Universidad
Rey Juan Carlos

Tesis Doctoral

Simulation and Haptic Rendering of the Interaction with Diverse Fluid Media

Author Héctor Barreiro Cabrera

Supervisor Miguel A. Otaduy Tristán

Programa de Doctorado en Tecnologías de la Información y las Comunicaciones

Escuela Internacional de Doctorado

Madrid, 2021

Abstract

Virtual reality (VR) technologies are committed to the development of solutions that enable the long-standing vision of creating immersive synthetic worlds that transcend the boundaries of reality. Although VR is at the peak of its history thanks to the confluence of various technologies developed over decades, the diversity of interactions that may be portrayed in VR experiences remains limited.

While interaction with solid objects and deformable bodies has attracted a great attention from researchers, other interesting media, such as fluids, have been largely ignored. In the particular case of fluids, this is primarily due to a combination of two factors. First, interactive fluid simulation methods are incapable of conveying materials other than inviscid fluids. This severely limits the capacity to replicate fascinating everyday materials such as honey, whipped cream, paint, or clay. Second, conventional haptic devices struggle to provide compelling contact with fluid media, particularly in applications requiring direct manipulation.

In this thesis, we investigate strategies to overcome the limitations of the current state of the art in order to enable physical contact with rich and complex virtual phenomena such as fluids. This is accomplished in two ways. First, we present a highly efficient constraint-based method for viscoelastic fluid modeling. Our approach is motivated by a constitutive model for polymeric fluids, which enables the portrayal of a wide variety of materials under a single formulation. Second, we present two methods for depicting tactile interaction with such media by leveraging on the AM and STM control metaphors commonly employed in ultrasonic haptics. We approach the device actuation as a numerical optimization problem, finding the control variables that best reproduce the pressures arising from virtual interactions. Furthermore, we incorporate knowledge of the technical and perceptual constraints of both control metaphors to maximize the efficacy of our solutions.

To conclude, we demonstrate the applicability of the presented approaches, combining them to address the challenge of virtual simulation of clay interaction. As a

result, our method enables unprecedented degree of realism in natural manipulation of materials exhibiting extreme viscoplastic behavior.

Acknowledgements

This thesis concludes one of the longest and most challenging chapters of my life. At the risk of sounding *cliché*, I am certain that I would not have gotten here without the help of many friends and loved ones that helped shape me into the person I am today. To each one of you, even if you're not on this list, thank you.

To *Miguel Ángel*, thank you for your tremendous patience and wisdom. By giving me the opportunity to work with you, you have allowed me to discover a world that I was completely unaware of. Thank you for all of your hard work and commitment in bringing all these projects to completion.

To *Iván*, *Ignacio* and *Stephen*, thank you for being so generous as to assist and support me during the development of these projects. You gave me the opportunity to learn a great deal from you. To all the people at *MSLab* and *Next Limit Technologies*. To *William* and *Maurizio*, for giving me the opportunity to work with you during my internships at *Ultraleap* and *Facebook* Reality Labs respectively. It's been a real pleasure working with all of you.

To the members of my Ph.D. committee, thank you for evaluating this thesis. To the reviewers, thank you for providing feedback for our publications.

To the organizations and institutions that have supported this thesis. This work was funded in part by the European Research Council (ERC Consolidator Grant 772738 TouchDesign), the Spanish Ministry of Economy (grant TIN2015-70799-R Simverso) and the Spanish Ministry of Science (grant RTI2018-098694-B-I00 VizLearning).

To the mentors who have influenced my life throughout the years, you have opened the doors of knowledge for me, and inspired me with the desire to continue learning and taking on new challenges.

To my father, *Francisco*, my brothers and family. Your kindness and patience helped me overcome all of the difficulties that stood in my way, allowing me to arrive to where I am now. I love you.

To all of my friends, old and new, who have always been there for me. I owe you some of the most wonderful experiences of my life. It would have been a lot more boring without you. Thank you for being there; you are part of my family as well. I adore all of you.

To *Rosa*, who has brought light into my life for more than ten years now. We were fortunate to embark on this adventure together, and now we can complete it together. You have always been there for me, both in the good times and the bad. I will be eternally grateful for your love, patience and unwavering support since we began our life together. I love you more than anything in the world.

To you, dear reader, because you are spending your valuable time reading this thesis, the fruit of years of hard work. I hope you find it interesting.

Finally, to my mother, *Araceli*, whom I never stopped thinking about day after day. I dedicate this thesis and all future endeavors to you. I am certain you would be proud of them. I shall cherish the memories of your love and kindness for the rest of my life.

Contents

List of Figures

List of Tables

1 Introduction

Since the advent of computing, researchers have long dreamed of harnessing its possibilities for building synthetic worlds where users could engage in immersive experiences transcending the boundaries of reality. Virtual reality (VR) technologies embrace this challenge and are devoted to the development of solutions towards the achievement of this long-standing dream.

Today, VR is at the peak of its history. The convergence of numerous technologies developed over the past few decades, such as head-mounted displays, motion trackers, and haptic devices, enables the generation of multisensory stimuli that provide an unprecedented level of immersion and embodiment. Furthermore, the emergence of commercial consumer-grade solutions has sparked renewed interest from the industry, academia and the general audience. The rapid evolution of these technologies, both in fidelity and economic terms, shapes the horizon of VR and positions it as one of the fastest growing industries of the coming years.

However, despite technological advancements, the variety of interactions that can be portrayed in VR experiences remains limited. The reason behind this lies in the computational models used to describe the interaction between the user and the virtual environment. While many works leverage on the use of physics-based models for achieving natural and plausible interactions with virtual objects, most of these focus primarily on the manipulation of rigid solids and, to a lesser extent, deformable bodies. Researchers paid relatively little attention to the representation of other ubiquitous and interesting media.

Fluids are a prime example of such media. They constitute a fascinating medium whose complex and visually rich behaviors pervades many facets of our daily life. Mundane and creative activities such as drinking water, pouring honey on toast, painting a picture with your hands or modeling clay into unique shapes; are all the consequence of interactions with one or more fluid substances (cf. Fig. 1.1).

Fig. 1.1.: Examples of interactions with fluid phenomena.

It is not surprising that VR applications could find significant appeal in naturally reproducing such interactions in virtual worlds.

However, representing such richness comes at a high computational cost. While methods exist for visually simulating some types of fluids at interactive rates (e.g. liquids and gases), attractive materials such as highly viscous or viscoelastic fluids, as well as viscoplastic substances such as clay, remain an open challenge. Solving the equations of motion that govern their behavior commonly requires the integration of numerically stiff differential equations, imposing severe constraints on the size of the integration step in order to avoid numerical instabilities. Moreover, the complexity of the system scales proportionally to the desired level of detail. As a result, representing high frequency details, which is where the most visually appealing behavior resides, usually conflicts with the computational budget constraints of interactive applications such as VR.

Alongside visual simulation, haptic feedback is critical for offering immersive interaction experiences to users. However, traditional haptic devices have a difficult time providing compelling interaction with fluid media. While tool-based devices are suitable for some fluid contact applications, they fall short of adequately expressing these in scenarios where direct manipulation is desired (e.g. clay modeling). In the recent years, haptic display technologies that enable the production of direct tactile sensations on the skin in mid-air (i.e. without the need to carry or wear a haptic device) have arisen. These technologies have the potential of being a good match for displaying such interactions, as they are capable of producing stimuli without restricting the user's motion. However, given the inmaturity of these technologies, it is unclear how the interactions occurring in the virtual fluid can be mapped to device actuation, therefore remaining as an open challenge.

In this thesis, we address these challenges and explore ways to push the boundaries of current state of the art in VR interactions with fluids. To accomplish this, we focus on two interesting challenges: the creation of novel models for the simulation of viscoelastic and viscoplastic fluids suitable for real-time interaction, and the development of various haptic rendering algorithms that enable users to interact naturally with this kind of media.

1.1 Fluid Simulation in Computer Graphics

Fluid simulation has been one of the most active research topics in the field of computer graphics for the last three decades, giving rise to a vast amount of work dedicated to modeling attractive fluid phenomena for a wide variety of applications such as feature films, commercials, video games or medical simulation. However, despite its widespread application, the simulation of fluid phenomena continues to be a difficult task. While fluid mechanics is nowadays fairly well understood, numerically describing its motion requires the resolution of computationally challenging nonlinear differential equations. As a result, researchers have historically constrained their simulations to the representation of surface phenomena for settings where interactivity is required, relegating the simulation of fluid volumes to offline applications.

Over the years, computer graphics researchers have concentrated on carefully developing approximations that significantly reduce the computational cost associated with the dynamic simulation of specific types of fluids. Advancements such as low-dissipation stable advection schemes, constrained dynamics solvers to guarantee fluid incompressibility, and advanced numerical approaches such as multigrid techniques with sophisticated boundary condition treatment; have resulted in the development of high-performance and high-resolution fluid dynamics solvers. These advancements, together with the popularization of Graphic Processing Units (GPUs) as massively parallel computational tools, has enabled the development of methods suitable for their application in interactive settings (Crane et al., 2007; Macklin & Müller, 2013).

Fig. 1.2.: While real-time fluid simulation methods exist, the range of behaviors they are able to model is limited. **Source:** Macklin and Müller, 2013.

However, these simulation methods have primarily focused on the representation of Newtonian low-viscosity incompressible fluid materials such as water (Fig. 1.2). Many everyday substances, such as honey, ketchup, whipped cream, and clay; exhibit interesting highly viscous, viscoelastic, or non-Newtonian (i.e. whose viscosity is shear rate or history dependent) behavior that cannot be reproduced under such formulations. Therefore, to accomplish the objective of broadening the possibilities of fluid interaction in VR, it would be interesting to find methods that enabled the simulation of such materials in real-time.

Unsurprisingly, simulation of high viscosity poses additional computational challenges. Implicit formulations are required to robustly solve the numerically stiff differential equations. In addition, due to the difficulty in computing the strain of a fluid, numerical drift turns into perceptible loss of viscoelasticity. Some works have explored the modeling of moderate to high viscosity materials for interactive applications (Alduán et al., 2017; Macklin & Müller, 2013; T. Takahashi et al., 2016; T. Takahashi et al., 2014). However, they are limited to modeling Newtonian viscosity, and fail to achieve extreme viscous behavior without introducing artifacts in the form of excessive drift or undesired elastic oscillations.

In Chapter 3 of this thesis, we address this challenge and propose a method for the simulation of highly viscous and viscoelastic fluids that is suitable for interactive applications. The key behind its high performance lies in the use of constrained dynamics solvers as an alternative to implicit formulations to describe these materials. Our solution is inspired by a constitutive model of polymeric fluids (i.e. fluids where elastic polymers are dissolved), supporting a large range of viscoelasticity behaviors under one common formulation.

Although the parameters of our model can be artistically tuned to phenomenologically portray viscoelastic fluids that are not strictly characterized under our polymeric model, it is insufficient to represent materials exhibiting viscoplastic behavior such as clay. In Chapter 6 we address this problem by presenting a method for the simulation of viscoplastic materials. We leverage on a simplified version of the method presented in Chapter 3, augmented with an elastoplasticity model to capture the main features of clay-like materials.

1.2 Mid-air Haptic Rendering

While our senses allow us to perceive and comprehend different facets of the reality in which we live, touch is the only sense that binds us to the world. It enables human beings to engage with and manipulate their surroundings physically, revealing features of objects that are not usually discernible through other senses: shape, compliance, roughness, texture, and warmth, for example. However, its study and understanding (haptics) falls far behind the degree of comprehension of other senses such as sight or hearing.

