger particles which is an effective technique for compressible SPH-data [HE03, FSW09, FAW10]. However, up-sampling may introduce visual artifacts [BOT01, KAG*06] and should be applied only to avoid under-sampling in case particle sizes fall below the pixel size, i.e. $\Delta s > h$. A different strategy is to adapt the number of samples in viewing direction to the image plane resolution. The following perspective transformation $\mathbf{C}^{-1} : \mathbb{N}^3 \to \mathbb{R}^3$ maps samples from uniform sampling space to non-uniform view space [FAW10]:

Sampling Operators

Logarithmic Distribution

$$s_i = s_{\text{near}} \left(\frac{s_{\text{far}}}{s_{\text{near}}} \right)^{\frac{i}{N}}, \tag{6.2}$$

Still, in many regions, the contribution of individual samples to the rendering integral is relatively small. Thus, the idea is to take non-uniform sampling of Eq. (6.2) one step further by locally adapting sampling rates to the saliency of the data, i.e. varying the sampling step size with respect to the underlying quantity field. Even though, adaptive sampling is known for grid-based data [LGM*08, BHMF08, KHW*09], it has not been applied for volume ray casting of SPH data yet.

Adaptive Sampling

Challenge: Memory Coherence Memory coherence is usually enforced by space subdivision schemes in object-space, i.e. the simulation domain is subdivided using kd-trees [ZHWG08, LLRR08, JFSP10] or octrees [GGG08, ZGHG10, FAW10] that store particle references as a means to fast particle lookup. However, object-aligned access structures require cell finding logic. Divergent traversal decisions among adjacent rays introduce high thread divergence which drastically reduces parallelism [PGSS07]. In contrast, our proposed access structure employs view-aligned cells which facilitates memory coherence and frees the ray casting algorithm from any cell finding logic. Due to it's simplicity, the proposed rendering system is fully parallelizable. It strictly uses building blocks known from particle search algorithms simulation [SHG09, SHZO07] (cf. Sec. 3.3.3).

Access Structures

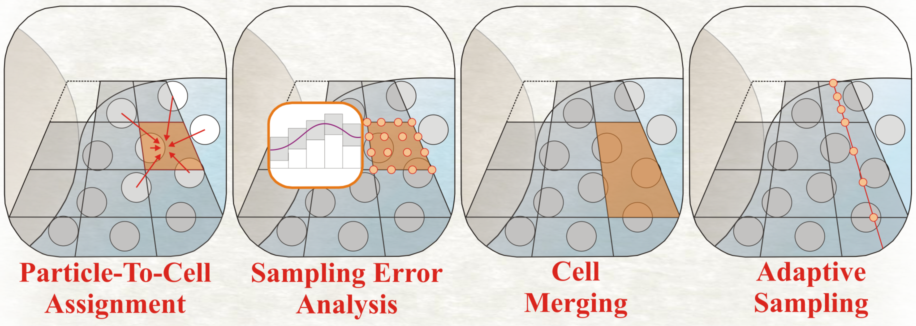


Figure 6.2: Building blocks of the proposed ray casting pipeline. After particles are assigned to view-aligned cells, an analysis of the sampling error in screen-space triggers a merging of cells, which ensures a constant number of samples per cell during ray casting.

6.2 Ray Casting Pipeline

As depicted in Fig. 6.2, the proposed rendering pipeline comprises four components incorporating *coherent particle access* and *adaptive samples*:

Particle Assignment: First particles are assigned to the cells of a perspective access structure covering the view frustum (see Sec. 6.3.1).

Sampling Analysis: Based on the volume rendering integral, in a second step, irradiance and transparency bounds are computed for different *sampling levels* of each cell and for the entire cell sequence associated to a ray bundle. The resulting screen-space error bounds are used to compute per-cell sampling levels (see Sec. 6.4).

Cell Merging: Consecutive cells in view direction which support higher sampling levels are merged. The merging results in a constant number of samples per cell (see Sec. 6.3.2).

Adaptive Sampling: The final volume ray casting algorithm simplifies to a front-to-back traversal of cells. All rays passing through cells sample exactly the same subset of particles, efficiently combining adaptive sampling step sizes with a memory coherent particle access (see Sec. 6.5).

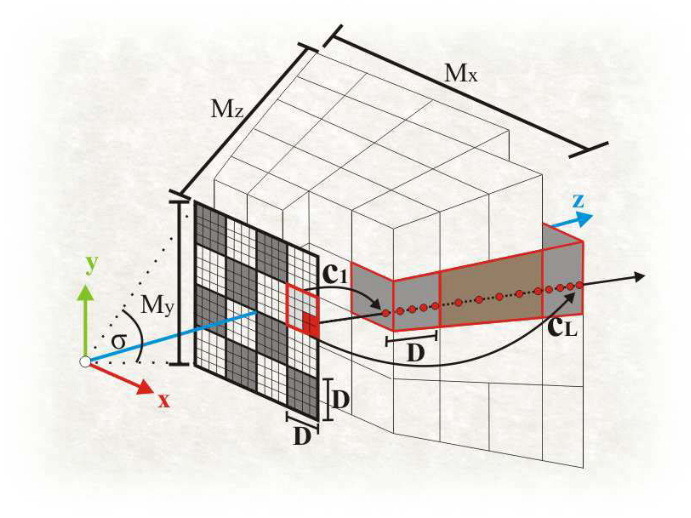


Figure 6.3: The sparse access structure. Cell merging ensures a constant number of $D \times D \times D$ samples per cell. The image plane is split into ray bundles of size $D \times D$ each storing cells c_1 and c_L in a look-up table.

6.3 Coherent Particle Access

Perspective Access Structure

Fast query operations yielding the minimum set of potentially relevant particles is crucial for computational efficiency. The core of the algorithm is a perspective particle access structure as shown in Fig. 6.3. Compared to object-space data structures which usually cover the whole simulation domain, here particles are assigned to cells of a perspective access structure covering the view frustum. Therefore, the view frustum is subdivided into $M_x \times M_y \times M_z$ (virtual) cells, which are fully aligned with viewing rays and which span a fixed number of $D \times D$ pixels of the image plane and D samples in view direction. All $D \times D$ rays access the same cells, from cell c_1 to cell c_L.

Cell-based Rendering Integral

The perspective access structure effectively splits the rendering integral given in Eq. (6.1) into cells:

$$I = \sum_{c=1}^{L} \underbrace{\sum_{k=(c-1)D}^{cD-1} I_k \prod_{j=(c-1)D}^{k-1} T_j}_{I_c} \prod_{d=1}^{c-1} \underbrace{\prod_{k=(d-1)D}^{dD-1} T_k}_{T_d}, \qquad (6.3)$$

where I_c, T_d are the irradiance of cell c and the transparency of cell d, respectively. Due to construction (cf.Sec. 6.3.1), each cell contains exactly the particles to composite irradiance I_c and transparency T_d. Depending on the outcome of the error analysis, cells merge in viewing direction to support larger step sizes while maintaining a constant number of D^3 samples per cell (cf.Sec. 6.3.2).

6.3.1 Particle-to-Cell Assignment

In order to speed-up reconstruction of quantity fields, each cell c stores references to particles $[j_c, j_c + N_c[$ which overlap the cell's space $\Omega_c := \{\mathbf{x} \mid C(\mathbf{x}) = c\}$. The corresponding indexing function $C : \mathbb{R}^3 \to \mathbb{N}_0$ which subdivides the view space into view-aligned cells is defined as:

Indexing Function

$$C(\mathbf{x}) = (C_x(\mathbf{x})\, M_y + C_y(\mathbf{x}))\, M_z + C_z(\mathbf{x}), \tag{6.4}$$

for which cell coordinates $(C_x(\mathbf{x}), C_y(\mathbf{x}), C_z(\mathbf{x}))$ at position $\mathbf{x} = (x, y, z)^T$ in view space are given as

$$\left(\left\lfloor \frac{x}{\varsigma\sigma(z)} + \frac{M_x}{2} \right\rfloor, \ \left\lfloor \frac{y}{\sigma(z)} + \frac{M_y}{2} \right\rfloor, \ \left\lfloor M_z \left(\frac{\ln\left(\frac{z}{s_{\text{near}}}\right)}{\ln\left(\frac{s_{\text{far}}}{s_{\text{near}}}\right)} \right) \right\rfloor \right).$$

Here, C_z is derived using the inverse of Eq. (6.2). C_x and C_y are defined by the aspect ratio ς which relates the horizontal to the vertical resolution of the image plane. The height of a cell $\sigma(z) = z\,\frac{2}{M_y} \cdot \tan\left(\frac{\alpha}{2}\right)$ depends on the distance z to the image plane, and the current field of view α of the view frustum.

Grid Construction

In order to build the access structure, each particle j computes its cell footprint, i.e. an index set $P_j := \{c_i \mid \Omega_c \cap \operatorname{supp}(W_j) \neq \emptyset\}$ containing cells c_i which overlap with particle j. An sorting of particle-to-cell pairs $p = (j, c)$ then yields the perspective access structure. The advantages of this sorting is two-fold: Firstly, particles of a cell are also neighbours in memory which increases cache coherence. Secondly, with geometric and memory proximity, merging of neighbouring cells simplifies dramatically.

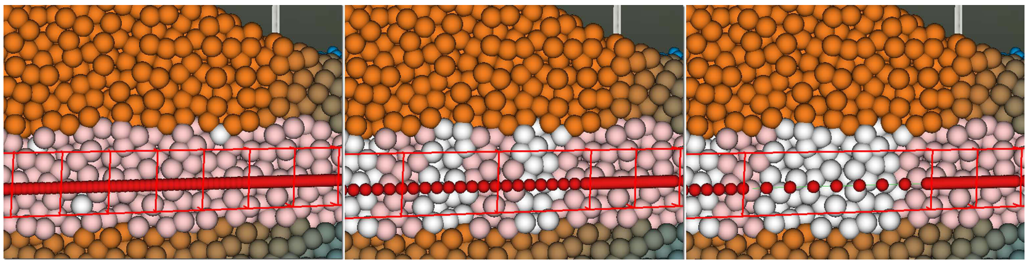


Figure 6.4: The sampling distance (red) doubles due to merging of neighboring cells. Redundant particle references (pink) are removed during merging.

6.3.2 Cell Merging

Adaptive Sampling

In a subsequent step, sample positions $\Delta s_i = s_{i+\Delta i_c} - s_i$ in view space are adapted in order to account for homogeneous regions in the fluid's bulk. Therefore, in sampling space, integer step sizes $\Delta i_c = 2^{l_c^{opt}}$ are computed based on per cell sampling levels l_c^{opt} which are the outcome of the proposed error analysis (cf. Sec. 6.4). As shown in Fig. 6.4, in order to support adaptive samples, neighbouring cells merge bottom-up from level $l = 0$ to the next higher level until either the maximum possible sampling level $l^{\max} = \log_2 D$ or the requested optimal sampling level $l_c^{opt} \in \{0, l^{\max}\}$ is reached.

Bottom-up merging

However, redundant particle references (pink) in the merged cell must be removed in order not to let those particles contribute twice during field reconstruction. For this reason, the following characteristic function is employed to indicate redundant assignments for subsequent cells, i.e. redundant particle-to-cell pairs p:

Redundant Particles

$$\chi(p) = \begin{cases} 1 & c_i \in P_j \wedge (c_{i+1}) \in P_j \\ 0 & \text{otherwise.} \end{cases}$$

During merging, each particle-to-cell pair p of cell c_i is removed if $\chi(p) = 1$.

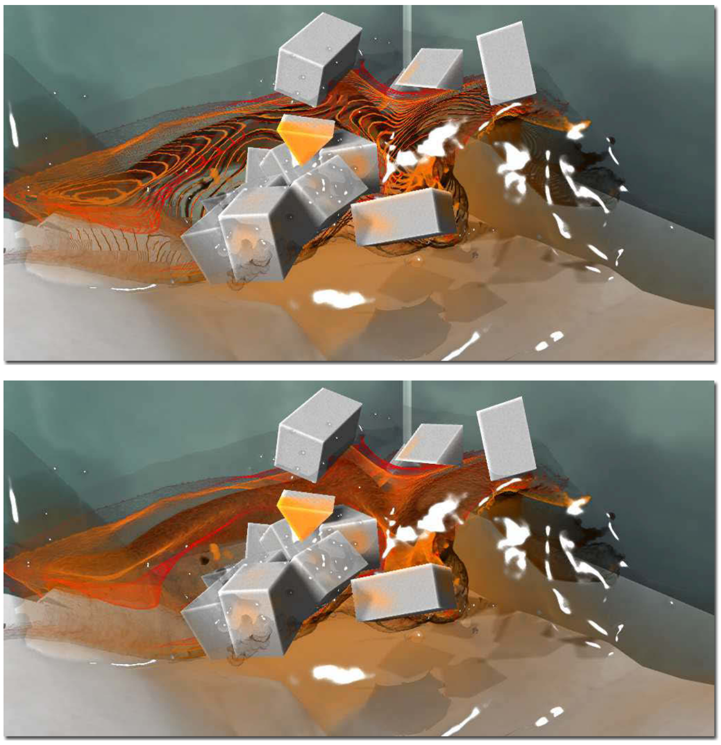


Figure 6.5: High-frequent transfer functions demonstrate the effectiveness of the proposed adaptive sampling. Compared to naive sampling (top), a screen space analysis reveals local extrema in quantity fields (bottom).