Aligned with the current renaissance of virtual reality (VR), haptic devices (i.e., technology that stimulates users' sense of touch) have also grown rapidly in popularity due to the significant benefits they provide to the human-computer interaction (HCI) experience. As a result, we have witnessed the emergence of novel haptic technologies that employ a variety of actuation principles to enable convincing virtual touch directly with our hands.

Mid-air haptic displays are a promising example of such emerging technologies. Volumetric displays, together with hand-tracking technologies, enable users to experience tactile stimuli directly in mid-air, freeing them from the constraints of other display technologies such as contact surfaces or wearable devices, while significantly expanding their workspace. Although researchers have developed displays relying on a variety of physical phenomena (e.g., air flows (Tsalamlal et al., 2014), lasers (Ochiai et al., 2016) or electric arcs (Spelmezan et al., 2016)), those based in ultrasonic phased arrays have gained the most popularity in recent years due to their low latency, large stimulus size, and ample workspace.

Nonetheless, the generation of tactile percepts with these devices remains a largely unknown process due to the absence of a computational model that maps activation patterns to perception. Researchers have focused primarily on the representation of holographic objects, developing different high-level metaphors to command these ultrasound devices for such purposes. However, these command metaphors by themselves are insufficient for portraying fluid interactions.

When we interact with fluids, our skin is subjected to a temporally and spatially variable pressure field whose characteristics are determined by our motion and the inherent properties of the flow. As such, tactile interaction with such media utilizing ultrasonic haptics could be posed as the problem of dynamically reproducing a pressure field on the user's skin. To date, no command metaphor enjoys the ability to directly reproduce an arbitrary spatially varying pressure field.

In Chapters 4 and 5 of this thesis, we address this challenge and propose several methods for the tactile rendering of fluid media using ultrasonic phased arrays. We approach the rendering problem as a dynamic mapping of target pressure fields to control metaphors. By formulating this mapping as an optimization problem, we are able to account for the known perceptual and technical limitations of the various control metaphors, resulting in the best-match reconstruction of the target pressure field. Moreover, we achieve efficient solutions by relying on the phenomenon of the persistence of tactile perception, which allows to simplify the problem by neglecting the temporal component of the actuation over a short time window (i.e. by assuming a *quasi-static* pressure field).

The method presented in Chapter 4 is later extended in Chapter 6 for the rendering of clay interaction, providing a more efficient use of the limited device resources by biasing the optimization problem solutions based on perceptual weight maps.

1.3 Overview and Contributions

The main contributions of this thesis can be summarized as follows:

- In Chapter 3 we introduce a constraint-based model of viscoelasticity. We describe a constitutive model of viscoelasticity in polymeric fluids, which is

the stress-based counterpart for our constraint-based formulation. Then, we describe the derivation of implicit conformation constraints acting on fluid velocities, as well as the parameters of our model.

- In Chapter 3 we propose a *doubly constrained* position-based dynamics (DC-PBD) solver, which inherits the robustness and efficiency of the original PBD method, but exhibits improved stability under velocity-based constraints, such as those in our viscoelasticity formulation, specially with large time steps.

- In Chapter 3 we discuss the phenomenological application of our approach to the representation of a variety of materials ranging from highly viscous to viscoelastic, as well as practical implementation details for achieving high-performance simulations while minimizing numerical artifacts that result in undesired kinetic dissipation.

- In Chapter 4 we introduce an efficient algorithm for ultrasound rendering of tactile interaction with fluids based on the *amplitude modulation* (AM) command metaphor. We characterize the actuation of the device using a set of focal points, optimizing the location and intensity of such focal points to best approximate the pressure field on the skin. The key to the efficiency of our solution is the assumption that the rendered pressure locally depend only on the closest focal point, which enables the decoupling of location and intensity into separate optimization problems. The resulting method produces a responsive experience while dynamically interacting with a virtual fluid.

- In Chapter 5 we propose another efficient rendering algorithm based on the *spatiotemporal modulation* (STM) command metaphor, which characterizes device actuation through the control of paths of focal points. We propose a two-level path routing optimization to render the force distribution resulting from a dynamic virtual interaction. A key aspect of our method is to pose STM rendering as a quasi-static problem, which eliminates the temporal variable on each dynamic rendering update, thus making the problem computationally tractable.

- Chapter 5 also compares the reconstruction quality of the method with respect to the algorithm presented in Chapter 4, observing that our STM-based algorithm succeeds to provide larger and smoother coverage than the AM-based method, while exhibiting superior performance in discrimination tasks.

- Finally, in Chapter 6, we propose a computational solution for the interactive simulation of clay-like materials with unprecedented realism, coupled with free-air tactile rendering that provides a natural tangible experience. Our solution extends the methods proposed in Chapters 3 and 4 by including a novel model of elastoplasticity and an optimization formulation that also accommodates to perceptual weight maps respectively. Our algorithm takes as input the interaction forces between a virtual hand model and the clay-like material. We demonstrate the effectiveness of our method through expressive creative experiences.

1.4 Publications

The following chapters are based on the following publications:

- Chapter 3: **Conformation Constraints for Efficient Viscoelastic Fluid Simulation** - Héctor Barreiro, Ignacio García-Fernández, Iván Alduán and Miguel A. Otaduy - *ACM Transaction on Graphics (Proceedings of ACM SIGGRAPH Asia), 2017* (Barreiro et al., 2017).

- Chapter 4: **Ultrasound Rendering of Tactile Interaction with Fluids** - Héctor Barreiro, Stephen Sinclair and Miguel A. Otaduy - *Proceedings of World Haptics Conference, 2019* (Barreiro et al., 2019).

- Chapter 5: **Path Routing Optimization for STM Ultrasound Rendering** - Héctor Barreiro, Stephen Sinclair and Miguel A. Otaduy - *IEEE Transactions on Haptics, 2020* (Barreiro et al., 2020).

- Chapter 6: **Natural Tactile Interaction with Virtual Clay** - Héctor Barreiro, Joan Torres and Miguel A. Otaduy - *Proceedings of World Haptics Conference, 2021* (Barreiro et al., 2021).

1.5 Outline

Chapter 2. Background.

In this chapter we will make a bibliographic review of the previous works that have motivated this thesis. We have divided this chapter in three main blocks: *Computational Fluid Dynamics and Simulation Methodologies*, that reviews the relevant works in fluid simulation; *Ultrasound Haptics*, that describes the literature of ultrasound-based haptic rendering; and *Coupling Fluids and Haptics*, that addresses the coupling between the two previous ones.

Chapter 3. Simulation of Viscoelastic Fluids.

This chapter introduces our constraint-based model of viscoelasticity. We start with a description of a constitutive model of viscoelasticity in polymeric fluids. Then, we describe in detail the derivation of our implicit conformation constraints acting on fluid velocities, as well as the parameters of the proposed model. Finally, we review the practical implementation details of our approach, introducing our DC-PBD solver; and showcase the method capabilities for the representation of a variety of phenomenologically-reproduced materials.

Chapter 4. Ultrasound Rendering of Fluids through Amplitude Modulation.

This chapter describes our novel algorithm for ultrasound rendering of tactile interaction with fluids based on the *amplitude modulation* command metaphor. We also present an efficient solver especially designed for this optimization problem, and we show results of interactive experiments with several fluid simulations.

Chapter 5. Ultrasound Rendering of Fluids through Spatiotemporal Modulation.

In this chapter, we study the problem of rendering interactions with virtual environments under the *spatiotemporal modulation* command metaphor. We detail the novel optimization approach we propose, which also takes into account known

perceptual parameters and limitations of the STM method. Finally, we compare its performance against the method presented in Chapter 4 in the context of fluid interaction rendering through a user discrimination task.

Chapter 6. Natural Interaction with Virtual Clay.

This chapter bridges our works in fluid simulation and haptic rendering, describing our approach for the natural interaction with virtual clay. We describe the characteristics of clay-like materials and then we introduce the multiple extensions to the methods presented in Chapters 4 and 6 to account for their characteristic behavior. Finally, we showcase the effectiveness of our simulation model and rendering algorithm on several examples of creative experiences with complex and rich clay material.

Chapter 7. Conclusions.

This chapter gives a final overview of the proposed methods and discusses the limitations as well as the different opportunities for future work.

2 Background

Reaching the ambitious goal of rendering interactive fluids (both in the simulation and haptic senses) is only possible through the confluence of the research efforts of the computer graphics and haptics communities. These efforts, however, are the results of achievements and innovations that spanned the course of decades, resulting in an enormous body of literature that is impossible to cover in a mere few pages. Nonetheless, in this chapter we strive to provide a comprehensive assessment of the state of the art, while placing a particular emphasis on those works that serve as the foundation for the development of this thesis.

In Section 2.1, we describe the most relevant works in fluid simulation, outlining the many strategies typically employed in the Computer Graphics literature while focusing on those approaches that enable high-performance fluid dynamics. Our discussion begins with the modeling of low-viscosity incompressible fluids, which are the most commonly portrayed fluid phenomena in graphics. Then we steer on to the particularities and considerations required for the simulation of more interesting media, concluding with a thorough examination of the works that address the simulation of viscoelastic fluid materials.

In Section 2.2, we review the literature on ultrasound-based haptic rendering. We begin with a description of the fundamental concepts and principles behind these technologies and then proceed onto describing the various techniques and metaphors used to command their actuation, as well as their limitations and perceptual implications.

Finally, and to conclude, in Section 2.3 we provide an overview of the works addressing the coupling between fluid simulation methods and haptics devices for the rendering of virtual interactions.

2.1 Computational Fluid Dynamics and Simulation Methodologies

The modeling of complex fluid phenomena such as liquids, gases and, to a lesser extent, plastic solids; has attracted the interest of researchers from a wide range of disciplines over the years. Fluids are ubiquitous to a broad range of engineering challenges: mechanical, civil, chemical, biomedical, and so on.

However, many fluid mechanics problems (especially those involving fluid dynamics) are difficult to solve analytically due to their complexity. Only partial solutions exist for very specific subsets of problems. Furthermore, designing and constructing laboratory experiments involving fluids under controlled conditions is both challenging and costly, requiring great effort to ensure the quality of the results. It is not surprising that scientists have focused their efforts on developing tools that leverage on computing and modern numerical methods for finding accurate solutions to such problems. The discipline of fluid mechanics devoted to the development of such tools is known as Computational Fluid Dynamics (CFD).

Computer graphics researchers are no strangers to CFD. Fluid simulation has been one of the most active research topics in the field for the past three decades, attracting hundreds of researchers and producing a vast amount of work dedicated to modeling appealing fluid phenomena for a variety of applications. Feature films, commercials, video games or medical simulation are just a few of the applications where fluid simulations are extensively used. However, while CFD strives for high-accuracy/low-performance solutions, computer graphics applications are often willing to sacrifice such numerical accuracy for the sake of increased computational speed. As a result, much of the research in the field has focused on developing careful approximations to produce faster algorithms while improving or preserving the visual quality of the simulations produced.

The common ground for most interesting fluid flows modeled in computer graphics are the Navier-Stokes equations (NS) (Bridson, 2015). These equations represent a reformulation of Newton's Second Law for fluid substances. In the case of an

incompressible Newtonian fluid (i.e. with constant density ρ and kinematic viscosity μ), the flow evolution is described by:

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \mu \nabla^2 \mathbf{u} + \rho \mathbf{g}, \tag{2.1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2.2}$$

where the unknowns $\mathbf{u}$ and p are the velocity and pressure of the fluid respectively. Equation (2.1), also known as momentum equation, describes the evolution of the fluid due to the interaction of both internal and external forces in terms of the material derivative $D\mathbf{u}/Dt$. Equation (2.2), known as continuity equation, ensures the conservation of mass (i.e. the fluid's incompressibility condition).

The momentum equation can be further broken down into three distinct terms.

- **Pressure.** The first one is the pressure term (∇p), which models the forces arising from imbalances in internal pressures throughout the fluid. Higher pressure regions will flow towards lower pressure ones. In the particular case of an incompressible fluid, this may be deemed as the force required to satisfy the continuity equation.

- **Diffusion.** The second one is the diffusion term ($\mu \nabla^2 \mathbf{u}$) which models viscosity. Viscous fluids oppose to shearing deformations, which manifest as local differences in velocity that can be measured through the Laplacian differential operator (∇^2). In the special case where $\mu = 0$, the NS equations simplify into the so-called Euler equations.

- **External body forces.** Finally, the third term are the external body forces ($\rho \mathbf{g}$), which model external interactions such as gravity or contact.

With appropriate initial and boundary conditions, these equations can be discretized and solved. The exact method used for this, however, is tied to the choice of fluid flow description. In mechanics, there are two traditional approaches or viewpoints for tracking the motion of a moving fluid or deformable solid (continuum): The Lagrangian viewpoint and the Eulerian viewpoint.

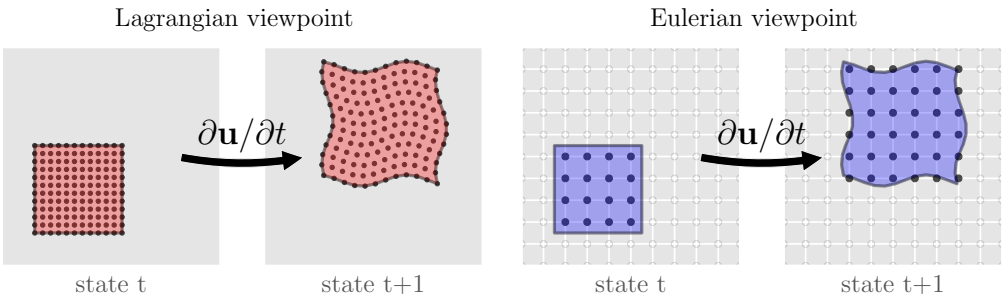


Fig. 2.1.: In the Lagrangian viewpoint **(left)** the motion of individual fluid elements is tracked as the fluid flows and evolves over time. In the Eulerian viewpoint **(right)** fluid properties are tracked at fixed points in space (e.g. in a grid) as the volume flows through them.

The Lagrangian approach treats the continuum as a particle system. This way, each material point of the fluid (or solid) is labeled as an independent particle, tracking their evolution over time. The Eulerian approach, however, fixates on particular points in space and records the evolution of the properties of the fluid elements passing through them. Fig. 2.1 illustrates a visual comparison between the two descriptions.

Both approaches have been effectively used in the computer graphics literature. The works presented in Chapters 3 and 6 of this thesis employ a Lagrangian discretization for the representation of highly viscous and viscoelastic materials, whereas the works presented in Chapters 4 and 5 employ an Eulerian discretization for the simulation of gaseous media.

In the following sections we will provide a more in-depth overview of approaches based on both perspectives. Section 2.1.1 summarizes methods based on the Eulerian viewpoint, while Section 2.1.2 covers those based on the Lagrangian viewpoint. Finally, in Section 2.1.3, we will briefly discuss other methods based on hybrid schemes that succeeded in compensating the limitations of both perspectives.

2.1.1 Eulerian Methods

Most of the early computer graphics literature on fluid simulation concentrated on the challenge of solving the Navier-Stokes equations from the Eulerian perspective.

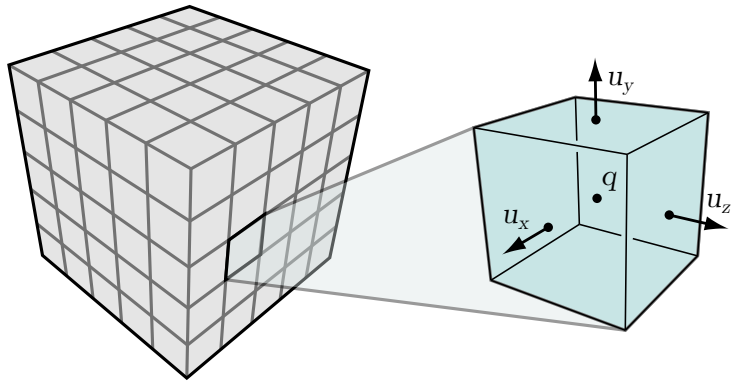


Fig. 2.2.: Typical discretization scheme for Eulerian fluids in computer graphics. The simulation domain $\mathcal{D}$ is partitioned into cells of size $C \in \mathbb{R}^3$. Fluid quantities q (e.g. pressure, ink, etc.) are typically stored at the center of the cell, whereas velocity components u_x, u_y, u_z are stored at the cell faces. This scheme enables a robust estimation of second order derivatives.

The seminal works of Foster et al. (Foster & Fedkiw, 2001; Foster & Metaxas, 1996) and Stam (2009) laid the foundations of the approximations commonly found in graphics.

Typically, Eulerian methods follow a Chorin-style advection-projection scheme (Chorin, 1968). The success of such schemes lies, at their core, on the use of *operator-splitting* approximations. By dividing the partial differential equation (PDE) to solve into multiple independent parts, splitting methods allow the computation of individual solutions which, when combined, form an approximated solution to the original equation up to first order accuracy.

In the particular case of the Navier-Stokes equations, operator splitting is primarily used to decouple the pressure term of the momentum equation (2.1) from the remaining inertial and internal forces, simplifying the overall process while enabling high-performance solutions. Within this scheme, the robustness, accuracy and performance of the resulting simulation depend primarily on three key ingredients: the choice of discretization, the advection scheme and the pressure projection strategy employed.

Discretization

Though Eulerian methods can be expressed over arbitrary meshes, grid discretization approaches are the most frequent solution used in computer graphics. They enable easy computation of spatial derivatives through finite differences, significantly simplifying the implementation process.

The simulated fluid domain Ω is partitioned into cells of size $C \in \mathbb{R}^3$. Fluid field quantities q such as pressure, temperature, or other non-physical properties for e.g. visualization purposes, are assumed to be sampled at the center of the cell. Flow velocity components may also be sampled at the center of the cell. However, this approach leads to poor estimation of unbiased derivatives due to a non-trivial null space (Bridson, 2015), which could introduce problems during the projection step of the solver.

This problem is well-known in the CFD literature. Harlow and Welch (1965) addressed this problem by introducing a novel grid structure as part of their marker-and-cell (MAC) method. As Fig. 2.2 illustrates, rather than sampling velocities at the center of each cell, normal velocity components are sampled at the center of each cell face. This staggered method circumvents the null space limitation, facilitating the calculation of robust unbiased velocity derivatives at cell centers. The use of MAC grids became common practice in computer graphics after their introduction by Foster and Metaxas (1996).

Spatial and computational limitations constrain the scale and resolution of fluids discretized as grids. Applications such as liquid simulation may incur in a wasteful use of resources when simulating or storing in memory the entire domain if the regions of interest span only a small percentage of the domain. Moreover, full fidelity over the entire domain may not be necessary. In such cases, we could perform a coarser treatment of those low importance regions in an adaptive manner. This prompted the study of sparse or adaptive representations in the graphics community.

Sparse block grids are a straightforward substitute for dense grids. Bridson (2015) proposed a simple scheme based on hashing-backed associative structures that could be applied to fluid simulation. He noted that special care must be put to

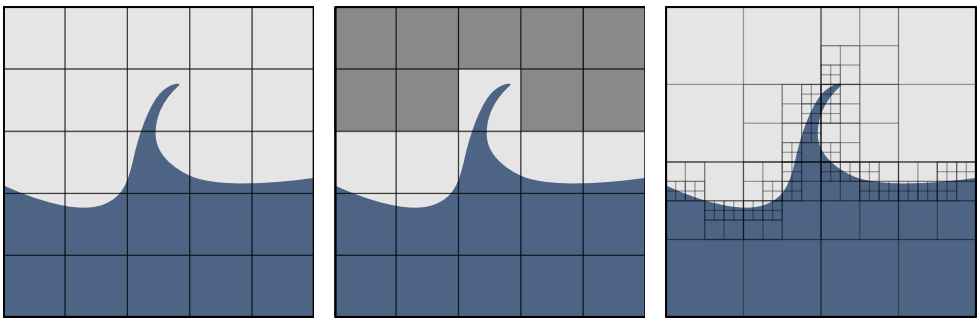


Fig. 2.3.: Comparison between dense, sparse and ntree-based adaptive representations. Dense grids **(left)** keep unoccupied cell information in memory at all times, which is resource-wasteful, especially in high-resolution simulations. In contrast, sparse **(center)** and adaptive ntree-based **(right)** approaches dynamically allocate resources to regions of high importance (e.g. inside the liquid volume or near the free-surface).

prevent redundant degrees of freedom in the interface between sparse blocks. This approach has recently found renewed interest in high-performance computing thanks to advances in efficient parallel sparse data structures. Bailey et al. (2015) presented a framework built upon OpenVDB (Museth, 2013; Museth et al., 2013) for efficient distributed computation of large-scale liquid simulations. Wu et al. (2018) leverage on GVDB data structures (Hoetzlein, 2016) for parallel sparse grid hierarchy construction and fast incremental updates on the GPU in the context of hybrid Eulerian-Lagrangian schemes.

Alternatively, refinement strategies based in octree structures (Meagher, 1982) have been shown to be successful substitutes to regular dense grids. Nonetheless, these structures are prone to produce T-junction structures, which must be handled carefully to prevent visual and numerical artifacts. Losasso et al. (2004) proposed the first simulation method for liquids based in octrees. In their work, pressure gradients near the T-junctions are approximated through the use of a Laplacian matrix, which limits the accuracy of pressures to first-order and results in unphysical eddy currents in T-junctions even in hydrostatic situations. This work was later extended (Losasso et al., 2006) to improve the precision to second order, at the cost of only allowing adaptivity in fully enclosed regions. Aanjaneya et al. (2017) proposed a method that eliminates numerical and visual artifacts of prior octree schemes. This is achieved by adapting the operators acting on the simulation variables to reflect the structure and connectivity of a power diagram, eliminating problematic T-junction configurations.

Other adaptive schemes may include the use of non-uniform grids. Chentanez and Müller (2011) proposed a method for simulating Eulerian fluids in real-time using a hybrid grid representation composed of regular cubic cells on top of a layer of tall cells. B. Zhu et al. (2013) introduced a grid structure that dynamically extends cells surrounding a fine uniform grid to represent far-field regions. These methods retain the efficiency of regular grids while including some of the advantages of adaptivity.

Advection Schemes

The momentum equation describes the evolution of the flow in terms of the material derivative. This derivative, though, is associated with a Lagrangian representation of the fluid. As the Eulerian description focuses on the evolution at fixed points in space, this expression must be generalized to include the concept of advection.