6.4 Sampling Error Analysis

Before merging, cells have to determine an appropriate sampling level. As shown in Fig. 6.5, care must be taken to choose appropriate sampling levels for which one has to estimate the screen space error which results from coarser sampling. However, measuring the irradiance error when changing sampling rates would require the same number of samples as the actual ray casting.

Cell Sequences

In order to reduce the number of samples, the image plane is subdivided into ray bundles consisting of $D \times D$ rays (cf. Fig. 6.3). The core idea is to approximate the $D \times D$ samples in lateral direction of the ray bundle by their irradiance bounds and then to composite these bounds in viewing direction, instead of the actual irradiance as given by Eq. (6.3). On a top level view, each ray bundle analyses its cell sequence from cell c_1 to cell c_L in view direction. The analysis then leads to individual levels l_c^{opt} for each cell

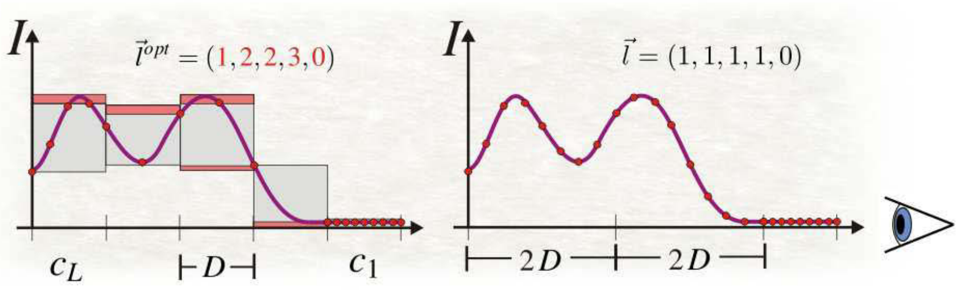


Figure 6.6: Optimal sampling levels (left) and merged cells (right) for an unknown signal (purple). Reducing sampling rates changes the irradiance bounds (grey boxes), which introduces an error (red area). The greedy approach decreases sampling levels until an user-defined error margin is reached resulting in optimal sampling levels $\vec{l}^{opt}$. Note that the final sampling levels $\vec{l}$ do not necessarily correspond to the optimal sampling levels due to subsequent merging of cells.

c in this sequence, which are then fed into the cell merging step as shown in Fig. 6.6.

Greedy Approach

Cell levels have to be selected in such a way that a given screen-space error E_I is not exceeded. Ideally, we would identify the optimal sampling levels $\vec{l}^{opt} = (l_1^{opt}, \ldots, l_L^{opt})$ by checking all level combinations, and then taking the one combination that provides the largest sampling levels and that is below the error margin E_I. However, due to performance considerations, this technique is impractical. Instead, the following greedy approach is applied to identify optimal sampling levels:

Per-Cell Bounds

Cell Irradiance Boundaries: Initially, irradiance and transparency bounds are determined separately for each cell c and level l in this sequence. Therefore, in each cell, irradiance and transparency samples $I_k^{+/-,l}, T_k^{+/-,l}$ are determined in lateral direction as described in Sec. 6.4.1. These irradiance and transparency bounds are then composited per cell (cf. Eq. (6.3)):

$$I_c^{+/-,l} = \sum_{k=0}^{D-1} I_k^{+/-,l} \prod_{j=0}^{k-1} T_j^{+/-,l}, \qquad T_c^{+/-,l} = \prod_{i=0}^{D-1} T_i^{+/-,l}. \qquad (6.5)$$

Irradiance I_c and transparency T_c in cell c are thus bounded by $I_c^{-,l} \leq I_c \leq I_c^{+,l}$ and $T_c^{-,l} \leq T_c \leq T_c^{+,l}$, when sampling level l is applied.

Front-to-Back Compositing on Finest Level: In a second step, the algorithm composites irradiance and transparency bounds for partial cell sequences $c_1, \ldots, c_m$, $m = 1, \ldots, L$, but only at the finest sampling level $\vec{l} = (0, \dots, 0)$, i.e. without any sampling errors. Considering an arbitrary ray, the composited upper bounds of this ray are defined as

Ground-Truth Bounds

$$\widetilde{I}^{+}_{m} = \sum_{c=1}^{m} I^{+,0}_{c} \prod_{d=1}^{c-1} T^{+,0}_{d}, \qquad \widetilde{T}^{+}_{m} = \prod_{c=1}^{m} T^{+,0}_{c}. \tag{6.6}$$

Similarly, lower bounds $\widetilde{I}^{-}_{m}$ and $\widetilde{T}^{-}_{m}$ are derived. With these bounds, the algorithm has collected all the information necessary to deduce the individual cell levels l^{opt}_{c}.

Back-to-Front Exchange of Levels: The iteration starts by assigning the finest sampling level to the complete cell sequence. It then walks backwards from cell $c_L, \dots, c_1$ and tries to replace fine by coarser sampling levels, as long as the screen space error is not exceeded. In other words, it greedily exchanges levels of cells while iterating over the full cell sequence of a ray bundle until the error bound E_I is reached. The compositing equation which supports individual sampling levels $\vec{l} = (l_1, \dots, l_L)$ is given as

Greedy Exchange of Levels

$$\widetilde{I}^{+,\vec{l}} = \sum_{c=1}^{L} I^{+,l_c}_{c} \prod_{d=1}^{c-1} T^{+,l_d}_{d}, \qquad \widetilde{T}^{+,\vec{l}} = \prod_{c=1}^{L} T^{+,l_c}_{c}, \tag{6.7}$$

where l_c is the sampling level of cell c. Similarly, $\widetilde{I}^{-,\vec{l}}$ and $\widetilde{T}^{-,\vec{l}}$ are derived. Sec. 6.4.2 describes the corresponding error estimation in detail. The optimal sampling levels are then fed back to the cell merging step in order to retrieve the final sampling rate as shown in Fig. 6.6.

6.4.1 Cell Irradiance Boundaries

Before the greedy approach can start upper bounds $I^{+,l}_{c}, T^{+,l}_{c}$ and lower bounds $I^{-,l}_{c}, T^{-,l}_{c}$ for each sampling level l and for each cell are required. Lets take a look at how to compute lateral boundaries in order to composite these bounds. For simplicity, lets focus on a single cell c which contains samples $k = 0, \dots, D$ as depicted in Fig. 6.7. We will first examine how to compute accurate bounds for level $l = 0$ and then derive corresponding bounds for larger sampling levels:

Lateral boundaries on level $l = 0$: We estimate the maximum and minimum for all $D \times D$ samples at each sampling depth s_k. Since CSPH quantities are always affine combinations of the particle quantities (cf. Sec. 3.1.4),

Min/Max Particle Quantities

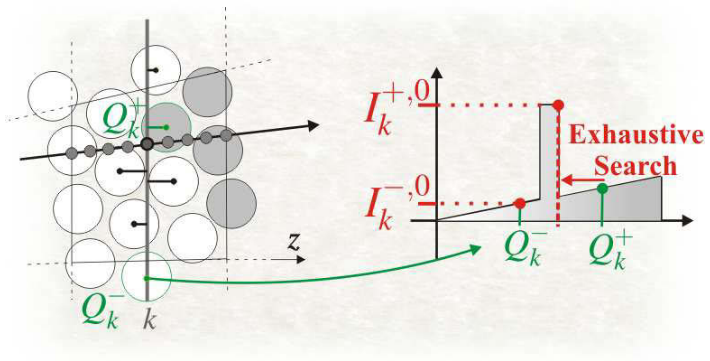


Figure 6.7: An exhaustive search in the transfer function's parameter domain detects extremes $I_k^{+,0}, I_k^{-,0}$ for maximum and minimum particle quantities Q_k^+, Q_k^- in the lateral neighborhood of samples at depth s_k.

minimum and maximum quantities are bounded by the minimum and maximum particle quantities of particles that contribute to the lateral neighbourhood at depth s_k, i.e.

$$Q_k^+ = \max_{|z_j - s_k| < h_j} Q_j, \; Q_k^- = \min_{|z_j - s_k| < h_j} Q_j,$$

where z_j is the z-coordinate of particle j in view space and h_j its radius. Higher order transfer functions can introduce high frequencies as shown in Fig. 6.5. Upper and lower bounds Q_k^+, Q_k^- for the quantity field are thus transferred to irradiance and transparency values using an exhaustive search for extremes in the respective transfer functions as shown in Fig. 6.7:

Exhaustive Extrema Search

$$\begin{aligned} I_k^{+,0} &= \max_{q \in [Q_k^-, Q_k^+]} \tau_I(q), & I_k^{-,0} &= \min_{q \in [Q_k^-, Q_k^+]} \tau_I(q), \\ T_k^{+,0} &= \max_{q \in [Q_k^-, Q_k^+]} \tau_T(q), & T_k^{-,0} &= \min_{q \in [Q_k^-, Q_k^+]} \tau_T(q). \end{aligned}$$

Lateral boundaries on levels $l > 0$: A closer inspection of Eq. (6.5) reveals that applying larger sampling distances increases the bounds I_c^+ and T_c^+ only if the irradiance or transparency value at the coarser sampling position is larger than the one on the finest level. Thus, for levels $l > 0$, one simply can replace the original irradiance and transparency by the maximum (or minimum for the lower bound) of this value and the value induced by a

Higher Sampling Levels

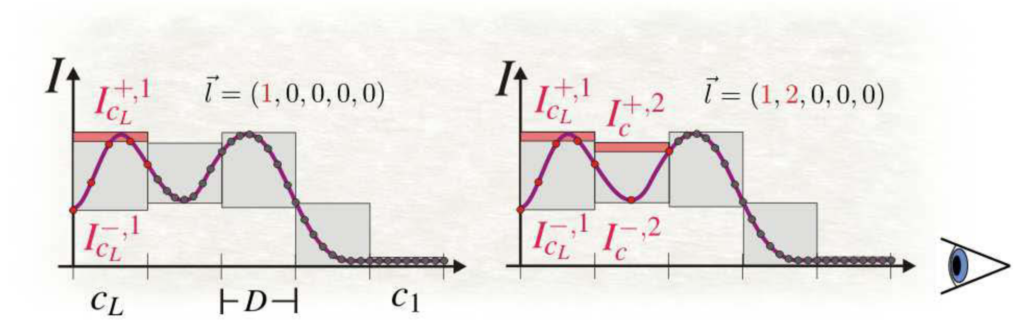


Figure 6.8: Back to front exchange of sampling levels. In the depicted step, the greedy algorithm estimates the screen space error (red area) for sampling level $l_c = 2$ of cell c, i.e. by choosing bounds $I_c^{+/-,2}$ in combination with the composited background irradiances $\widetilde{I}_{back}^{+/-}$.

coarser sampling. In detail, for samples $k = 0, \ldots, D$ one computes

$$I_k^{+,l} = \max\left\{I_k^{+,0}, I^{+,0}_{\lfloor \frac{k}{2^l} \rfloor \cdot 2^l}\right\}, \qquad I_k^{-,l} = \min\left\{I_k^{-,0}, I^{-,0}_{\lfloor \frac{k}{2^l} \rfloor \cdot 2^l}\right\},$$
$$T_k^{+,l} = \max\left\{T_k^{+,0}, T^{+,0}_{\lfloor \frac{k}{2^l} \rfloor \cdot 2^l}\right\}, \qquad T_k^{-,l} = \min\left\{T_k^{-,0}, T^{-,0}_{\lfloor \frac{k}{2^l} \rfloor \cdot 2^l}\right\}.$$

Due to this restrictive estimation it is ensured that the composited values within this cell stays within these boundaries. We will now see how to use composited irradiance and transparency bounds per cell to estimate the screen space error.

6.4.2 Greedy Optimization of Sampling Levels

Incremental Cell Marching

This step analyses the individual cell contribution to the final pixel irradiance values for the rays in the full ray bundle and leads to individual levels l_c^{opt} for each cell c. The algorithm works by exchanging particle levels from back to front. Given the volume rendering at the finest level over the full cell sequence, it decomposes the rendering by splitting off the last cell and replacing it by coarser sampling level l (cf. Eq. (6.5)) which for both bounds read

$$\widetilde{I}_L^{+} = \widetilde{I}_{L-1}^{+} + \widetilde{T}_{L-1} \cdot I_L^{+,0} \leq \widetilde{I}_{L-1}^{+} + \widetilde{T}_{L-1} \cdot I_L^{+,l} =: \widetilde{I}_L^{+,l},$$
$$\widetilde{I}_L^{-} = \widetilde{I}_{L-1}^{-} + \widetilde{T}_{L-1} \cdot I_L^{-,0} \geq \widetilde{I}_{L-1}^{-} + \widetilde{T}_{L-1} \cdot I_L^{-,l} =: \widetilde{I}_L^{-,l}.$$

The algorithm allows the last cell to be coarsened until the maximal level l_L^{max} is reached, as long as the irradiance bound stays below the given screen space error E_I. In detail, it sets

$$l_L^{opt} = \arg\max_{l \leq l_c^{max}} \left\{ \widetilde{I}_L^{+,l} - \widetilde{I}_L^{-,l} < E_I \right\}. \tag{6.8}$$

Maximum Sampling Levels

However, cells with $I_c^{-,l} \ll I_c^{+,l}$, i.e. with high irradiance variations, have strong influence on the error bounds, thus they may easily "consume" the given error margin without leading to significant increases in the sampling distance. Therefore, a maximal sampling level per cell is defined based on a separate error threshold E_c:

$$l_c^{max} = \arg\max_{l} \left\{ I_c^{+,l} - I_c^{-,l} < E_c \right\},$$

which suppresses coarser sampling for those cells that show large irradiance and/or transparency variations.

Compositing of Irradiance Bounds

As shown in Fig. 6.8, having decided on the sampling level for the last cell, it collects the final results for cell c_L in background irradiances $I_{back}^{+/-} = I_L^{+/-,l_L^{opt}}$. Now, the algorithm proceeds backwards, i.e. it sequentially tests coarser sampling levels l in cell c in back-to-front order:

$$\begin{aligned} \widetilde{I}_c^{+} + \widetilde{T}_c^{+} \cdot I_{back}^{+} &= \widetilde{I}_{c-1}^{+} + \widetilde{T}_{c-1}^{+} \cdot (I_c^{+,0} + T_c^{+,0} \cdot I_{back}^{+}) \\ &\leq \widetilde{I}_{c-1}^{+} + \widetilde{T}_{c-1}^{+} \cdot (I_c^{+,l} + T_c^{+,l} \cdot I_{back}^{+}) = I_c^{+,l} \end{aligned}$$

and

$$\widetilde{I}_c^{-} + \widetilde{T}_c^{-} \cdot I_{back}^{-} \geq \widetilde{I}_{c-1}^{-} + \widetilde{T}_{c-1}^{-} \cdot (I_c^{-,l} + T_c^{-,l} \cdot I_{back}^{-}) = I_c^{-,l}.$$

Analogously to Eq. (6.8) it computes the optimal level l_c^{opt} and, finally, it updates the background irradiances:

$$I_{back}^{+/-} \leftarrow I_c^{+/-,l_c^{opt}} + T_c^{+/-,l_c^{opt}} \cdot I_{back}^{+/-}$$

Note, due to the definition of the error bounds, if $\widetilde{I}^{+,\vec{l}} - \widetilde{I}^{-,\vec{l}} < E_I$, it is guaranteed that the error induced by the sampling using $\vec{l}$ is below the given screen space error E_I.

6.5 Implementation Details

Algorithm 7: Volume ray casting featuring an adaptive sampling step size (orange) and cache-coherent memory access (green). Particles contribute to D samples of a ray at once using a thread-local cache.

```
$I = \mathbf{0}, T = 1,\ \mathbf{x}[D] = \mathbf{0},\ Q[D] = 0,\ V[D] = 0$
foreach cell $c = c_1 < c_L$ do
    $[j_c, N_c, i_c, \Delta_c]$ = read_celldata($c$);
                    Sampling
    ------------------------------------------------

    foreach sample $k = 0 < D$ do
        $Q[k] = V[k] = 0$
        $\mathbf{x}[k] = \mathbf{x}(s(i_c + \Delta i_c\, k))$
    foreach particle $j = j_c < j_c + N_c$ do
        $[\mathbf{x}_j, h_j, Q_j, V_j]$ = read_particledata($j$);
        foreach sample $k = 0 < D$ do
            $Q[k] = Q[k] + Q_j\ V_j\ W_j(\mathbf{x}[k])$
            $V[k] = V[k] + V_j\ W_j(\mathbf{x}[k])$
    foreach sample $k = 0 < D$ do
        if $(V[k] > 0)\ Q[k] = Q[k]/V[k]$
                    Compositing
    ------------------------------------------------

    foreach sample $k = 0 < D$ do
        $\Delta s_k = s(i_c + \Delta i_c\,(k+1)) - s(i_c + \Delta i_c\, k)$
        $I = I + T\,\tau_I(Q[k])$
        $T = T\,\tau_T(Q[k])^{\Delta s_k}$
```

Alg. 7 gives a pseudo-code for the resulting ray casting algorithm implementing the volume rendering integral given in Eq. (6.1). Since consecutive cells in view direction are neighbours in memory no specific cell finding logic is required. Since each cell has a fixed number of samples, coherent memory access between the $D \times D$ adjacent rays is ensured (cf. green code lines in Alg. 7). Using a small thread-local cache, particles scatter their data to D samples of a ray at once which further reduces memory traffic since each particle is read only once for each cell. Coherent Traversal

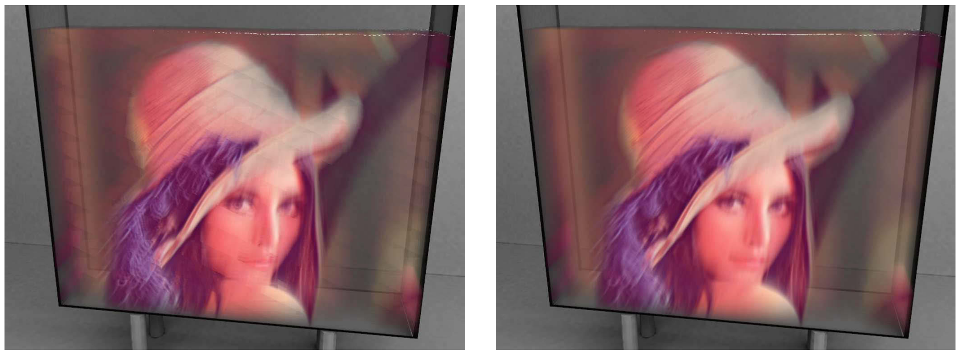


Figure 6.9: Adaptive sampling at fluid surfaces results in visual artifacts due to under-sampling 6.9(a). In contrast, Fig 6.9(b) shows the result with proper surface detection via surface particles (cf. Sec.4.3).

6.6 Results and Discussion

Scenarios

In order to demonstrate the performance and to evaluate the image quality resulting the proposed volume ray casting has been tested for various scenarios. Among them in this chapter three have been examined closer. The flooding of a valley (see Fig. 6.5), the showering of the Stanford bunny with water colors (see Fig. 6.1) and heavy rigid objects have been dropped into a tank containing a fluid that has been dyed with the famous image of Lena (see Fig. 6.9). Simulations and renderings of all scenes were carried out on an NVIDIA GeForce GTX Titan with 6 GiB of VRAM. Table 6.1 shows particle counts, timings as well as errors using different tolerances E_I for the three scenes. The following discussion is split into an analysis of the image quality and into an presentation of performance characteristics.

Image quality As can be clearly seen in Tab. 6.1, screen-space errors stay below the feature-preserving error-bound, i.e. user-defined error tolerance E_I is preserved at any time. The right hand side of Fig. 6.1 shows the inner structure of the adaptive grid for $E_I = 0.5$. One can easily identify homogeneous regions and match them with the merged cells sampled with increased sampling distance.

Dynamic Grid Adaption

Error Tresholds

In order to illustrate the errors caused by applying different error tolerances, the Lena scene has been rendered with $E_I = 0.2$, 0.5 and 0.8. Using these values, maximum pixel errors of 0.006, 0.01 and 0.019, respectively have been obtained. The image error as shown in Fig. 6.10 increases from

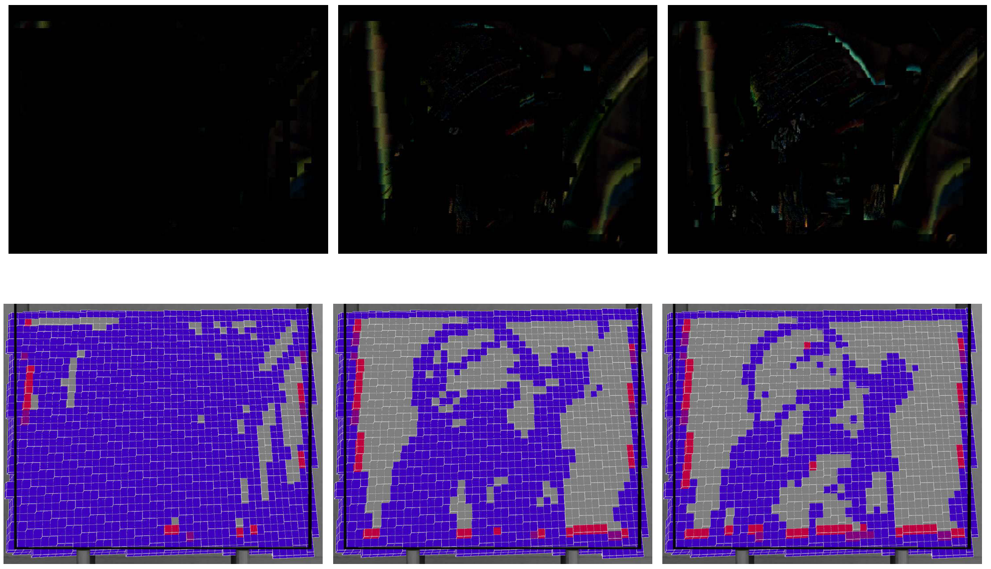
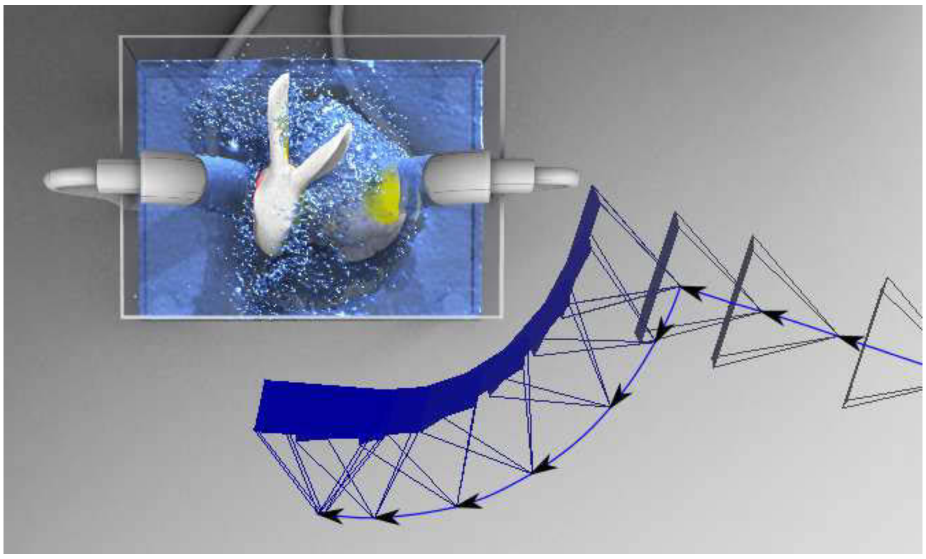


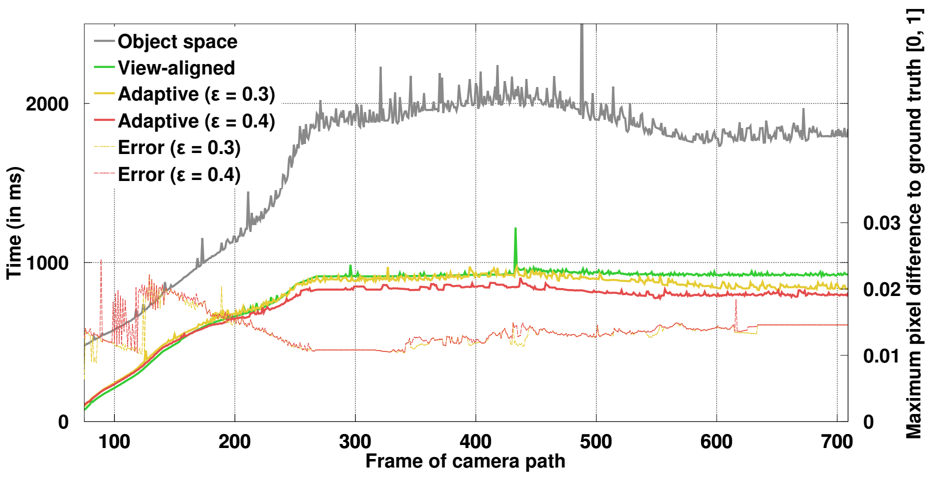
Figure 6.10: Differences to the original Lena rendering in Fig. 6.9 amplified by a factor of 50 for error tolerances 0.2, 0.5, 0.8 from left to right and the corresponding inner cell structure of the view-aligned grid. The colors indicate the sampling distance. Indigo cells are sampled at the smallest sampling distance and cells colored in purple, magenta, and gray with increasing sampling distances.