Advection is the term used to describe the transport of a substance or quantity as a result of the bulk motion of a fluid. Given some physical quantity q of a material element that is subjected to a space-and-time-dependent macroscopic velocity field $\mathbf{u}$, the material derivative is defined as:

$$\frac{Dq}{Dt} \equiv \frac{\partial q}{\partial t} + \mathbf{u} \cdot \nabla q, \tag{2.3}$$

in which the term $\mathbf{u} \cdot \nabla q$ models the rate of change of q due to transport. Plugging back this definition of the material derivative to (2.1) yields the Eulerian momentum equation:

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \mu \nabla^2 \mathbf{u} + \rho \mathbf{g}. \tag{2.4}$$

Solving the advection term $\left(\frac{\partial q}{\partial t} = -\mathbf{u} \cdot \nabla q \right)$ then becomes an additional stage in the solver pipeline as a result of operator splitting (Chorin, 1968). Unfortunately, standard numerical integration schemes (such as forward Euler) fail to find reliable solutions regardless of the choice of time step Δt due to instabilities caused by the discretization of the spatial derivatives (Bridson, 2015).

The physically-motivated semi-Lagrangian schemes offer a robust alternative for solving advection. First introduced by Robert (1981) for meteorological analysis, semi-Lagrangian schemes originate from the atmospheric sciences community for

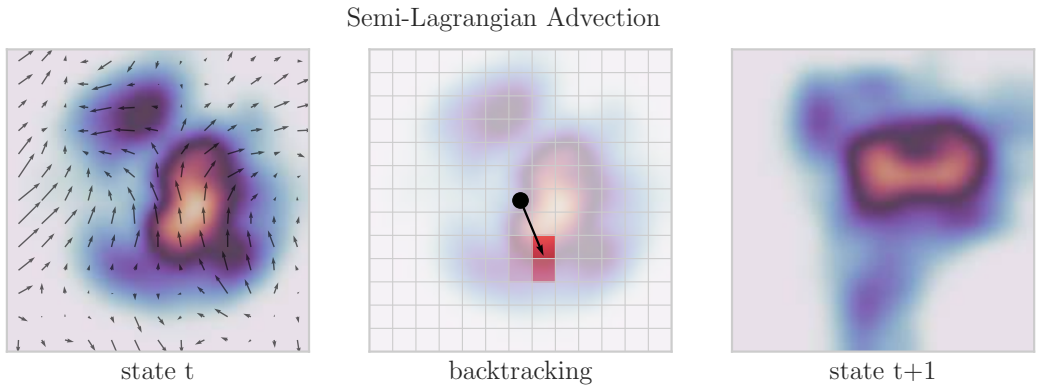


Fig. 2.4.: Semi-Lagrangian schemes solve advection for an arbitrary quantity subject to a macroscopic velocity field **(left)** by tracing the trajectory of each sampling point backwards in time **(center)**. The advected field is computed by interpolating the field quantities at the backtraced locations **(right)**.

modeling large scale flows, where large time steps are desired. The central concept behinds this scheme consists of placing fictitious particles at the grid's sampling points. Given that the velocity field $\mathbf{u}(t)$ at a given time instant t is known, it is possible to trace the particle's trajectories (e.g. through forward Euler or higher order integration schemes such as Runge-Kutta) and thereby calculate their location at a previous time instant. Then, as Fig. 2.4 illustrates, a solution for advection is obtained by interpolating the field quantities at these origin positions.

This simple scheme enjoys several benefits. Perhaps the most significant one is that it is *unconditionally stable*, allowing for the use of time steps far larger than the CFL criterion. Nonetheless, this stability is attained at the expense of significant numerical dissipation, which manifests as artificial diffusion. Fedkiw et al. (2001) and D. Kim et al. (2008) suggested higher-order interpolation schemes to mitigate diffusion in semi-Lagrangian advection schemes. B. Kim et al. (2005) introduced the BFECC method which compensates spatial and temporal errors through multiple backward and forward advection steps. Selle et al. (2008) generalized this technique by proposing a semi-Lagrangian variant of the MacCormack method and demonstrating second-order spatial and temporal precision. Molemaker et al. (2008) suggested using the QUICK advection scheme for low-dissipation advection, but this approach is constrained by the CFL stability condition. Alternatively, kinetic dissipation can be mitigated by artificially restoring energy back into the fluid. Fedkiw et al. (2001) employed vorticity confinement forces to counteract dissipation at vortices and eddies.

Enforcing Incompressibility

After solving the advection step, the resulting velocity field might not be divergence-free. This implies that the resulting flow is not incompressible, thus violating the continuity equation. To enforce incompressibility, advection-projection schemes map the velocity field to its closest divergence-free field. For this end, they rely on the Helmholtz-Hodge decomposition theorem (Bhatia et al., 2012), which states that any sufficiently smooth, rapidly decaying vector field ξ can be decomposed into the form:

$$\xi = \nabla\Phi + \mathbf{r}, \tag{2.5}$$

where Φ is a scalar potential function, whose gradient encodes the irrotational (curl-free) component of ξ; and $\mathbf{r}$ is a vector field that encodes the solenoidal (divergence-free) component. Fig. 2.5 illustrates the Helmholtz-Hodge decomposition of an arbitrary vector field. In the context of incompressible fluid dynamics, such decomposition is achieved by finding the pressure function p that satisfies the Poisson equation:

$$\nabla^2 \cdot p = \frac{\rho}{\Delta t}\nabla \cdot \mathbf{u}. \tag{2.6}$$

Under no constraints, an infinite set of functions p satisfy (2.6). To ensure the uniqueness of the solution, boundary conditions must be introduced. As illustrated by Fig. 2.6, two kind of boundary conditions are typically employed in Computer Graphics applications: Dirichlet and Neumann boundary conditions. On one hand, Dirichlet boundary conditions impose specific pressure values on the boundary (e.g.

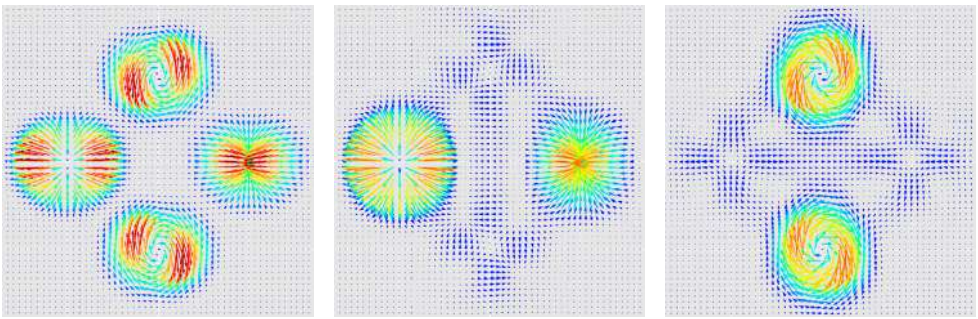


Fig. 2.5.: The Helmholtz-Hodge decomposition states that any sufficiently smooth, rapidly decaying vector field **(left)** can be decomposed into curl-free **(center)** and divergence-free **(right)** components. **Source:** Ribeiro et al., 2016.

$p(\Gamma_D) = 0$) often to model empty or open boundaries. On the other hand, Neumann boundary conditions impose particular pressure derivatives on the boundary (e.g. $\frac{\partial p}{\partial \mathbf{n}}\Big|_{\Gamma_N} = 0$ where $\mathbf{n}$ is the boundary normal) often used to model solid obstacles.

Discretizing (2.6) and carefully imposing the appropriate boundary conditions yields a linear system $\mathbf{Ap} = \mathbf{b}$ whose solution contains the necessary values to enforce the incompressible velocity field. Additionally, the resulting system is sparse, symmetric, and positive-definite (SPD), allowing for the resolution of the system to be accomplished using a large number of high-performance numerical algorithms. Iterative methods based on Krylov subspaces such as the Preconditioned Conjugate Gradient method (PCG) are popular options for solving the pressure Poisson problem (Foster & Fedkiw, 2001). Nonetheless, solving this linear system usually becomes the bottleneck of a typical fluid solver pipeline. The number of iterations required to achieve convergence dramatically increases with the scale of the problem, and the cost of storing and updating the preconditioner is prohibitive for large domains. Therefore, applications seeking interactive times must resort to alternate schemes susceptible to parallelization.

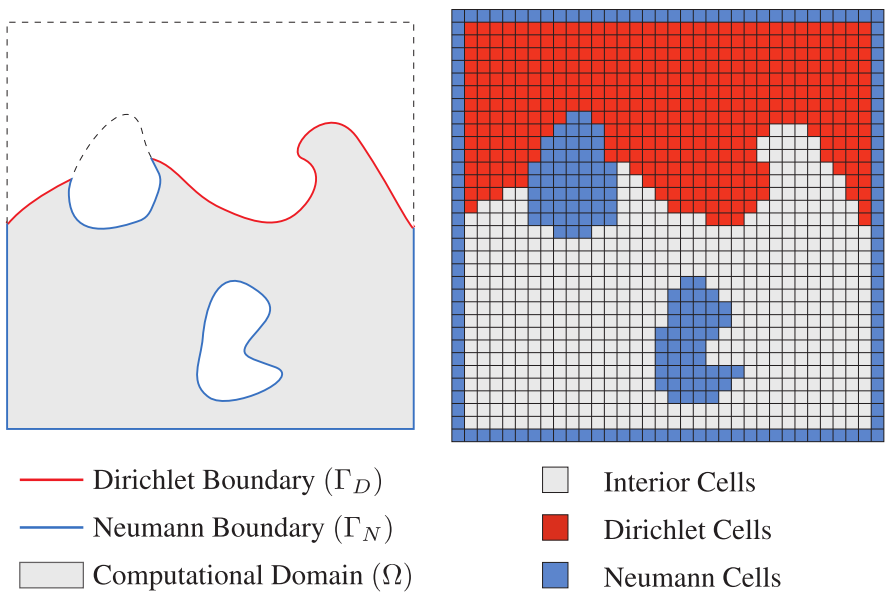


Fig. 2.6.: To ensure the uniqueness of the solution to the pressure Poisson equation, boundary conditions must be introduced. **(left)** Dirichlet and Neumann boundary conditions are introduced to model empty and solid boundaries respectively. **(right)** The corresponding boundary cells are flagged accordingly. **Source:** McAdams et al., 2010.

Certain discretization choices such as regular grids permit a direct mapping to a variety of acceleration schemes. Molemaker et al. (2008) took advantage of the regularity of Eulerian grids to develop a multigrid solver for inviscid incompressible flow. McAdams et al. (2010) employed multigrid cycles (McCormick, 1987) as preconditioner for the Conjugate Gradient method, dramatically improving convergence and robustness on irregular domains.

Over the last decades, advances in graphics hardware have also motivated researchers to explore novel strategies that make efficient use of their computational prowess. Krylov solvers, which are very efficient on CPUs, become comparatively less efficient on GPUs. The reason is that GPUs are memory bound, and the system assembly and sparse matrix-vector multiplications of Krylov solvers become inefficient. In contrast, relaxation methods (Jacobi, Chebyshev) enable almost trivial parallelization on GPUs, at the expense of more but faster iterations. A prime example of this is the work of Crane et al. (2007), who leveraged on the graphics hardware to implement a high performance parallel solver.

Due to its ability to produce highly-detailed flows at interactive rates, we follow the method of Crane et al. for the simulation of gaseous media in Chapters 3 and 4.