	View-aligned			**Adaptive, view-aligned**				
Scene (#Part.)	Grid	RC	**Total**	E_I	Error	Merge	RC	**Total**
				∞	0.9	20	395	**442**
Flooding (1 M)	27	550	**577**	0.8	0.08	70	460	**557**
				0.5	0.02	70	470	**567**
Bunny (2.4 M)	50	820	**870**	0.5	0.015	113	514	**677**
				0.2	0.013	113	670	**833**
				0.8	0.019	156	934	**1.14 s**
Lena (7.2 M)	146	1.27 s	**1.42 s**	0.5	0.01	156	995	**1.3 s**
				0.2	0.006	155	1.21 s	**1.51 s**

Table 6.1: GPU timings (in ms if not stated otherwise) of the access structure alone and in comparison with an additional adaptive sampling. 'Grid' gives the timings for the particle-to-cell assignment and setting up the grid, and 'RC' means the time taken for the actual ray casting. 'Merge' is the computation time for the sampling error analysis and the merging of cells. The error is given as the maximum absolute pixel difference from the non-adaptive rendering taken over all pixels. E_I is the user-defined error margin for which ∞ means that all cells are merged regardless of the error.



(a) Camera path



(b) Timing and errors

Figure 6.11: Camera path of the Bunny scene 6.11(a) and the rendering times and maximum per pixel differences 6.11(b) for object-space and view aligned cells, adaptive and non-adaptive samplings. Solid lines are used for timing results and dashed, dotted lines for the maximum per pixel error. The first and last camera positions in 6.11(a) correspond to the first and last data points in 6.11(b), respectively.

left to right as the error tolerance is increased, allowing more cells to be merged. The lower row of Fig. 6.10 captures the resulting grid cell structures and demonstrates how precisely the sampling error analysis reveals salient regions of the underlying quality field, preserving the visible features of the volume. In the clipped view of the grid one can clearly recognize the outlines of the Lena rendering displayed in Fig. 6.9.

The fact that the greedy algorithm uses post-classification ensures that arbitrary transfer functions can be applied. The transfer function applied in Fig. 6.5 contains two spikes effectively creating thin iso-contours in the final rendering. Image quality suffers dramatically if statistical measures like variance applied to pre-classified quantity values are used. Even for such extreme transfer functions, the image error of the adaptive renderings remained well below the tolerance.

Iso-Surface Detection

Performance To assess the performance of the adaptive ray casting, freeze frames of the Bunny scene from different perspectives have been rendered, using the animated camera path shown in Fig. 6.11(a). The renderings were carried out using a view-aligned grid with and without adaptive sampling and have been compared with a state-of-the-art object-space ray caster using an octree as particle access structure to provide a frame of reference for timings. As can be seen, the adaptive ray casting using view-aligned particle access clearly outperformed the object-space ray caster. Due to the memory-coherent particle access and the simplified ray casting that now is free from the complex traversal logic inherent to object-space approaches, speed-ups of more than a factor of 2 have been achieved. This speed up could further be increased by factors of up to 10 and 15 % for an error tolerance E_I of 0.3 and 0.4, respectively, without causing noticeable loss of quality to the rendered images. The timings of the Bunny scene for Tab. 6.1 were obtained by rendering frame 600 of the camera path.

Side-by-Side Performance

However, setting E_I to 0.2 and 0.5, leads to very different performance characteristics. While the speedup of adaptive vs. non-adaptive ray casting using $E_I = 0.2$ was a mere 4.2 % it increased to a speedup of 22 % for $E_I = 0.5$. Using the smallest sampling distance for the Flooding scene, rendering took 577 ms per frame. With no error bound, rendering took only 442 ms, thus, yielding a speedup of 23 %. This speedup was the upper bound of speedup that could be obtained for the scene as all cells that did not merge were either surface cells or didn't find a merge partner. Using our adaptive approach, we obtain speedups of only 1.7 % and 3.5 % for $E_I = 0.5$ and 0.8, respectively. The rendering of the Lena scene at the smallest sampling distance took 1.42 s in total. This scene describes a worst case scenario where nearly no homogeneous regions to merge. With $E_I = 0.2$ the approach cannot compensate for the overhead induced by the sampling error analysis. The total rendering time increased by 6.4 %. For $E_I = 0.5$ and 0.8 it is able to gain speedups of 8.7 % and 20 %, respectively.

Feature Preservation vs Speedup

6.7 Conclusions

In this chapter a fast, high quality volume ray casting technique for unstructured particle data sets has been presented. The technique comprises a sparse perspective, view-aligned grid access data structure and does not require any pre-computations, thus, large dynamic particle data sets can be efficiently visualized. It comprises a built-in empty space skipping and completely releases the ray casting algorithm from any traversal logic. Additionally, it enables an on the fly error analysis for volume rendering of SPH-based quantity fields. A greedy algorithm reveals strict screen space error bounds which have been applied to determine per-cell sampling levels, resulting in a very reliable error control. The per-cell sampling information is used to merge homogeneous cells in order to preserve a constant number of samples per cell, thus shifting computational resources to salient regions of the fluid volume. For scenarios, where the orginal field shows large variance in irradiance, the approach might be too conservative which results in little speed ups. Nevertheless, the user-defined error margins are ensured. If the overall variance in irradiance is small, the proposed algorithm leads to significant rendering speed-ups without sacrificing image quality. It accelerates the rendering in case of scenarios with partially homogeneous regions (up to 20% with a maximal image space error of less than 0.02). The adaptive sampling algorithm nicely preserves features, even in scenarios with highly fluctuating irradiance values, e.g. for dirac-like transfer functions.

Chapter 7
Conclusion

This chapter reflects and summarizes the contributions of this thesis. Its objective is to provide a high-level review that abstracts from the details presented in previous chapters. A short discussion of limitations also leads to directions for future research and its application.

As became clearly visible in the last sections, the contributions of this thesis have the potential to improve the quality of SPH-based fluid animation and they resolve major research gaps for fluid transport problems of incompressible free surface flows. Since computational efficiency and rendering quality have been the fundamental motivation for the presented techniques, they add flow details otherwise not perceivable, which makes the visual results even more attractive. Many aspects may provide room for improvement as will be discussed, but the major contributions clear the way for a wide variety of new effects, which, due to the results and insights presented in here, become attainable in future applications.

7.1 Summary

Perhaps the most critical point when coupling surface and bulk dynamics is modelling of phase singularities and shifting available computational resources toward them while ensuring stability. Furthermore, unstructured particle configurations complicate any field reconstruction in terms of computational efficiency. This makes it especially challenging to produce high-quality renderings from large particle sets in a reasonable amount of time, despite the parallel nature of particle systems. This thesis provides efficient solutions to all of these issues. Specifically, it offers a stable implicit surface model, a temporally coherent blending of particle sets, and an adaptive, feature-preserving ray casting.

Stable Implicit Models

Surface Modelling: While the SPH literature mainly focuses on reconstructing of smooth surfaces for rendering purposes, this thesis extends the meaning of surface modelling by incorporating surface-bulk interactions. Compared with explicit polygonal meshes, the presented surface model offers a robust implicit representation with much greater capability to handle complex free surfaces including thin sheets and splashes.

Consistent Reconstructions

Due to computation of surface area, a consistent representation of quantity fields (both) on the surface and within the fluid's bulk is possible. Consistency is achieved by spreading the surface into the fluid's bulk, where it is sampled by bulk particles. Possibly the biggest strength of this consistent field reconstruction is, that practically all bulk dynamics can now directly be mapped to the embedded surface without reformulation.

Reliable Surface Detection

Over the course of this thesis, the surface model has proven to be a powerful tool to improve the stability at free surfaces. Reliable surface detection mechanisms are especially important for the quality of adaptive sampling mechanisms.

Particle Resolution

Adaptive Sampling: While large particle numbers are mandatory to resolve details on the fluid's surface, a high bulk resolution is often not required. Reducing the particle resolutions during simulation, or alternatively, reducing the number of samples during rendering, has proven to be a crucial step for reintroducing efficiency.

Temporal Coherence

The first principle to learn when sampling particle fields is that an instant replacement of particles introduces large pressure waves. The proposed blending of particle levels smooths this field perturbations over several time steps and preserves the temporal coherence of particle neighbourhoods.

As a result, the overall simulation time is drastically reduced.

Ray Samples

The value that is the particle count for the simulation, is for the rendering the number of sample positions along viewing rays. Naive sampling results in clearly visible artifacts, especially close to the free surface. The solution in this case is to estimate screen space errors on the fly while reducing sampling rates.

Error Estimation

In general, it has become recognized, that an error estimation provides a reliable a priori control of the introduced sampling errors. At first sight, error estimation often seems to reduce efficiency of adaptive methods, but preventing errors allows subsequent operations to perform much faster. The quality of the sampling is significantly improved, which finds expression in both, our simulation outcome and rendering results: Large integration time steps can be stably maintained and rendering artifacts are avoided, which results in smoother fields.

Data-Parallel Algorithms

Data Parallelism: Data-parallel realizations of SPH on GPUs, in the past, quite often let to specialized solutions optimized for specific hardware features. The proposed techniques of this thesis show that currently, with the ongoing development of generic programming APIs, quite the opposite is possible. Namely, coherent memory access and level of parallelism determine the performance of data parallel algorithms. Keeping solutions simple is often advantageous over developing specialized solutions that target specific hardware features. Fine granulated modules with small functional interfaces allow algorithms to naturally adapt to massive parallel architectures without introducing major structural changes.

7.2 Future Work

The presented sampling and reconstruction techniques introduce various options for future research. The most promising directions are fully conservative multi-scale simulations, the modelling of a scalable surface tension term, and the simulation of thermodynamic surface effects. All of these have the potential to greatly improve SPH-based fluid simulations.

Controllable Particle Configurations

Conservative Multi-Scale Simulations: Our temporal blending mechanism enables seamless optimization of local particle configurations even for stiff systems, e.g. in the future, it might be applied inter alia to stably adjust positions of ghost particles [SB12] or constrained particles [LD09].

Coupling of Particle Levels

However, direct coupling of particle levels using different smoothing radii [APKG07, ZSP08, HHK08] has shown its limitations: Mass differences at the contact line restrict neighboring particles to a factor of two in size [BOT01]. This constrains sampling mechanisms to increase resolutions gradually [APKG07]. Regarding this aspect, loosely coupled particle levels [KAG*06, SG11] provide a very promising direction for future research. However, inconsistencies between resolution levels introduce conservation problems. Unnatural creation and disappearance of transported materials lead to clearly visible artifacts. Recent work [HS13] shows some development in this direction by restricting the number of emitted particles. Still, the application to transport problems is unclear, and possibly a combination with other fluid solvers [CIPT14] can be employed.

Data Compaction

Beside computational efficiency, GPU algorithms often are limited by the available global memory. Data-parallel SPH simulations would thus profit from compaction algorithms, which more efficiently store particle data.

Scale- and Concentration-dependent Surface Tension: While formulations of physics via interactions of particle pairs are straightforward, finding a working set of parameters is often very complicated. Dimensionless formulations, which, for example, replace density by particle number density [HA06, SP08], have the advantage that their solution is invariant to the actual scale of the physical simulation domain.

Dimensionless Formulations

Variable Surface Tension

However, reinforcing scale-dependent surface dynamics such as surface tension is not trivial. As described by the Weber number, at smaller scales flows become laminar and features like splashes, which are very common for free surface SPH, disappear. It is at these scales, that concentrations of transported materials have a strong effect on the strength of surface tension, especially the Maragoni forces, which contract the fluid in tangential direction [FAB*11]. According to our observations, pressure solvers [BT07b, SP09b] still dominate convection. Surface tension forces [MCG03, BT07b] are not able to break loose the regular structure of the densely-packed particles [SP08].

Tensile Instability

Furthermore, current implementations suffer from under-resolved particle neighborhoods at free surfaces. The resulting pressure variations [Mon00] make it even more complicated to realize effective surface tension terms. Recent work [AAT13, ICS*13] appears promising to overcome some of these limitations. Nevertheless, finding a scalable formulation of surface dynamics would open a wide range of animation scenarios and would drastically increase the applicability of SPH.

Temperature-dependent (Surface) Effects: For the sake of simplicity, this thesis has assumed isothermal conditions. However, diffusion speed and reaction processes in nature strongly depend on the temperature of the system. Including such temperature-dependent effects requires the modelling of thermal conductivity [Bro85, SAC*99, CM99], which itself is a special form of fluid transport using Fourier's Law. To efficiently model heat conduction, diffusion operations must become less dependent on the choice of smoothing radii (cf. Sec. 3.2.2).

Thermal Conductivity

Turbulence Modelling

Additionally, temperature differences lead to density changes including large density contrasts [SP08], which in the context of SPH have only been considered for compressible fluids [MSKG05, SSP07]. Furthermore, heating creates turbulence, which needs to be accurately captured by fluid particles. However, artificial viscosity, which is needed to stabilize time steps in SPH, induces undesired damping of flow dynamics.

Evaporation and Condensation

Once this problem is solved, effects, in which mass loss and gain is desired, such as evaporation and condensation, become of interest. The simulation must be able to transport mass across the phase interface, which is somewhat similar to modeling of granular materials [LAD08, LD09]. In contrast to free surface scenarios, an efficient model for the adjacent air phase becomes mandatory.

7.3 Final Words

The beauty of SPH lies in its simplicity of describing physics in a natural way. Quite often it was observed that SPH itself favors simple over specialized solutions. This observation is clearly reflected in recent systems, which, despite the complex physics they support, are completely described on the basis of particle-to-particle interactions instead of global systems. Despite the difficulties of the first solvers, this is the very reason, that current grid-free methods have been so successful. We believe that, given enough particles, the applications of particle-based representations are nearly limitless. Hopefully, this work will inspire researchers to push SPH further to new frontiers. Last but not least, after all the hard work, Smoothed Particle Hydrodynamics provide a lot of fun.

Bibliography

[AA04] Alexa M., Adamson A.: On normals and projection operators for surfaces defined by point sets. In *Symposium on Point-based Graphics (SPBG)* (2004), pp. 149–155. 43

[AAIT12] Akinci G., Akinci N., Ihmsen M., Teschner M.: An efficient surface reconstruction pipeline for particle-based fluids. In *Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS)* (2012), pp. 61–68. 87

[AAOT13] Akinci G., Akinci N., Oswald E., Teschner M.: Adaptive surface reconstruction for sph using 3-level uniform grids. In *Winter School of Computer Graphics (WSCG)* (2013), pp. 195–204. 87

[AAT13] Akinci N., Akinci G., Teschner M.: Versatile surface tension and adhesion for sph fluids. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 182:1–182:8. 9, 24, 34, 109

[ACAT13] Akinci N., Cornelis J., Akinci G., Teschner M.: Coupling elastic solids with smoothed particle hydrodynamics fluids. *Journal of Computer Animation and Virtual Worlds (CAVW) 24* (2013), 195–203. 24

[AGDJ08] Anderson J. C., Garth C., Duchaineau M. A., Joy K. I.: Discrete multi-material interface reconstruction for volume fraction data. In *Visualization and Graphics Technical Community Conference (VGTC) on Visualization* (2008), pp. 1015–1022. 22

[AHA10] Adami S., Hu X. Y., Adams N. A.: A conservative SPH method for surfactant dynamics. *Journal of Computational Physics (JCP) 229*, 5 (2010), 1909–1926. 9, 42, 43, 56

[AIA*12] Akinci N., Ihmsen M., Akinci G., Solenthaler B., Teschner M.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics (TOG)* (2012), 62:1–62:8. 8, 24, 29, 34, 53

[AIAT12] AKINCI G., IHMSEN M., AKINCI N., TESCHNER M.: Parallel surface reconstruction for particle-based fluids. *Computer Graphics Forum (CGF) 31* (2012), 1797–1809. 10, 36, 43, 86, 87

[AIY*04] AMADA T., IMURA M., YASUMURO Y., MANABE Y., CHIHARA K.: Particle-based fluid simulation on GPU. In *ACM Workshop on General-Purpose Computing on Graphics Processors (GPCGP)* (Aug. 2004). 38

[ALD06] ADAMS B., LENAERTS T., DUTRÉ P.: *Particle splatting: interactive rendering of particle-based simulation data.* Tech. rep., Katholieke Universiteit Leuven, 2006. 87

[AN05] ANGELIDIS A., NEYRET F.: Simulation of smoke based on vortex filament primitives. In *ACM Symposium on Computer Animation (SCA)* (2005), pp. 87–96. 24

[AO11] ALDUAN I., OTADUY M. A.: Sph granular flow with friction and cohesion. In *ACM Symposium on Computer Animation (SCA)* (2011), pp. 25–32. 8

[APKG07] ADAMS B., PAULY M., KEISER R., GUIBAS L. J.: Adaptively sampled particle fluids. *ACM Transactions on Graphics (TOG) 26* (2007), 48:1–48:7. 9, 10, 24, 36, 42, 43, 62, 64, 66, 67, 68, 78, 81, 87, 88, 108

[ATT12] ANDO R., THUREY N., TSURUNO R.: Preserving fluid sheets with adaptively sampled anisotropic particles. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 18*, 8 (Aug. 2012), 1202–1214. 23

[ATW13] ANDO R., THÜREY N., WOJTAN C.: Highly adaptive liquid simulations on tetrahedral meshes. *ACM Transactions on Graphics (TOG)* (July 2013). 23

[Bar97] BARAFF D.: An introduction to physically based modeling: rigid body simulation 1 - unconstrained rigid body dynamics. In *SIGGRAPH '97 Course Notes* (1997), p. 97. 34

[BB08] BATTY C., BRIDSON R.: Accurate viscous free surfaces for buckling, coiling, and rotating liquids. In *ACM Symposium on Computer Animation (SCA)* (2008), pp. 219–228. 22

[BB12] Boyd L., Bridson R.: Multiflip for energetic two-phase fluid simulation. *ACM Transactions on Graphics (TOG) 31*, 2 (2012), 16:1–16:12. 22

[BBB07] Batty C., Bertails F., Bridson R.: A fast variational framework for accurate solid-fluid coupling. *ACM Transactions on Graphics (TOG) 26*, 3 (2007). 22

[BBB10] Brochu T., Batty C., Bridson R.: Matching fluid simulation elements to surface geometry and topology. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 47:1–47:9. 22

[BCOS01] Bertalmio M., Cheng L.-T., Osher S., Sapiro G.: Variational problems and partial differential equations on implicit surfaces: the framework and examples in image processing and pattern formation. *Journal of Computational Physics (JCP) 174* (2001), 759–780. 9, 43, 45

[BGB11] Bhatacharya H., Gao Y., Bargteil A.: A level-set method for skinning animated particle data. In *ACM Symposium on Computer Animation (SCA)* (2011), pp. 17–24. 10, 24, 87

[BGOS06] Bargteil A. W., Goktekin T. G., O'brien J. F., Strain J. A.: A semi-lagrangian contouring method for fluid simulation. *ACM Transactions on Graphics (TOG) 25*, 1 (2006), 19–38. 22

[BHMF08] Beyer J., Hadwiger M., Möller T., Fritz L.: Smooth mixed-resolution gpu volume rendering. In *Visualization and Graphics Technical Community (VGTC) conference on Point-Based Graphics (SPBG)* (2008), pp. 163–170. 88

[BIT09] Becker M., Ihmsen M., Teschner M.: Corotated sph for deformable solids. In *Fifth Eurographics conference on Natural Phenomena (NPH)* (2009), pp. 27–34. 24

[BK02] Bonet J., Kulasegaram S.: A simplified approach to enhance the performance of smooth particle hydrodynamics methods. *Journal of Applied Mathematics and Computation (IJAMC) 126*, 2-3 (2002), 133–155. 8, 30, 47, 73

[BKDG98] Belytschko T., Krongauz Y., Dolbow J., Gerlach C.: On the completeness of meshfree particle methods. *International Journal of Numerical Methods in Engineering (IJNME) 43* (1998), 785–819. 31

[BKOF96] Belytschko T., Krongauz Y., Organ D., Fleming M.: Meshless methods: An overview and recent developments. In *Computer Methods in Applied Mechanics and Engineering (CMAME)* (1996), pp. 3–47. 9

[BKP*10] Botsch M., Kobbelt L., Pauly M., Alliez P., Lévy B.: *Polygon mesh processing.* CRC press, 2010. 43

[BKZ92] Brackbill J. U., Kothe D. B., Zemach C.: A continuum method for modeling surface tension. *J. Comput. Phys. 100*, 2 (June 1992), 335–354. 19

[BLS12] Bodin K., Lacoursiere C., Servin M.: Constraint fluids. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 18*, 3 (2012), 516–526. 24

[BOT01] Børve S., Omang M., Trulsen J.: Regularized smoothed particle hydrodynamics: a new approach to simulating magnetohydrodynamic shocks. *The Astrophysical Journal Supplement Series (AJSS) 561* (2001), 345–367. 10, 68, 82, 88, 108

[BR86] Brackbill J. U., Ruppel H. M.: Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics (JCP) 65*, 2 (1986), 314–343. 22

[Bri08] Bridson R.: *Fluid Simulation for Computer Graphics.* A K Peters/CRC Press, 2008. 15, 23, 35

[Bro85] Brookshaw L.: A method of calculating radiative heat diffusion in particle simulations. In *In Astronomical Society of Australia (ASA)* (1985), vol. 6, pp. 207–210. 8, 32, 109

[BS09] Berthier J., Silberzan P.: *Microfluidics for biotechnology.* Artech House, 2009. 3, 15, 53

[BT07a] Bajpai D., Tyagi V.: Laundry detergents: an overview. *Journal of Oleo Science (JOS) 56*, 7 (2007), 327–340. 3

[BT07b] Becker M., Teschner M.: Weakly compressible SPH for free surface flows. In *ACM Symposium on Computer Animation (SCA)* (2007), pp. 209–217. 8, 9, 24, 25, 32, 34, 35, 43, 53, 62, 109

[BTT09] Becker M., Tessendorf H., Teschner M.: Direct forcing for lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 15*, 3 (2009), 493–503. 8, 24

[BXH10] Batty C., Xenos S., Houston B.: Tetrahedral embedded boundary methods for accurate and flexible adaptive fluids. In *Computer Graphics Forum (CGF)* (2010). 22

[CBP05] Clavet S., Beaudoin P., Poulin P.: Particle-based viscoelastic fluid simulation. In *ACM Symposium on Computer Animation (SCA)* (2005), SCA '05, pp. 219–228. 8, 9

[CCM*04] Cardoze D., Cunha A., Miller G. L., Phillips T., Walkington N.: A bezier-based approach to unstructured moving meshes. In *Annual Symposium on Computational Geometry (SoCG)* (2004), SCG '04, pp. 310–319. 23

[CEL06] Colin F., Egli R., Lin F. Y.: Computing a null divergence velocity field using smoothed particle hydrodynamics. *Journal of Computational Physics (JCP) 217*, 2 (2006), 680–692. 32, 51

[CGFO06] Chentanez N., Goktekin T. G., Feldman B. E., O'Brien J. F.: Simultaneous coupling of fluids and deformable bodies. In *ACM Symposium on Computer Animation (SCA)* (2006), pp. 83–89. 22

[CIPT14] Cornelis J., Ihmsen M., Peer A., Teschner M.: IISPH-FLIP for incompressible fluids. *Computer Graphics Forum (CGF) 33*, 2 (2014), 255–262. 24, 108