2.1.2 Lagrangian Methods

Early Lagrangian-based methods did not find the same degree of success for the simulation of fluid dynamics as Eulerian methods did. Despite their widespread adoption in the field of deformable body modeling, standard Lagrangian descriptions present significant challenges when applied to fluid media. While mesh-based representations (which are the standard approach for deformable bodies) enable robust evaluation of the quantities within the fluid volume, they are hindered by the need for sophisticated remeshing techniques to handle the complex dynamic topologies that arise in fluids. Mesh-free approaches (i.e. particle systems), on the other hand, allow a simpler representation of these complex topologies, but introduce the problem of computing the spatial derivatives needed to model the flow's evolution.

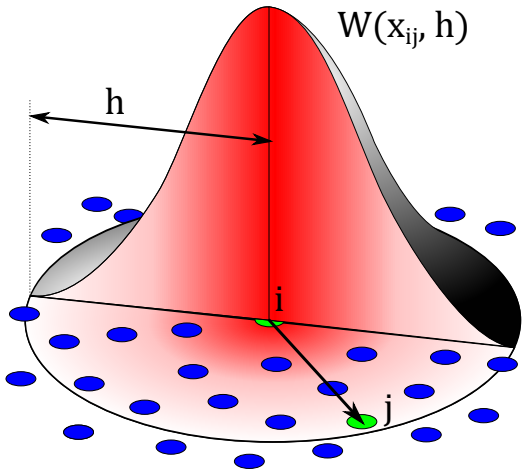


Fig. 2.7.: Smoothed Particle Hydrodynamics (SPH) provides a framework to approximate a spatially-continuous field at an arbitrary location i through weighted summation. Particle contributions are modulated according to the smoothing kernel function W. **Source:** Wikimedia.

Researchers found renewed interest in Lagrangian descriptions for fluid simulation with the introduction of the Smoothed Particle Hydrodynamics (SPH) method to the graphics community during the late 90s (Desbrun & Gascuel, 1996). Developed by Monaghan (1992) for its application in the field of computational astrophysics, SPH is a Lagrangian method specifically designed for the analysis of compressible flow problems typically arising in astrophysical phenomena. At its core, SPH is a mesh-free interpolation scheme. The method samples the fluid domain Ω into a set of discrete points (also known as *smoothed particles*) which carry intrinsic flow properties and quantities such as mass, position, velocity, temperature, etc. Then, as Fig. 2.7 shows, these particles are used to approximately reconstruct spatially-continuous field quantities through a weighted summation (i.e. smoothing) of the discrete values. Approximate derivatives required to model flow evolution can then be determined through analytical differentiation.

This Lagrangian scheme presents several advantages over traditional Eulerian techniques. Conservation of mass is guaranteed without extra computation as particles themselves represent mass. Pressures can be computed from contributions of neighboring particles rather than by solving linear systems of equations. Moreover, free surface tracking for multi-phase fluids is automatic thanks to the particle representation. Nonetheless, compared to grid-based techniques, a large amount of particles is required to produce simulations of equivalent resolution.

Graphics researchers found that this approach could be extended beyond its original scope to address a wide range of computer graphics problems. The seminal works of Desbrun and Gascuel (1996) and Müller et al. (2003) demonstrated its applicability to the animation of inelastic bodies and fluid dynamics respectively. Since then, the majority of research on Lagrangian fluid simulation has focused on finding strategies to overcome its limitations while extending the method's applicability to a wider range of materials.

Smoothed Particle Hydrodynamics

Within the SPH framework, any arbitrary spatially-continuous field quantity A can be approximated through the interpolant:

$$A(\mathbf{x}) = \sum_j A_j \frac{m_j}{\rho_j} W(\mathbf{x} - \mathbf{x}_j, h), \tag{2.7}$$

where $\mathbf{x}$ is the point being evaluated, A_j is the discrete field attribute A of particle j and $\mathbf{x}_j$, m_j and ρ_j are the particle's position, mass and density respectively. The *smoothing kernel* function W (usually formulated as isotropic Gaussian-like functions with finite support h) modulates particle contributions to the interpolation. Fig. 2.8 depicts examples of smoothing kernel functions typically employed in graphics. Spatial derivatives can be easily approximated thanks to the linearity of the operator (∇):

$$\nabla A(\mathbf{x}) = \sum_j A_j \frac{m_j}{\rho_j} \nabla W(\mathbf{x} - \mathbf{x}_j, h), \tag{2.8}$$

$$\nabla^2 A(\mathbf{x}) = \sum_j A_j \frac{m_j}{\rho_j} \nabla^2 W(\mathbf{x} - \mathbf{x}_j, h). \tag{2.9}$$

With these tools, fluid dynamics can be achieved by discretizing and solving the individual terms of the Navier-Stokes momentum equation (2.1). As particle representations preserve mass, the continuity equation (2.2) can be omitted. Müller et al. (2003) follows this approach to solve free-surface fluid dynamics. In their

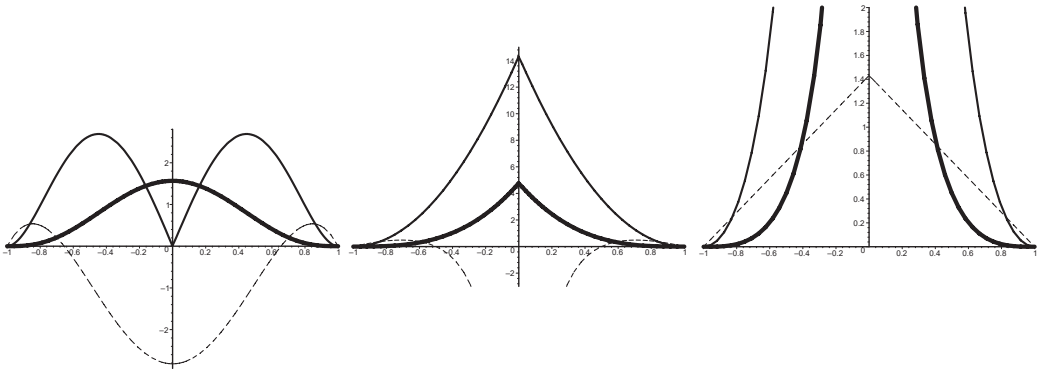


Fig. 2.8.: The three smoothing kernels W_{poly6}, W_{spiky} and $W_{viscosity}$ (from left to right) defined in the work of Müller et al. (2003) that are commonly used for the simulation of fluid dynamics. The thick lines show the kernels, the thin lines their gradients in the direction towards the center and the dashed lines the Laplacian. Note that the diagrams are differently scaled. The curves show 3-d kernels along one axis through the center for smoothing length $h = 1$. **Source:** Kelager, 2006.

work, internal pressures are estimated through the the ideal gas equation of state (EOS):

$$p_i = k(\rho - \rho_0), \tag{2.10}$$

where ρ_0 is the fluid's rest density and k is the gas constant which depends on temperature. While the method produces convincing results, this treatment of pressure terms often leads to drift in the incompressibility condition under moderately large time steps ($\Delta t \leq 10^{-4}$), rendering it infeasible for the modeling of certain types of liquids (e.g. water) in graphics applications unless prohibitively small time steps are used.

Enforcing Incompressibility

Strategies for strictly enforcing incompressibility have become a major focus of computer graphics research. Pressure propagation is closely related to the equation of state employed for the computation of pressure forces. Becker and Teschner (2007) proposed a weakly compressible approach (WCSPH) based on the Tait EOS to achieve higher incompressibility, usually tuned to never exceed density deviations of 1%. However, this method suffers from numerical instabilities caused

by the resulting stiff pressure forces, leading to severe time step restrictions to satisfy the CFL condition.

Similarly to Eulerian methods, incompressibility can be enforced through projection schemes. Incompressible SPH (ISPH) methods are devoted to find strategies in this direction. Colin et al. (2006) enforced a divergence-free velocity field by solving the pressure Poisson equation (PPE) within a SPH framework. While this approach admits larger time steps than WSCPH, the computational cost per time step is significantly higher. Later, Ihmsen et al. (2013) proposed a novel discretization of the PPE that considers the relation between solved pressures and forces to improve the convergence rate of the solver. Alternatively, Solenthaler and Pajarola (2009) proposed a predictive-corrective scheme (PCISPH) that iteratively adjusted particle pressures to achieve a target rest density. This approach avoids the computational costs associated with solving the PPE, resulting in a significant reduction in computational cost while maintaining the ability to use large time steps. Bender et al. (2014) further extended these works by proposing a dual solver that enforced low volume compression and divergence-free velocity fields, resulting in increased stability and convergence (and consequently, performance).

Projection schemes may be deemed as a particular case of the more general idea of constrained dynamics solvers. These formulations have proven successful for the simulation, among others, of articulated bodies (Baraff, 1996), contact (Baraff, 1989; Kaufman et al., 2008), deformation limits (H. Wang et al., 2010), inextensibility (Goldenthal et al., 2007), volume preservation for solids (Irving et al., 2007), fluid incompressibility (Foster & Metaxas, 1996; Solenthaler & Pajarola, 2009), or even generic dynamic deformations (Bender et al., 2014; Müller et al., 2006; Stam, 2009). By expressing stiff properties as constraints, a large variety of high-performance solvers become available.

A popular constrained dynamics solver in computer graphics is the Position-based Dynamics (PBD). Proposed by Müller et al. (2006), this method addresses the resolution of stiff elastic potentials by modeling them as constraints formulated on particle positions. Similarly to advection-projection schemes (Chorin, 1968), PBD constraints are enforced after evaluating a constraint-free state for the next time step. However, instead of employing a global solver, constraint projection in PBD is typically performed using Fast-Projection Jacobi or Gauss-Seidel iterations. Constraints are linearized after each iteration, and Fast Projection implies that,

within each iteration, the projection is computed by minimizing the distance to the result from the previous iteration, not to the initial value (Goldenthal et al., 2007; Hairer et al., 2002).

In contrast to schemes based on system linearization, this iterative algorithm is more robust in handling complex (non-linear) configurations. Moreover, the algorithm is straightforward to implement, highly parallelizable, and robust to large time steps and degenerate configurations, making it specially suitable for its use in interactive applications. Although the original formulation of the method is not derived from continuum mechanics foundations, its XPBD extension (Macklin et al., 2016) links the method to implicit integration schemes, adding a relaxation term to the constraint projection in order to model constraint compliance.

Position-Based Fluids

Macklin and Müller (2013) extended the Position-Based Dynamics method to address the problem of simulating incompressible fluids, known as Position-Based Fluids (PBF). In their method, pressure computation is omitted in favor of imposing a constant density constraint formulated for each fluid particle in the system:

$$C_i(\mathbf{x}_1, ..., \mathbf{x}_n) = \frac{\rho_i}{\rho_0} - 1, \tag{2.11}$$

where C_i corresponds to the density constraint of the i-th particle, ρ_i to its evaluated density (as computed by the standard SPH interpolant) and ρ_0 corresponds to the fluid's rest density. Enforcement of this constraint leads to configurations that satisfy fluid incompressibility. However, the resulting particle distributions are prone to clustering (cf. Fig. 2.9). This problem arises as a consequence of deficient neighborhoods appearing near the free-surface. Macklin et al. (2014) address this problem by adding an artificial pressure term which keeps particle densities slightly lower than the rest density, thus producing a surface-tension effect and an improved particle distribution.