[CKS00] Cottet G.-H., Koumoutsakos P., Salihi M. L. O.: Vortex methods with spatially varying cores. *Journal of Computational Physics (JCP) 162* (2000), 164–185. 9, 62, 66

[CKSW08] Cuntz N., Kolb A., Strzodka R., Weiskopf D.: Particle level set advection for the interactive visualization of unsteady 3D flow. *Computer Graphics Forum (CGF) 27*, 3 (2008), 719–726. 22

[Cle98] Cleary P. W.: Modelling confined multi-material heat and mass flows using sph. *Applied Mathematical Modelling (AMM) 22*, 12 (1998), 981 – 993. 5

[CLT07] Crane K., Llamas I., Tariq S.: *Real Time Simulation and Rendering of 3D Fluids.* Addison-Wesley, 2007, ch. 30. 21

[CM99] Cleary P. W., Monaghan J. J.: Conduction modelling using smoothed particle hydrodynamics. *Journal of Computational Physics (JCP) 148* (1999), 227–264. 32, 36, 51, 109

[CM12] Chentanez N., Müller M.: Mass-conserving eulerian liquid simulation. In *ACM Symposium on Computer Animation (SCA)* (2012), pp. 245–254. 21

[CMT04] Carlson M., Mucha P. J., Turk G.: Rigid fluid: animating the interplay between rigid bodies and fluid. *ACM Transactions on Graphics (TOG) 23*, 3 (2004), 377–384. 21

[CMVHT02] Carlson M., Mucha P. J., Van Horn III R. B., Turk G.: Melting and flowing. In *ACM Symposium on Computer Animation (SCA)* (2002), pp. 167–174. 21

[CPK02] Chaniotis A. K., Poulikakos D., Koumoutsakos P.: Remeshed smoothed particle hydrodynamics for the simulation of viscous and heat conducting flows. *Journal of Computational Physics (JCP) 182* (2002), 67–90. 9, 23, 62

[CR99] Cummins S. J., Rudman M.: An sph projection method. *Journal of Computational Physics (JCP) 152*, 2 (1999), 584–607. 24

[CTG10] Cohen J. M., Tariq S., Green S.: Interactive fluid-particle simulation using translating eulerian grids. In *Symposium on Interactive 3D Graphics and Games (i3D)* (2010), pp. 15–22. 21

[CWSO13] Clausen P., Wicke M., Shewchuk J. R., O'brien J. F.: Simulating liquids and solid-liquid interactions with lagrangian meshes. *ACM Transactions on Graphics (TOG) 32*, 2 (2013), 17:1–17:15. 23, 42, 43

[DC96] Desbrun M., Cani M.-P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Workshop on Computer Animation and Simulation (EGCAS)* (1996), pp. 61–76. 15, 23, 28, 35, 62

[DC99] Desbrun M., Cani M.-P.: *Space-time adaptive simulation of highly deformable substances.* Tech. Rep. 3829, 1999. 62, 66

[EHK*06] Engel K., Hadwiger M., Kniss J. M., Rezk-salama C., Weiskopf D.: *Real-time volume graphics.* A. K. Peters, Ltd., Natick, MA, USA, 2006. 85, 86, 87

[ELF05] Enright D., Losasso F., Fedkiw R.: A fast and accurate semi-lagrangian particle level set method. *Computers and Structures 83*, 6-7 (2005), 479–490. 21

[EMB11] Erleben K., Misztal M. K., Bäerentzen J. A.: Mathematical foundation of the optimization-based fluid animation method. In *ACM Symposium on Computer Animation (SCA)* (2011), pp. 101–110. 23

[EMF02] Enright D., Marschner S., Fedkiw R.: Animation and rendering of complex water surfaces. *ACM Transactions on Graphics (TOG) 21*, 3 (2002), 736–744. 21

[EQYF13] English R. E., Qiu L., Yu Y., Fedkiw R.: Chimera grids for water simulation. In *ACM Symposium on Computer Animation (SCA)* (2013), pp. 85–94. 22

[ETK*07] Elcott S., Tong Y., Kanso E., Schröder P., Desbrun M.: Stable, circulation-preserving, simplicial fluids. *ACM Transactions on Graphics (TOG) 26*, 1 (2007). 22

[FAB*11] Fell D., Auernhammer G. K., Bonaccurso E., Liu C., Sokuler R., Butt H.-J.: Influence of surfactant concentration and background salt on forced dynamic wetting and dewetting. *Langmuir 27*, 6 (2011), 2112. 3, 109

[FAW10] Fraedrich R., Auer S., Westermann R.: Efficient high-quality volume rendering of SPH data. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 16*, 6 (2010), 1533–1540. 10, 24, 86, 87, 88

[FB07] Feldman J., Bonet J.: Dynamic refinement and boundary contact forces in SPH with applications in fluid flow problems. *International Journal of Numerical Methods in Engineering (IJNME) 72*, 3 (2007), 295–324. 62, 66

[FGE10] Falk M., Grottel S., Ertl T.: Interactive image-space volume visualization for dynamic particle simulations. In *Proceedings of The Annual SIGRAD Conference* (2010), pp. 35–43. 86

[FM96] Foster N., Metaxas D.: Realistic animation of liquids. *Journal of Graphical Models and Image Processing (CVGIP) 58*, 5 (1996), 471–483. 15, 21

[FOA03] Feldman B. E., O'Brien J. F., Arikan O.: Animating suspended particle explosions. *ACM Transactions on Graphics (TOG) 22*, 3 (2003), 708–715. 21

[FOK05] Feldman B. E., O'Brien J. F., Klingner B. M.: Animating gases with hybrid meshes. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 904–909. 22

[FOKG05] Feldman B. E., O'Brien J. F., Klingner B. M., Goktekin T. G.: Fluids in deforming meshes. In *ACM Symposium on Computer Animation (SCA)* (2005), pp. 255–259. 22

[FSJ01] Fedkiw R., Stam J., Jensen H. W.: Visual simulation of smoke. In *In Proc. ACM SIGGRAPH* (2001), pp. 15–22. 21

[FSW09] Fraedrich R., Schneider J., Westermann R.: Exploring the millennium run - scalable rendering of large-scale cosmological datasets. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 15*, 6 (2009), 1251–1258. 10, 88

[GB13] Gerszewski D., Bargteil A. W.: Physics-based animation of large-scale splashing liquids. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 185:1–185:6. 23

[GGG08] Guennebaud G., Germann M., Gross M. H.: Dynamic sampling and rendering of algebraic point set surfaces. *Computer Graphics Forum (CGF) 27*, 2 (2008), 653–662. 88

[GM77] Gingold R., Monaghan J.: Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Notices of the Royal Astronomical Society 181* (1977), 375–389. 5, 25, 29

[GP11] Goswami P., Pajarola R.: Time adaptive approximate SPH. In *Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS)* (2011), pp. 19–28. 62

[Gre09] Green S.: *Particle simulation using CUDA*. Tech. rep., NVIDIA, 2009. 23, 24, 36, 38

[Gre10] Green S.: *Screen space fluid rendering for games.* Tech. rep., NVIDIA, 2010. 87

[GSSP10] Goswami P., Schlegel P., Solenthaler B., Pajarola R.: Interactive SPH simulation and rendering on the GPU. In *ACM Symposium on Computer Animation (SCA)* (2010), pp. 55–64. 10, 24, 36, 37, 38, 65, 87

[HA06] Hu X. Y., Adams N. A.: A multi-phase SPH method for macroscopic and mesoscopic flows. *Journal of Computational Physics (JCP) 213*, 2 (2006), 844–861. 8, 29, 34, 108

[Har64] Harlow F.: The particle-in-cell computing method for fluid dynamics. *Methods in Computational Physics 3* (1964), 319–343. 22

[HCM06] Hegeman K., Carr N. A., Miller G. S. P.: Particle-based fluid simulation on the gpu. In *International conference on Computational Science (ICCS)* (2006), pp. 228–235. 38

[HE03] Hopf M., Ertl T.: Hierarchical splatting of scattered data. In *IEEE Visualization (VIS)* (2003), pp. 57–. 88

[HHK08] Hong W., House D. H., Keyser J.: Adaptive particles for incompressible fluid simulation. *Visual Computer 24*, 7 (2008), 535–543. 9, 88, 108

[HK89] Hernquist L., Katz N.: TREESPH: a unification of SPH with hierarchical tree method. *Astrophysical Journal Supplement Series 70* (1989), 419–446. 8

[HK05a] Hieber S. E., Koumoutsakos P.: A lagrangian particle level set method. *J. Comput. Phys. 210* (2005), 342–367. 30

[HK05b] Hong J.-M., Kim C.-H.: Discontinuous fluids. *ACM Trans. Graph. 24*, 3 (July 2005), 915–920. 21

[HKK07a] Harada T., Koshizuka S., Kawaguchi Y.: Sliced data structure for particle-based simulations on GPUs. In *Computer graphics and interactive techniques (GRAPHITE)* (2007), ACM, pp. 55–62. 38

[HKK07b] Harada T., Koshizuka S., Kawaguchi Y.: Smoothed particle hydrodynamics on GPUs. *Computer Graphics International* (2007), 63–70. 8, 24, 38

[HLS05] Hessel V., Löwe H., Schönfeld F.: Micromixers - a review on passive and active mixing principles. *Chemical Engineering Science Volume 60, Issues 8-9* (2005), 2479–2501. 3

[HLSR08] Hadwiger M., Ljung P., Salama C. R., Ropinski T.: Advanced illumination techniques for GPU volume raycasting. In *Proc. SIGGRAPH Asia* (2008). 10, 85, 86

[HLYK08] Hong J.-M., Lee H.-Y., Yoon J.-C., Kim C.-H.: Bubbles alive. *ACM Transactions on Graphics (TOG) 27*, 3 (2008), 48:1–48:4. 42

[How07] Howes L.: *Loading Structured Data Efficiently With CUDA*. Tech. rep., NVIDIA, 2007. 38

[HS13] Horvath C. J., Solenthaler B.: Mass preserving multi-scale sph. Pixar Technical Memo 13-04, Pixar Animation Studios, 2013. 62, 108

[IAAT12] Ihmsen M., Akinci N., Akinci G., Teschner M.: Unified spray, foam and bubbles for particle-based fluids. *The Visual Computer 28* (2012), 669–677. 10, 25, 42

[IABT11] Ihmsen M., Akinci N., Becker M., Teschner M.: A parallel SPH implementation on multi-core CPUs. *Computer Graphics Forum (CGF) 30* (2011), 99–112. 24, 36, 37, 38

[IAGT10] Ihmsen M., Akinci N., Gissler M., Teschner M.: Boundary handling and adaptive time-stepping for PCISPH. In *Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS)* (2010), pp. 79–88. 8, 24, 36, 62, 66, 68

[IBAT11] Ihmsen M., Bader J., Akinci G., Teschner M.: Animation of air bubbles with SPH. In *Proc. Int. Conf. Comp. Graph. Theory & App. (GRAPP)* (2011), pp. 225–234. 8, 42

[ICS*13] Ihmsen M., Cornelis J., Solenthaler B., Horvath C., Teschner M.: Implicit incompressible sph. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 20*, 3 (2013), 426–435. 24, 25, 35, 109

[IKC04] Ihm I., Kang B., Cha D.: Animation of reactive gaseous fluids through chemical kinetics. In *ACM Symposium on Computer Animation (SCA)* (2004), pp. 203–212. 21

[IOS*14] Ihmsen M., Orthmann J., Solenthaler B., Kolb A., Teschner M.: Sph fluids in computer graphics. In *Eurographics (State-of-the-Art Report)* (2014). 8, 13, 25, 35

[IUYTN10] Iwasaki K., Uchida H., Y.Dobashi, T. Nishita " (PG 2010) t. a.: Fast particle-based visual simulation of ice melting. *Computer Graphics Forum (CGF) 29*, 7 (2010), 2215–2223. 42

[JFSP10] Jang Y., Fuchs R., Schindler B., Peikert R.: Volumetric evaluation of meshless data from smoothed particle hydrodynamics simulations. In *Volume Graphics* (2010), pp. 45–52. 88

[KAG*05] Keiser R., Adams B., Gasser D., Bazzi P., Dutré P., Gross M.: A unified lagrangian approach to solid-fluid animation. In *Symposium on Point-based Graphics (SPBG)* (2005), pp. 125–133. 9, 34, 43

[KAG*06] Keiser R., Adams B., Guibas L. J., Dutré P., Pauly M.: *Multiresolution particle-based fluids*. Tech. rep., ETH, 2006. 9, 24, 36, 43, 62, 66, 68, 82, 88, 108