Position-Based Fluids achieves comparatively similar convergence rates to modern SPH solvers in moderately-sized scenes (∼100K particles) at a fraction of their computational cost, leading to real-time simulations. However, these desirable qualities

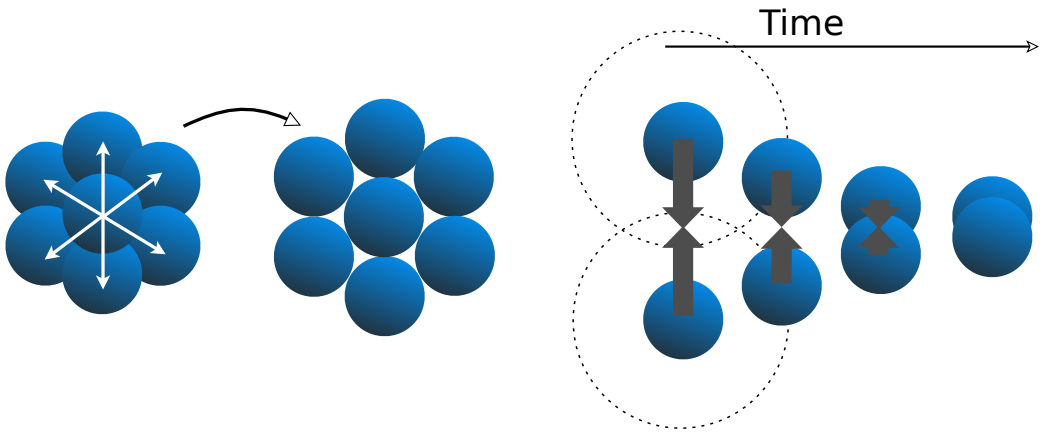


Fig. 2.9.: From left to right: **(a)** The incompressibility constraint produces a displacement on the particles position in order to enforce a constant density across the fluid. **(b)** Particle clumping, or tensile instability, is produced when the constraint is unable to satisfy the nominal density due to deficient neighborhoods. **Source:** Macklin and Müller, 2013.

come at a cost. Fluids simulated with PBF suffer from numerical dissipation. Such dissipation can be mitigated through the use of second order schemes (Macklin et al., 2014) and vorticity confinement techniques (Fedkiw et al., 2001; Macklin & Müller, 2013). In addition, certain configurations may complicate the convergence of the Jacobi or Gauss-Seidel solvers, specially due to the nonlinearities introduced by the SPH kernels used in PBF. In such cases, great care must be put to ensure the stability of the simulation. Finally, the range of materials that PBF is capable of representing is limited to inviscid fluids or fluids with a low coefficient of viscosity. As detailed in Section 2.1.5, other interesting media such as highly viscous or non-Newtonian materials are difficult to reproduce with this method.

Despite this, the method's widespread adoption in industry-standard animation software packages and real-time applications demonstrates that it is a viable approach for simulating fluid dynamics in interactive graphics applications. In Chapter 4, we introduce a PBF-based approach for efficiently reproducing highly viscous and viscoelastic fluids. To accomplish this, we expand the Position-Based Dynamics algorithm to enforce constraints at both position and velocity level. In Chapter 6 we build upon a simplified version of this model for the interactive simulation and rendering of clay-like materials.

2.1.3 Hybrid Methods

While Lagrangian or Eulerian methods constitute the traditional approaches in the CFD and graphics literature for solving fluid mechanics problems, alternative schemes for modeling flow dynamics exist. Over the past decades, hybrid methods combining grid and particle representations have become very popular in computer graphics applications. Among these, the Particle-In-Cell (PIC) technique family (Harlow & Welch, 1965) is perhaps the most well-known.

Introduced by Y. Zhu and Bridson (2005) to computer graphics through the Fluid Implicit Particles method (FLIP), the central idea behind these methods is to transfer the fluid quantities back and forth between Lagrangian and Eulerian discretizations to solve different aspects of the flow (Fig. 2.10). While particles are used to evaluate the advection of fluid physical quantities, an auxiliary Eulerian grid is employed to project the velocities to guarantee incompressibility.

The original PIC method achieves high-resolution flows even on coarse meshes at the cost of increased dissipation and memory requirements. FLIP improves on PIC by incorporating a simple blending scheme between particle and grid velocities, which minimizes dissipation. However, manual adjustment of the blend weights on a case-by-case basis is necessary to ensure the simulation's stability.

Dissipation in PIC is related to information loss during particle-grid transfers. To address this problem, Jiang et al. (2015) developed the Affine Particle-In-Cell (APIC) scheme. In their work, an affine representation of local momentum is explicitly stored on a per-particle basis to preserve affine velocity fields when transferring from grid to particle. As a result, this scheme achieves extremely stable, low-dissipation flows with little overhead with respect to PIC/FLIP. This scheme was further generalized by Fu et al. (2017) with the introduction of the Polynomial Particle-in-Cell (PolyPIC) scheme which stores a polynomial representation instead.

Recently, the Material Point Method (MPM) has gained popularity in the graphics community for its ability to simulate a variety of different effects and materials. Developed by Sulsky et al. (1995), MPM was conceived as a generalization of the PIC/FLIP methods for the simulation of multiphase (solid-fluid-gas) interaction

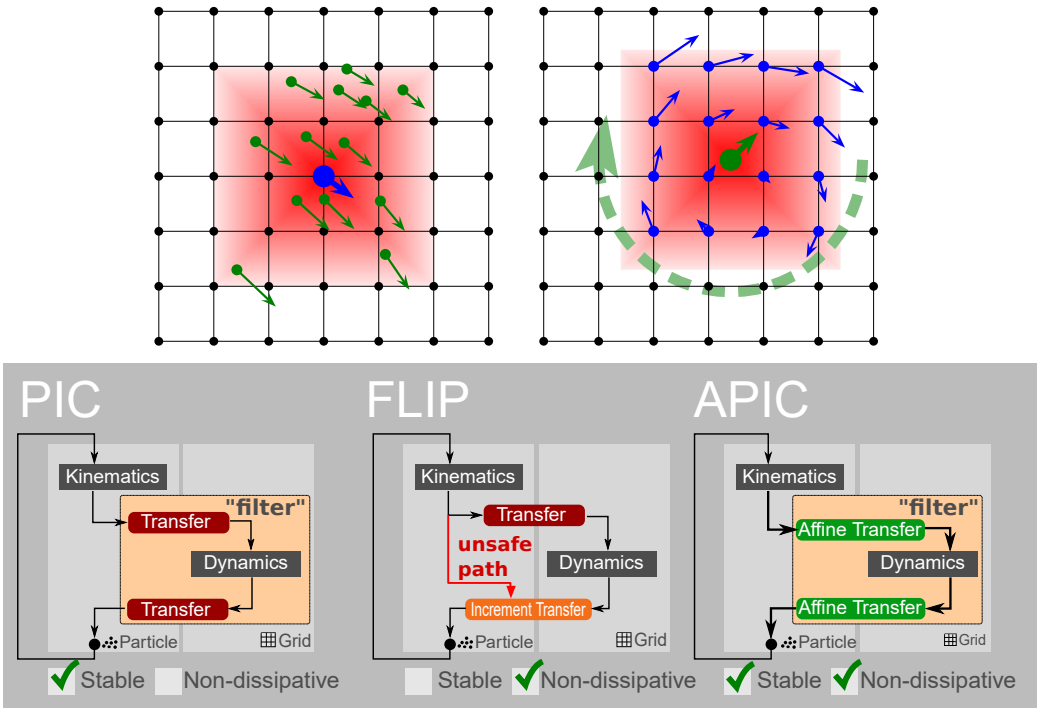


Fig. 2.10.: **(top)** Traditional PIC methods transfer fluid information from particles to grid and viceversa through averaging. **(bottom)** The specific transfer scheme employed has a direct impact on the stability and dissipative properties of the resulting simulation. **Source:** Jiang et al., 2015.

problems. Although the method produces excellent results, it is computationally intensive.

2.1.4 Medium-Specific Details

Additional considerations may be necessary depending on the fluid phenomena being simulated. Fire and smoke are examples of this. As fuel chemically reacts to form hot gaseous products (smoke and soot), these mediums expand and rise due to buoyancy forces. As such, the incorporation of reaction-diffusion models for combustion and the consideration of thermal effects such as buoyancy are required to produce compelling results (Fedkiw et al., 2001; Nguyen et al., 2002).

In the case of liquids, the interfacing media (e.g. air) can be disregarded, and the problem translated into how to track the fluid's free surface. While tracking of the

free-surface is inherent to Lagrangian methods, Eulerian methods require additional mechanisms to determine the occupied regions of the domain. Hybrid approaches such as marker particles (Harlow & Welch, 1965) are among the simplest tracking mechanisms to determine occupied cells, but resampling mechanisms are needed to ensure consistent representation of small-scale details. Alternatively, level set approaches (Osher & Sethian, 1988) provide an implicit representation of the liquid that enables robust free surface tracking. However, special care must be put to prevent volume loss as a result of numerical dissipation. Particle level set methods (Y. Zhu & Bridson, 2005) circumvent this problem by inserting particles around interfaces. Accurate surface tracking is provided by updating the particles independently of the level set and using their information to correct the function. However, this accuracy is achieved at the expense of high computational cost.

2.1.5 Viscoelasticity

Beyond modeling inviscid fluids, the simulation of complex fluid materials such as those presenting high viscosity or nonlinear stress-strain relationships has captured the interest of many researchers in the field of computer graphics for many years. This is unsurprising, given that all of these visually appealing behaviors and effects are unique to these types of fluids.

In the case of high viscosity, naive integration (e.g. employing explicit schemes) of the diffusion term present in the momentum equation (2.1) significantly restricts the variety of materials that can be expressed under reasonable computational time due to the resulting stiff forces (CFL condition). Consequently, researchers have devoted considerable effort to develop alternative integration schemes for representing these materials.

The first attempts in computer graphics relied on grid-based discretizations (Terzopoulos & Fleischer, 1988). Carlson et al. (2002) simulated highly viscous fluid materials and melting effects using an implicit viscosity formulation over a Marker-and-Cell (MAC) approach. Goktekin et al. (2004) extended the Navier-Stokes equations incorporating elastic and plastic terms to simulate viscoelastic fluids over a level-set discretization. Batty and Bridson (2008) designed an accurate method for the simulation of characteristic effects in free-surface viscous fluids,

such as buckling and coiling. They emphasized the formulation of correct boundary conditions, outlined a variational formulation of viscosity, and designed an efficient solver. Wojtan and Turk (2008) enabled the preservation of thin viscoelastic features in finite element simulations. Larionov et al. (2017) proposed an implicit Stokes solver to simulate highly viscous Newtonian fluids also using a grid-based discretization. The focus of these methods has been to enable complex effects, without particular attention to computational performance.

Some works have designed solutions optimized for the simulation of specific viscosity effects. Bergou et al. (2010) and Batty et al. (2012) focused on the simulation of viscous threads and thin layers, respectively. Remeshing strategies helped them preserve thin surfaces and reduce the simulation cost. B. Zhu et al. (2015) developed a method for the simulation of viscous effects on features of different dimensions, all handled in a uniform manner. They achieve high-quality simulation of very thin features for non-Newtonian fluids.

Other works leveraged on the natural ability of methods such as the Material Point Method (MPM) to model viscoelasticity. Stomakhin et al. (2013) first applied MPM to snow simulation, and later extended it to phase changes and high viscosity (Stomakhin et al., 2014). Ram et al. (2015) used the MPM formalism to simulate viscoelastic materials, while Yue et al. (2015) applied it for the simulation of foam. More recently, this method has been applied to the simulation of sand (Daviet & Bertails-Descoubes, 2016; Klár et al., 2016) and the interaction between sand and water (Tampubolon et al., 2017).