[KB04] Kobbelt L., Botsch M.: A survey of point-based techniques in computer graphics. *Computers and Graphics 28*, 6 (2004), 801–814. 85

[KBKS09] Kristof P., Benes B., Křivánek J., Stava O.: Hydraulic erosion using smoothed particle hydrodynamics. *Computer Graphics Forum (CGF) 28*, 2 (2009), 219–228. 9, 25, 33, 42

[KC05] Kolb A., Cuntz N.: Dynamic particle coupling for GPU-based fluid simulation. In *Symposium on Simulation Technique* (2005), pp. 722–727. 38, 62

[KCR08] Koumoutsakos P., Cottet G.-H., Rossinelli D.: Flow simulations using particles: bridging computer graphics and CFD. In *Proc. SIGGRAPH classes* (2008), pp. 25:1–25:73. 5, 8, 15, 24, 25, 26

[KFCO06] Klingner B. M., Feldman B. E., Chentanez N., O'Brien J. F.: Fluid animation with dynamic meshes. *ACM Transactions on Graphics (TOG) 25*, 3 (2006), 820–825. 22

[KHW*09] Knoll A., Hijazi Y., Westerteiger R., Schott M., Hansen C., Hagen H.: Volume ray casting with peak finding and differential sampling. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 15*, 6 (2009), 1571–1578. 88

[KJI07] Kang B., Jang Y., Ihm I.: Animation of chemically reactive fluids using a hybrid simulation method. In *ACM Symposium on Computer Animation (SCA)* (2007), pp. 199–208. 21

[KL07] Kim T., Lin M.: Stable advection-reaction-diffusion with arbitrary anisotropy. *Computer Animation and Virtual Worlds 18*, 4-5 (2007), 329–338. 21

[KLRS04] Kolb A., Latta L., Rezk-Salama C.: Hardware-based simulation and collision detection for large particle systems. In *Graphics Hardware (GH)* (2004), pp. 123–131. 38

[KmWH10] Kirk D. B., mei W. Hwu W.: *Programming massively parallel processors - a hands-on approach.* Morgan Kaufmann, 2010. 38

[Kou05] Koumoutsakos P.: Multiscale flow simulations using particles. *Annual review of Fluid Mechanics 37* (2005), 457–487. 8, 9

[KPR*06] Körner C., Pohl T., Rüde U., Thürey N., Zeiser T.: Parallel lattice boltzmann methods for cfd applications. In *Numerical Solution of Partial Differential Equations on Parallel Computers*. 2006, pp. 439–466. 22

[KPyNS10] Kang N., Park J., yong Noh J., Shin S. Y.: A hybrid approach to multiple fluid simulation using volume fractions. *Computer Graphics Forum (CGF)* (2010), 685–694. 22

[KSW04] Kipfer P., Segal M., Westermann R.: Uberflow: a GPU-based particle engine. In *Graphics Hardware* (2004), pp. 115–122. 38

[KTH*05] Körner C., Thies M., Hofmann T., Thürey N., Rüde U.: Lattice boltzmann model for free surface flow for modeling foaming. *Journal of Statistical Physics 121*, 1-2 (2005), 179–196. 22

[LAD08] Lenaerts T., Adams B., Dutré P.: Porous flow in particle-based fluid simulations. *ACM Transactions on Graphics (TOG) 27*, 3 (2008), 49:1–49:8. 42, 83, 109

[LAF11] Lentine M., Aanjaneya M., Fedkiw R.: Mass and momentum conservation for fluid simulation. In *ACM Symposium on Computer Animation (SCA)* (2011), pp. 91–100. 21

[LB95] Lapenta G., Brackbill J.: Control of the number of particles in fluid and mhd particle in cell methods. *Computer Physics Communications 87*, 1-2 (1995), 139 – 154. 66

[LC87] Lorensen W. E., Cline H. E.: Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph. 21*, 4 (Aug. 1987), 163–169. 43, 87

[LD09] Lenaerts T., Dutré P.: Mixing fluids and granular materials. *Computer Graphics Forum (CGF) 28*, 2 (2009), 213–218. 42, 63, 108, 109

[LFO06] Losasso F., Fedkiw R., Osher S.: Spatially adaptive techniques for level set methods and incompressible flow. *Computers and Fluids 35*, 10 (2006), 995–1010. 22

[LGF04] Losasso F., Gibou F., Fedkiw R.: Simulating water and smoke with an octree data structure. *ACM Transactions on Graphics (TOG) 23*, 3 (2004), 457–462. 22

[LGM*08] Ledergerber C., Guennebaud G., Meyer M. D., Bächer M., Pfister H.: Volume MLS ray casting. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 14*, 6 (2008), 1372–1379. 88

[LKO05] Liu J., Koshizuka S., Oka Y.: A hybrid particle-mesh method for viscous, incompressible, multiphase flows. *Journal of Computational Physics (JCP) 202*, 1 (2005), 65–93. 24

[LLRR08] Linsen L., Long T. V., Rosenthal P., Rosswog S.: Surface extraction from multi-field particle volume data using multi-dimensional cluster visualization. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 14*, 6 (2008), 1483–1490. 88

[LPA10] Lewis J. P., Pighin F., Anjyo K.: Scattered data interpolation and approximation for computer graphics. In *ACM*

SIGGRAPH ASIA 2010 Courses (2010), SA '10, pp. 2:1–2:73. 74

[LQB05] Lastiwka M., Quinlan N. J., Basa M.: Adaptive particle distribution for smoothed particle hydrodynamics. *International Journal of Numerical Methods in Fluids 46*, 10-11 (2005), 1403–1409. 9, 62

[LS94] Leimkuhler B., Skeel R.: Symplectic numerical integrators in constrained Hamiltonian systems. *Journal of Computational Physics (JCP) 112* (1994), 117–125. 36

[LSSF06] Losasso F., Shinar T., Selle A., Fedkiw R.: Multiple interacting liquids. *ACM Transactions on Graphics (TOG) 25*, 3 (2006), 812–819. 21

[LTKF08] Losasso F., Talton J., Kwatra N., Fedkiw R.: Two-way coupled SPH and particle level set fluid simulation. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 14*, 4 (2008), 797–804. 22, 42

[Luc77] Lucy L. B.: A numerical approach to the testing of the fission hypothesis. *Astronomical Journal 82* (1977), 1013–1024. 5, 25

[MBE*10] Misztal M. K., Bridson R., Erleben K., Bärentzen J. A., Anton F.: Optimization-based fluid simulation on unstructured meshes. In *Workshop on Virtual Reality Interaction and Physical Simulation (VRIPHYS)* (2010), pp. 11–20. 23

[MCG03] Müller M., Charypar D., Gross M.: Particle-based fluid simulation for interactive applications. In *ACM Symposium on Computer Animation (SCA)* (2003), pp. 154–159. 9, 23, 25, 28, 32, 34, 35, 38, 43, 52, 62, 81, 82, 87, 109

[MCP*09] Mullen P., Crane K., Pavlov D., Tong Y., Desbrun M.: Energy-preserving integrators for fluid animation. *ACM Transactions on Graphics (TOG) 28*, 3 (2009), 38:1–38:8. 21

[MCPN08] Molemaker J., Cohen J. M., Patel S., Noh J.: Low viscosity flow simulations for animation. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2008), SCA '08, Eurographics Association, pp. 9–18. 21

[MEB*12] Misztal M. K., Erleben K., Bargteil A., Fursund J., Christensen B. B., Bäerentzen J. A., Bridson R.: Multiphase flow of immiscible fluids on unstructured moving meshes. In *ACM Symposium on Computer Animation (SCA)* (2012), pp. 97–106. 23

[MM13] Macklin M., Mller M.: Position based fluids. In *ACM Transactions on Graphics (SIGGRAPH)* (2013). 24, 35, 38, 87

[MMTD07] Mullen P., McKenzie A., Tong Y., Desbrun M.: A variational approach to eulerian geometry processing. *ACM Trans. Graph. 26*, 3 (July 2007). 21

[Mon89] Monaghan J. J.: On the problem of penetration in particle methods. *Journal of Computational Physics (JCP) 82*, 1 (1989), 1–15. 8

[Mon92] Monaghan J. J.: Smoothed particle hydrodynamics. *Annual review of Astronomy and Astrophysics 30* (1992), 543–574. 8, 28, 29, 32, 36, 66, 82

[Mon94] Monaghan J. J.: Simulating free surface flows with sph. *Journal of Computational Physics (JCP) 110*, 2 (1994), 399–406. 24

[Mon00] Monaghan J. J.: Sph without a tensile instability. *Journal of Computational Physics (JCP) 159*, 2 (2000), 290–311. 8, 109

[Mon05] Monaghan J. J.: Smoothed particle hydrodynamics. *Reports on Progress in Physics 68* (2005), 1703–1759. 24, 25, 27, 32, 34, 66

[MSD07] Mueller M., Schirm S., Duthaler S.: Screen space meshes. In *ACM Symposium on Computer Animation (SCA)* (2007), pp. 9–16. 87

[MSKG05] Müller M., Solenthaler B., Keiser R., Gross M.: Particle-based fluid-fluid interaction. In *ACM Symposium on Computer Animation (SCA)* (2005), pp. 237–244. 8, 24, 25, 33, 42, 109

[NMK*06] Nealen A., Müller M., Keiser R., Boxerman E., Carlson M.: Physically based deformable models in computer graphics. *Computer Graphics Forum (CGF) 25*, 4 (2006), 809–836. 23

[NMM*06] Neophytou N., Mueller K., McDonnell K. T., Hong W., Guan X., Qin H., Kaufman A. E.: GPU-accelerated volume splatting with elliptical RBFs. In *IEEE TCVG Symposium on Visualization (Eurovis)* (2006), pp. 13–20. 86

[NVI11] NVIDIA: *NVIDIA CUDA C Programming Guide*, June 2011. 38

[OCD11] Onderik J., Chladek M., Durikovic R.: SPH with small scale details and improved surface reconstruction. In *Spring Conference on Computer Graphics (SCCG)* (2011). 10, 24, 42, 43, 44, 86, 87

[OF02] Osher S., Fedkiw R.: *Level set mMethods and dynamic implicit surfaces.* Springer-Verlag, 2002. 21, 45

[OHB*13] Orthmann J., H.Hochstetter, Bader J., Bayraktar S., Kolb A.: Consistent surface model for sph-based fluid transport. In *ACM Symposium on Computer Animation (SCA)* (2013), pp. 95–103. 13, 41

[OK12] Orthmann J., Kolb A.: Temporal blending for adaptive SPH. *Computer Graphics Forum (CGF) 31*, 8 (2012), 2436–2449. 13, 62

[OKK09] Orthmann J., Keller M., Kolb A.: Integrating gpgpu functionality into scene graphs. In *International Workshop on Vision, Modeling, and Visualization (VMV)* (2009). 13, 25, 38

[OKK10] Orthmann J., Keller M., Kolb A.: Topology-caching for dynamic particle volume raycasting. In *International Workshop on Vision, Modeling, and Visualization (VMV)* (2010), pp. 147–154. 13, 56, 85

[PGSS07] Popov S., Günther J., Seidel H.-P., Slusallek P.: Stackless KD-tree traversal for high performance GPU ray tracing. *Computer Graphics Forum (CGF) 26*, 3 (2007), 415–424. 88

[PH10] Pelfrey B., House D.: Adaptive neighbor pairing for smoothed particle hydrodynamics. In *Advances in Visual Computer* (Berlin, Heidelberg, 2010), Springer-Verlag, pp. 192–201. 24, 37, 38

[PK05] Park S. I., Kim M. J.: Vortex fluid for gaseous phenomena. In *ACM Symposium on Computer Animation (SCA)* (2005), pp. 261–270. 24

[PP04] Pilliod Jr. J. E., Puckett E. G.: Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *Journal of Computational Physics (JCP) 199*, 2 (2004), 465–502. 22

[PTC*10] Pfaff T., Thuerey N., Cohen J., Tariq S., Gross M.: Scalable fluid simulation using anisotropic turbulence particles. *ACM Transactions on Graphics (TOG) 29*, 6 (2010), 174:1–174:8. 21

[RBCK10] Rossinelli D., Bergdorf M., Cottet G.-H., Koumoutsakos P.: Gpu accelerated simulations of bluff body flows using vortex particle methods. *Journal of Computational Physics (JCP) 229*, 9 (2010), 3316–3333. 24

[REN*04] Rasmussen N., Enright D., Nguyen D., Marino S., Sumner N., Geiger W., Hoon S., Fedkiw R.: Directable photorealistic liquids. In *ACM Symposium on Computer Animation (SCA)* (2004), pp. 193–202. 22

[RK04] Rosen M. J., Kunjappu J. T.: *Surfactants and interfacial phenomena.* John Wiley & Sons, Inc, 2004. 3, 54

[RK08] Rossinelli D., Koumoutsakos P.: Vortex methods for incompressible flow simulations on the gpu. *Visual Computer 24*, 7 (2008), 699–708. 24

[RSKN08] Rungjiratananon W., Szego Z., Kanamori Y., Nishita T.: Real-time animation of sand-water interaction. *Computer Graphics Forum (CGF) 27* (2008), 1887–1893. 42

[RT04] Rude U., Thürey Nils N.: Free surface lattice-boltzmann fluid simulations with and without level sets. *International Workshop on Vision, Modeling, and Visualization (VMV)* (2004), 199–207. 22

[RWT11] Raveendran K., Wojtan C., Turk G.: Hybrid smoothed particle hydrodynamics. In *ACM Symposium on Computer Animation (SCA)* (2011), pp. 33–42. 23, 24

[SAC*99] Stora D., Agliati P.-O., Cani M.-P., Neyret F., Gascuel J.-D.: Animating lava flows. In *Graphics Interface* (1999), pp. 203–210. 33, 42, 109

[SB12] Schechter H., Bridson R.: Ghost SPH for animating water. *ACM Trans. Graph. 31*, 4 (2012), 61:1–61:8. 8, 24, 63, 83, 108

[SBH09] Sin F., Bargteil A. W., Hodgins J. K.: A point-based method for animating incompressible flow. In *ACM Symposium on Computer Animation (SCA)* (2009), pp. 247–255. 36

[SDG08] Stantchev G., Dorland W., Gumerov N. A.: Fast parallel particle-to-grid interpolation for plasma PIC simulations on the GPU. *Journal of Parallel and Distributed Computing (JPDC) 68*, 10 (2008), 1339–1349. 38

[SF95] Stam J., Fiume E.: Depicting fire and other gaseous phenomena using diffusion processes. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* (1995), pp. 129–136. 15

[SFK*08] Selle A., Fedkiw R., Kim B., Liu Y., Rossignac J.: An unconditionally stable maccormack method. *Journal of Scientific Computing 35*, 2-3 (2008), 350–371. 21

[SG11] Solenthaler B., Gross M.: Two-scale particle simulation. *ACM Transactions on Graphics (TOG) 30* (2011), 81:1–81:8. 10, 24, 62, 66, 67, 108

[She68] Shepard D.: A two-dimensional interpolation function for irregularly-spaced data. In *ACM National Conference* (1968), pp. 517–524. 30, 74

[SHG09] Satish N., Harris M., Garland M.: Designing efficient sorting algorithms for manycore GPUs. *Parallel and Distributed Processing Symposium* (2009), 1–10. 38, 88

[SHZO07] Sengupta S., Harris M., Zhang Y., Owens J. D.: Scan primitives for gpu computing. In *Graphics Hardware (GH)* (2007), pp. 97–106. 88

[SJ06] Shao S., Ji C.: Incompressible SPH simulation of wave breaking and overtopping with turbulence modelling. *International Journal for Numerical Methods in Fluids 50* (2006), 597–621. 24

[SKS05] Shaughnessy E. J., Katz I. M., Schaffer J. P.: *Introduction to fluid mechanics.* Oxford University Press, 2005. 3, 15, 53

[SP08] Solenthaler B., Pajarola R.: Density contrast SPH interfaces. In *ACM Symposium on Computer Animation (SCA)* (2008), pp. 211–218. 8, 24, 29, 42, 43, 82, 108, 109

[SP09a] Schlegel P., Pajarola R.: Layered volume splatting. In *International Symposium on Visual Computing (ISVC)* (2009), vol. 5876, pp. 1–12. 86

[SP09b] Solenthaler B., Pajarola R.: Predictive-corrective incompressible SPH. *ACM Transactions on Graphics (TOG) 28* (2009), 40:1–40:6. 24, 25, 35, 62, 80, 81, 82, 109

[SRF05] Selle A., Rasmussen N., Fedkiw R.: A vortex particle method for smoke, water and explosions. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 910–914. 21

[SS11] Stam J., Schmidt R.: On the velocity of an implicit surface. *ACM Transactions on Graphics (TOG) 30*, 3 (2011), 21:1–21:7. 43

[SSP07] Solenthaler B., Schläfli J., Pajarola R.: A unified particle model for fluid solid interactions: Research articles. *Journal of Computer Animation and Virtual Worlds (CAVW) 18*, 1 (2007), 69–82. 8, 10, 24, 42, 43, 86, 87, 109

[Sta99] Stam J.: Stable fluids. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* (1999), pp. 121–128. 15, 21

[Sus03] Sussman M.: A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *Journal of Computational Physics (JCP) 187*, 1 (2003), 110–136. 22

[SZP07] Solenthaler B., Zhang Y., Pajarola R.: Efficient refinement of dynamic point data. In *Symposium on Point-based Graphics (SPBG)* (2007), pp. 65–72. 43, 65

[THM*03] Teschner M., Heidelberger B., Müller M., Pomerantes D., Gross M. H.: Optimized spatial hashing for collision detection of deformable objects. In *International Work-*

shop on Vision, Modeling, and Visualization (VMV) (2003), pp. 47–54. 36

[TM05] Tartakovsky A., Meakin P.: Modeling of surface tension and contact angles with smoothed particle hydrodynamics. *Physical Review E 72* (2005). 34

[TPR*06] Thürey N., Pohl T., Rüde U., Oechsner M., Körner C.: Optimization and stabilization of lbm free surface flow simulations using adaptive parameterization. *Computers and Fluids 35*, 8 (2006), 934–939. 22

[TR09] Thürey N., Rüde U.: Stable free surface flows with the lattice boltzmann method on adaptively coarsened grids. *Computing and Visualization in Science 12*, 5 (2009), 247–263. 22

[TWGT10] Thürey N., Wojtan C., Gross M., Turk G.: A multiscale approach to mesh-based surface tension flows. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 48:1–48:10. 22

[vdLGS09] van der Laan W. J., Green S., Sainz M.: Screen space fluid rendering with curvature flow. In *Symposium on Interactive 3D Graphics and Games (i3D)* (2009), pp. 91–98. 10, 86, 87

[Wes90] Westover L.: Footprint evaluation for volume rendering. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* (1990), vol. 24, pp. 367–376. 86

[WH94] Witkin A. P., Heckbert P. S.: Using particles to sample and control implicit surfaces. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* (1994), pp. 269–277. 43

[WMT05] Wang H., Mucha P. J., Turk G.: Water drops on surfaces. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 921–929. 21

[WTGT10] Wojtan C., Thürey N., Gross M., Turk G.: Physics-inspired topology changes for thin fluid features. *ACM Transactions on Graphics (TOG) 29* (2010), 50:1–50:8. 22, 42, 43

[YT13] Yu J., Turk G.: Reconstructing surfaces of particle-based fluids using anisotropic kernels. *ACM Transactions on Graphics (TOG) 32*, 1 (2013), 5:1–5:12. 10, 24, 43, 48, 86, 87

[YWTY12] Yu J., Wojtan C., Turk G., Yap C.: Explicit mesh surfaces for particle based fluids. *Computer Graphics Forum (CGF) 31* (2012), 815–824. 9, 42, 43

[ZB05] Zhu Y., Bridson R.: Animating sand as a fluid. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 965–972. 10, 22, 42, 43

[ZGHG10] Zhou K., Gong M., Huang X., Guo B.: Data-parallel octrees for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* (2010). 10, 36, 37, 88

[ZHWG08] Zhou K., Hou Q., Wang R., Guo B.: Real-time KD-tree construction on graphics hardware. *ACM Transactions on Graphics (TOG) 27*, 5 (2008), 126ff. 10, 88

[ZLC*13] Zhu B., Lu W., Cong M., Kim B., Fedkiw R.: A new grid structure for domain extension. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 63:1–63:12. 22

[ZPvBG01] Zwicker M., Pfister H., van Baar J., Gross M.: Surface splatting. In *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)* (2001), pp. 371–378. 87

[ZSP08] Zhang Y., Solenthaler B., Pajarola R.: Adaptive sampling and rendering of fluids on the GPU. In *Symposium on Point-based Graphics (SPBG)* (2008), pp. 137–146. 24, 38, 62, 88, 108

Scene	Fig.	**"# Ptcls"**	**"Domain Size"**	**"Parameters"**
Bunny	1.1, 6.11, 6.1, 1.4	$4.7M$	$280 \times 170 \times 210$	$\sigma_r = 40, \sigma = 3.5$
T-Sensor	1.2	$1.5M$	$1020 \times 575 \times 15$	$\sigma_a = 1, \sigma_d = 0.01, \sigma_s = 1, \sigma_r = 10, \sigma = 10$
Wetting	1.2	$0.9M$	$110 \times 110 \times 110$	$\sigma_a = 10, \sigma_d = 1, \sigma_s = 10, \sigma_r = 40, \sigma = 3.5, \mu_r = 0.001 - 0.05, \alpha = 0.01 - 1.0$
Dye	3.4, 4.7	$1.1M$	$180 \times 120 \times 120$	$h = 1/2, \sigma_a = 0/10, \sigma_d = 0/1, \sigma_s = 10, \sigma_r = 10, \alpha = 0.01 - 1.0$
Pan	4.1	$2.1M$	$170 \times 170 \times 170$	$\sigma_a = 10, \sigma_d = 0.1, \sigma_s = 10, \sigma_r = 10, \sigma = 3.5, \mu_r = 0.001 - 0.05, \alpha = 0.01 - 1.0$
Rig. Flooding	1.6, 4.4, 4.10, 6.5	$1.1M$	$220 \times 180 \times 130$	$\sigma_a = 10, \sigma_d = 0.1, \sigma_s = 10, \sigma_r = 70, \sigma = 3.5, \mu_r = 0.001 - 0.05, \alpha = 0.01 - 1.0$
Kinect®	4.12	$0.75M$	$260 \times 260 \times 190$	$\sigma_a = 10, \sigma_d = 0.1, \sigma_s = 10, \sigma_r = 40, \sigma = 3.5, \mu_r = 0.001 - 0.05, \alpha = 0.01 - 1.0$
Lena	6.9, 6.10	$3M$	$340 \times 120 \times 450$	$\sigma_a = 10, \sigma_d = 1, \sigma_s = 20, \sigma_r = 40, \sigma = 3.5, \mu_r = 0.001 - 0.05, \alpha = 0.01 - 2.0$
Val. Flooding	1.5, 5.1, 5.8	$0.5 - 1M$	$220 \times 180 \times 130$	$\Delta t_{\min} = 40, \Delta t_{\max} = 200, E_{\eta,\max} = 0.06, l_{\max} = 3$
Teapot	5.3	$0.47 - 1M$	$130 \times 130 \times 80$	$\Delta t_{\min} = 40, \Delta t_{\max} = 200, E_{\eta,\max} = 0.06, l_{\max} = 3$
Pillars	5.4, 5.6, 5.9	$0.2 - 1.5M$	$220 \times 170 \times 100$	$\Delta t_{\min} = 40, \Delta t_{\max} = 200, E_{\eta,\max} = 0.06, l_{\max} = 6$

Table A.1: Parameters for all scenes presented in this thesis.

A.1 Test Scenes

This section provides configurations and parameter settings of all scenarios presented in this thesis. In all cases, the support radius has been set to twice the particle radius, i.e. $h = 1$. In combination with the Poly-6 and Spiky kernel, the particle's rest distance on average was $x = 0.92h$, which resulted in about 30 neighbours per particle. All scenarios used an estimated rest density of water $\rho_0 = 1000[kg\,m^{-3}]$. In order to maintain incompressibility, the pressure constant has been computed to $\beta \approx 0.5$, which for time step conditions $\lambda_u = 0.4, \lambda_F = 0.25$ resulted in an average of three iterations. In order to ensure stability, if not stated otherwise, artificial viscosity constant and boundary damping constant have been set to $\mu = 0.05$ and $\mu_r = 0.005$ respectively. The surface tension constant has been set to $\alpha = 1.0$. The acceleration due to gravity has been set to $9.81[m\,s^{-2}]$. Surface parameters for reference configurations as depicted in fig. 4.8 and $h = 1$ have been $A_{\min} \approx 0.1$, $\delta_{\max} \approx 0.65$ and $\epsilon_c \approx 40$. Other scenario specific parameters are summarized in Tab. A.1 and are described in the respective result sections of this thesis.

List of Figures

List of Tables