Particle-based discretizations such as Smoothed Particle Hydrodynamics (SPH) have also been extended to the simulation of these highly-deformable materials with good computational performance. Solenthaler et al. (2007) proposed a unified model to simulate melting and solidification effects. They incorporated an elastic force term based on a strain measure, inspired by the previous work of Müller et al. (2004). Paiva et al. (2006) used an XSPH velocity correction to simulate non-Newtonian fluids, which was been extended by Andrade et al. (2015) to reproduce buckling in Newtonian viscous fluids. Chang et al. (2009), on the other hand, used an SPH discretization of the elastic strain tensor to simulate viscoelastic behavior. He et al. (2012) used the SPH approximation to solve the Poisson equation locally and simulate moderately viscous fluids. Granular media can be considered a special type of non-Newtonian fluid. Lenaerts and Dutré (2009) extended the

work of Solenthaler et al. (2007) to simulate granular behavior, and Alduán and Otaduy (2011) proposed strain-rate-based models for granular friction and cohesion effects.

Recently, several authors have designed implicit SPH solvers for the simulation of highly viscous fluids. Bender and Koschier (2017) proposed a divergence-free SPH solver that also supports viscous materials. T. Takahashi et al. (2015) enforced incompressibility in simulations with high viscosity by solving for pressure implicitly. Peer et al. (2015) simulated high viscosity by canceling shear rate in a least squares manner. For each particle, they set a goal velocity gradient that cancels the local shear rate, and they solve for the velocity field that best matches the goal velocity gradients. Peer and Teschner (2017) later extended their method to improve vorticity handling. The resulting method achieves high performance in moderate to large scale scenarios.

The addition of elastic effects to viscosity simulations requires, in principle, knowledge of undeformed configurations to simplify the computation of deformation metrics. However, maintaining this knowledge becomes complicated under plastic flow. Clavet et al. (2005) used a spring-based approach to apply elastic forces in a particle-based fluid simulation, and they also proposed a strategy to create and remove elastic links between particles as the particle neighborhoods evolve. Gerszewski et al. (2009) introduced instead an approach to measure deformation that does not require an explicit undeformed configuration in a Lagrangian plastic flow model.

Viscous fluid simulation has also been addressed in the context of Position-based Fluids (PBF). T. Takahashi et al. (2014) proposed a constrained-based treatment of viscosity. They define particle links similar to those of Clavet et al. (2005) to describe the local material structure, addressing also elasticity and volume conservation (T. Takahashi et al., 2016). However, their model fails to prevent drift without suffering undesired elastic oscillations.

In Chapter 3 we propose a constraint-based solution for fluid viscoelasticity. Our solution is inspired by a constitutive model of polymeric fluids (i.e. fluids where elastic polymers are dissolved), which supports a large range of viscoelasticity behaviors under one common formulation (Deshpande et al., 2010). As T. Takahashi et al. (2014), our viscoelasticity model also avoids the explicit definition

of undeformed configurations. We achieve notion of this configuration thanks to a conformation tensor whose time evolution describes the state of the fluid. Moreover, our formulation reproduces a larger palette of viscoelastic materials and achieves even higher efficiency than the work of Peer et al. (2015) on moderately large scenes thanks to a PBD-style constrained dynamics solver. A simplified version of this formulation is employed in Chapter 6 for the simulation of clay materials.

Since the publication of our work, researchers have further studied the modeling of viscous and viscoelastic fluids. Weiler et al. (2018) proposed an implicit viscosity solver based on a combination of SPH and finite differences to discretize the Laplacian of the velocity field. The resulting method guarantees physically meaningful behavior under spatial and temporal refinement, preventing ghost forces caused by deficiencies at the free surface. However, this discretization of the Laplacian requires the enforcement of a divergence-free velocity field. In the context of MPM, Fang et al. (2019) presented a predictor-corrector finite strain integration scheme for general viscoelastic solids under arbitrarily large deformation and non-equilibrated flow.

2.2 Ultrasound Haptics

The advent of non-contact haptic displays has introduced new forms of interaction, allowing users to experience tactile stimuli in mid-air without the need for holding or wearing a device. Ultrasound haptic devices based on phased arrays are notable examples of this type of displays.

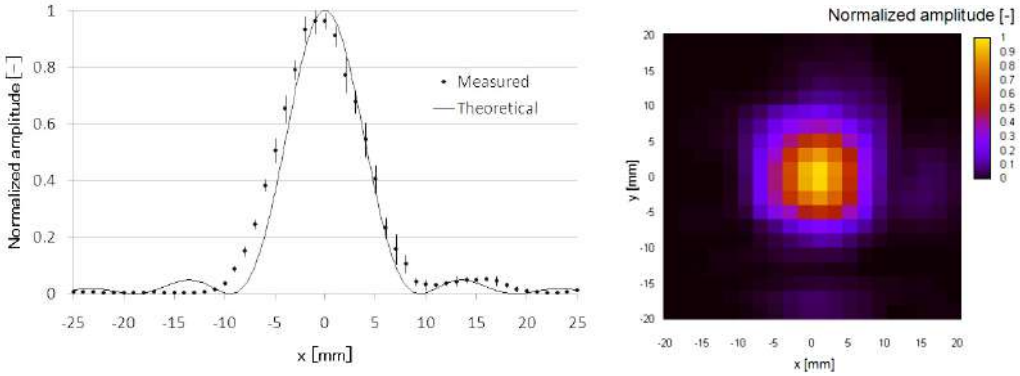
As illustrated by Fig. 2.11, ultrasonic phased arrays (UPA) achieve mid-air stimulation through a phenomenon known as *acoustic radiation pressure*. Multiple ultrasound transducers, producing an ultrasound wave of the same frequency, are modulated in phase to achieve maximal combined pressure intensity at a certain location in space, known as focal point. When interacting with skin tissue, the pressure exerted by these acoustic waves elicits complex mechanical waves on the skin, producing an at least equally complex activation and aggregation of mechanoreceptor signals to form tactile percepts. The size of the stimulus (focus) depends on the ultrasound wavelength (e.g. 8.5mm for 40kHz ultrasound (Hoshi

Fig. 2.11.: **(a)** Illustration of a ultrasonic phased array board modelled after the Ultraleap STRATOS Explore (USX). **(b)** Transducer activation is modulated to achieve maximal pressure intensity at desired locations in space.

et al., 2010), cf. Fig. 2.12). Although the force output of these phased arrays is comparatively lower to other mid-air devices (16mN *vs.* 1N for devices based on air flows), their low latency, high spatiotemporal resolution and large interaction space make ultrasonic phased arrays a reliable and appealing technology for mid-air tactile applications. As a result, these devices have grown in popularity over the past decade.

The mid-air interaction capabilities of ultrasound haptics have enabled some unprecedented applications. One of them is the interaction with floating images produced with 3D displays (Monnai et al., 2014), giving to the user the illusion of seeing and touching a virtual object in a fully co-located and natural manner. By tiling ultrasound phased arrays in 3D, the approach has been extended to enable tactile interaction with holograms (Inoue et al., 2015). Beyond conveying tactile feedback, UPAs have also been applied to applications such as acoustic levitation (Iodice et al., 2018; Morales et al., 2019; Morales González et al., 2020), parametric audio generation (Shakeri et al., 2019) and even wireless power transfer (Morales González et al., 2021).

The generation of tactile percepts using ultrasound devices is still a largely unknown process. In the absence of a computational model that maps activation patterns of transducers to tactile percepts, researchers have explored different high-level metaphors to command ultrasound devices. To date, two control metaphors prevail: amplitude modulation (AM) and spatiotemporal modulation (STM).



(a) (b)

Fig. 2.12.: Spatial distribution of the acoustic radiation pressure around a focal point centered at 20mm of distance of the UPA. The size of the main lobe (i.e. the lobe containing the higher power) depends on the ultrasound wavelength λ. In this case, $\lambda = 8.58mm$. **(a)**. Scan along one axis of the focal plane. **(b)**. Scan along the two axes of the focal plane. **Source:** Hoshi et al., 2010.

2.2.1 Amplitude Modulation

According to the Hyugens-Fresnel principle and the principle of superposition of waves, the pressure wave at an arbitrary point in space $\mathbf{x}$ can be expressed as a complex-valued linear function of the amplitudes and phases of the waves emitted by the transducers in a free field:

$$\mathbf{p}(\mathbf{x}) = \sum_i \frac{D(\theta_i)}{r_i} e^{-(\beta + jk)r_i} \mathbf{p}_i \tag{2.12}$$

where $\mathbf{p}_i$ is the acoustic pressure produced by transducer i located at a distance r_i and angle θ_i emitting a signal with wavenumber k. Physical characteristics of the transducer and the medium are parameterized through the directionality function $D(\theta)$ and exponential decay coefficient β which depends on acoustic wave frequency and air humidity. Given target pressure waves at a set of focal points, it is possible to obtain the optimal emitted waves $\mathbf{p}_i$ by solving a phase retrieval problem. However, typically only the target amplitudes are given, which makes the problem nonlinear.

Iwamoto et al. (2008) first demonstrated the capability of UPAs to focus sufficient acoustic radiation pressure to induce a haptic sensation in a localized area on the hand. Subsequently, Hoshi et al. (2010) extended this achievement to vary the focal location over time, producing the sensation of a moving stimulus. These early works used a precomputed time-varying solution for ultrasound phases given a desired spatial amplitude distribution. Because these phases must be determined by means of a non-linear quadratic optimization procedure, the calculations were not amenable to real-time control. Inoue et al. (2015) compared various solutions from the field of scattering diffraction imaging, while Long et al. (2014) designed an efficient solution that takes advantage of the linearity of the complex-valued eigenproblem associated with the desired focal point intensities.

Although the induced skin deflection is slight (in the order of micrometers), it is enough to trigger the mechanoreceptors embedded in the skin to evoke a tactile sensation. However, mechanoreceptors undergo rapid adaptation to stimuli in this frequency range, resulting in perception only at the onset and offset of the acoustic waves produced by spatially stable focal points (Frier, 2020). Researchers overcame this limitation by modulating the intensity of the field at a frequency to which the skin's mechanoreceptors (principally the Pacinian corpuscles) are sensitive, approximately in the range of 200 to 250 Hz.

Considering the focal point location as stable relative to this frequency range, this method later became known as *amplitude modulation* (AM). While this method is effective for local stimulation, it suffers from limitations. One of the most evident ones is that modulation induces a perceivable vibration on skin. Another limitation is that multiple focal points are necessary to produce stimuli covering surfaces larger than the focal size, resulting in a drastic reduction in the device's output capabilities when limited by safety constraints.

2.2.2 Spatiotemporal Modulation

To overcome the limitations of AM, researchers explored alternative modulation techniques. Korres and Eid (2016) presented a display method in which focal point amplitudes are relatively stable but their location is modulated at much higher rates. This approach, later dubbed *spatiotemporal modulation* (STM) by Kappus and Long

(2018), relies on the phenomenon of persistence of perception, which is related to the temporal resolution of the sense of touch: by rapidly and repeatedly moving a focal point across a given path, a continuous tactile perception is produced. Kappus and Long demonstrated that such paths could produce recognizable tactile shapes.

This control strategy enjoys several advantages over AM. The absence of pressure modulation results in a perceptually smoother stimulation. Furthermore, STM achieves larger coverage with a lower number of focal points when compared to AM, resulting in a perceivable stronger stimulation. However, since STM introduces a dependency between the refresh rate and the velocity of the focal point, new research is needed to understand the tactile perception induced by this method and how these parameters can be tuned to achieve optimal sensitivity.

2.2.3 Perceptual Effects

The perception of tactile stimulation produced by focused ultrasounds is widely influenced both by the spatial-temporal characteristics of the acoustic stimulus and by the structure and mechanical properties of the skin. The appearance and propagation over time of the ultrasound-induced mechanical waves plays an important role in the overall perceived intensity of the stimuli. However, this process is the result of complex interactions that remain not fully understood due to the difficulty in observing and capturing the evolution of these waves in real life. In the absence of a system that allows precise measurement of mechanical interactions and the consequent activation of skin mechanoreceptors, researchers have tried to approach the problem from a perceptual point of view.

Several works have analyzed perceptual aspects of ultrasound haptic stimulation, as well as performance implications of algorithmic parameters. Carter et al. (2013) studied how to position secondary focal points of low pressure to eliminate spurious pressure maxima within the device workspace. They also analyzed the ability of subjects to discriminate focal points. Wilson et al. (2014) studied the ability of subjects to locate focal points and the perception of motion, and they found, for example, an average localization error of 8.5mm. Hasegawa and Shinoda (2018)

have analyzed the pressure fields actually produced by ultrasound phased arrays, as well as perceptual detection thresholds.

Recent studies have analyzed the effects of rendering parameters of spatiotemporal modulation. Frier et al. (2018) found that there is some optimal focal point speed (between 5 and 8 m/s in their experiments) to maximize skin sensitivity. Recently, they have analyzed the combined influence of spatial and temporal sampling in spatiotemporal modulation (Frier et al., 2019). R. Takahashi et al. (2018) proposed a method that falls in between AM and STM, wherein they use a spatial modulation of the constant-intensity focal point at 1000 Hz to produce a vibration signal in a localized range. They report lower detection thresholds than for a single AM point.

Howard et al. (2019) investigated the detection and discrimination of line patterns with respect to intensities, as well as the discrimination of bumps and holes based on varying intensity over a line. They also found that STM line patterns had a lower detection threshold than a single AM point. Finally, Reardon et al. (2019) captured the wave propagation induced in the palm by ultrasound haptics using an optical vibrometer, and found that waves above 4 m/s induced compression. In this regime, it was found that inhibiting these waves reduced motion perception, thus a connection between wave propagation in the skin and perception was inferred.

2.3 Coupling Fluids and Haptics

Haptic rendering of interaction with fluids has received large attention. The various existing methods address the challenge of running interactive fluid simulations, and propose different coupling strategies between the haptic device and the simulated media.

2.3.1 Haptic Rendering of Fluid Media

Most of the works addressing haptic rendering of fluid media revolve around the use of tool-like devices. Baxter and Lin (2004) developed a 2D fluid simulation with haptic interaction, where they applied to the device the forces aggregated on the boundary of the simulated tool. Dobashi et al. (2006) developed a method that combines a real-time computation of linear force terms with precomputed nonlinear force terms. Mora and Lee (2008) computed a real-time 3D fluid simulation, and mapped viscous forces to the haptic device.

The advent of GPUs as computational platforms has allowed richer interactive methods. Yang et al. (2009) leveraged this technology and implemented methods to accumulate grid forces on the simulated tool directly on the GPU. Cirio, Marchal, Otaduy, et al. (2013) performed a particle-based simulation of the fluid on the GPU, and used a virtual coupling method to transfer fluid forces on the simulated tool to the haptic device.

Rendering tactile interaction with virtual environments (including fluids) using ultrasonic haptics can be formulated as a problem of dynamically reproducing a pressure field on the user's skin. However, neither AM nor STM alone satisfy the needs of our rendering problem, as they cannot reproduce a spatially varying pressure field.

In Chapters 4 and 5 we address this problem by introducing two rendering algorithms that dynamically map arbitrary target pressure fields to AM and STM control metaphors respectively. Our approaches pose this mapping as optimization problems that take into account known perceptual and technical limitations of both methods, resulting in an optimal set of focal points and paths that best reconstruct the target pressure field. We apply the resulting methods to the interaction with gaseous media, which we find can be reasonably matched even under the power limitations of current ultrasound hardware.

Prior to our work, haptic interaction with fluid media has been explored exclusively using tool-based interaction, with no work addressing direct-touch interaction, let alone mid-air interaction using ultrasound haptics. One tangentially related work is the use of vibrotactile feedback to render the interaction with splashing fluids,

conveying the vibrations produced by air bubbles (Cirio, Marchal, Lécuyer, et al., 2013). Following the publication of our work, Jang and Park (2020) presented a method for interacting with SPH-based liquids through the optimization of AM focal points, employing a hill-climbing algorithm.

2.3.2 Simulation of Virtual Clay

Clay is a viscoplastic material whose behavior resembles both a solid and a fluid. It exhibits microscopic material bonds that preserve shape, but these bonds are fragile and the material flows even under small stress, although with very high viscosity. Due to this complexity, simulation of clay is a computationally challenging problem, and existing interactive methods barely approximate its true behavior. Nevertheless, multiple works have attempted to partially model the behavior of clay in VR applications, including haptic feedback.

Early models for VR-based modeling of clay used combinations of spline surfaces and voxelized volumes (Krause & Lüddemann, 1997). To counteract the high cost of voxelization, McDonnell et al. (2001) proposed adaptive volumetric representations. They combined interactive editing capabilities with 3-Degree-of-Freedom (DoF) haptic feedback.

A critical challenge of clay-like materials is the combination of elastic behavior with plastic flow, which is essential for its modeling capabilities. Several works combine global (i.e., mesh-based elastic models) with local deformation (i.e., voxel-based flow models) (Dewaele & Cani, 2003). These models have also been integrated with 3-DoF haptic feedback, and augmented with end-effectors with pressure sensors, to better mimic the interaction of the hand with the clay material (Pihuit et al., 2008).

There are also models that address modeling clay on a potter's wheel, and leverage the revolution shape of objects, but limit general interaction. Some methods use spline surfaces combined with haptic feedback and augmented reality (Han et al., 2007), and others define circular sector elements, which also drive haptic feedback (Lee et al., 2008). It is also possible to add volume preservation to cylindrical elements (Chaudhury & Chaudhuri, 2014).

Despite all the efforts discussed above, the physical behavior of clay is only partially approximated in previous VR applications. No previous interactive simulation method reproduces the highly viscous flow and ductile fracture of clay-like materials. PBD and PBF methods (Bender et al., 2014; Macklin & Müller, 2013; Müller et al., 2006), as discussed above in Section 2.1.2, offer a potential solution due to their efficiency and flexibility.

One possible approach to model clay with PBD would be to consider it an elastic solid. Some works add efficient cutting to viscoelastic PBD solids (Berndt et al., 2017; Xu et al., 2018). However, clay exhibits a fluid-like behavior that allows material to merge, and these methods do not support merging. The other approach to model clay with PBD would be to consider it a viscous fluid. Again some works propose models for efficient viscosity simulation within PBF (T. Takahashi et al., 2014), but they do not reach the extreme viscous behavior of clay.

In Chapter 6 we propose a simulation model that allows the user to conform, split, and merge virtual clay much like in the real world. We leverage on a simplified version of the method presented in Chapter 3, augmented with an elastoplasticity model to capture the main features of clay-like materials. We couple this model with an existing natural hand simulation model to achieve bidirectional coupling, and use the resulting interaction forces to command an ultrasound-based tactile rendering algorithm based on the method presented in Chapter 4.

3 Simulation of Viscoelastic Fluids

Many real-world substances and materials exhibit a viscous fluid or viscoelastic behavior. However, simulation of high viscosity is a computationally challenging problem, since it requires implicit formulations in order to robustly solve the numerically stiff differential equations. In addition, due to the difficulty in computing the strain of a fluid, numerical drift turns into perceptible loss of viscoelasticity. A common alternative to the solution of stiff equations is to model stiff properties as constraints, thereby effectively removing degrees of freedom from the simulation.

In this chapter, we introduce our constraint-based model of viscoelasticity. We first describe a constitutive model of viscoelasticity in polymeric fluids (Section 3.1), which is the stress-based counterpart for our constraint-based formulation. Then, we describe the derivation of implicit conformation constraints acting on fluid velocities, as well as the parameters of our model.

With our constraint-based viscoelasticity model, simulation efficiency is determined by the choice of constrained dynamics solver. In Section 3.2, we propose a *doubly constrained* position-based dynamics (DC-PBD) solver, which inherits the robustness under constraint nonlinearity and the per-iteration efficiency of the original PBD method (Müller et al., 2006), but exhibits improved stability under velocity-based constraints, such as those in our viscoelasticity formulation, specially with large time steps.

As a corollary, while others have also accounted for viscosity in PBD solvers (Alduán et al., 2017; Macklin & Müller, 2013; T. Takahashi et al., 2016; T. Takahashi et al., 2014), our method is derived from a constitutive model, hence it allows physics-based parameterization. Its range of behaviors is also superior, comparable to the one obtained with methods that also compute viscoelastic stress from a discretization of constitutive models, albeit at a much lower computational

cost. In Section 3.3, we describe in detail the integration of our viscoelasticity formulation in a position-based fluids (PBF) model (Macklin & Müller, 2013).

We show results that range from interactive simulation of viscoelastic effects (Fig. 3.7) to large-scale simulation of high viscosity with competitive performance (Fig. 3.10). The materials exhibit the classic buckling and coiling effects produced by viscoelasticity (Fig. 3.9), and we also show how our method can be integrated seamlessly in rich multiphysics scenarios (Fig. 3.1).

3.1 Constitutive Model of Polymer Conformation

In polymeric fluids, the dissolved polymer endows the fluid with viscoelastic properties. Due to friction between the fluid and the polymer, the elasticity of the polymer is transmitted to the fluid, producing the overall viscoelastic behavior (Bird et al., 1977). This behavior can be represented using a constitutive model that relates viscoelastic stress to the change in a polymer conformation tensor $\mathbf{Q}$ (Deshpande et al., 2010).

The polymer conformation model defines a reference value $\bar{\mathbf{Q}} = \mathbf{I}$ for the conformation tensor at rest state, and reduces its time evolution to the solution of a first-order system with relaxation time constant $\tau = \frac{b}{k}$, which amounts to the ratio between the viscous friction b between the polymer and the fluid, and the stiffness k of the polymer. The first-order time evolution of the conformation tensor is defined as:

$$\frac{\mathcal{D}\mathbf{Q}}{\mathcal{D}t} = -\frac{1}{\tau}\left(\mathbf{Q} - \bar{\mathbf{Q}}\right). \tag{3.1}$$

In this equation, the time evolution of the conformation tensor is expressed using the *upper convected derivative* $\frac{\mathcal{D}\mathbf{Q}}{\mathcal{D}t}$, which is a derivative that takes into account local fluid translation and rotation. It is defined as:

$$\frac{\mathcal{D}\mathbf{Q}}{\mathcal{D}t} = \frac{D\mathbf{Q}}{Dt} - \mathbf{Q}\,\nabla\mathbf{u} - (\nabla\mathbf{u})^T\,\mathbf{Q}, \tag{3.2}$$

Fig. 3.1.: A complex multiphysics simulation involving viscoelastic fluids, rigid bodies, and deformable bodies. We simulate whipped cream and strawberry syrup efficiently using our novel viscoelasticity model based on conformation constraints. The complete scene consists of 150,000 particles and runs at 1.13 seconds per frame.

where $\frac{D\mathbf{Q}}{Dt}$ is the standard convective derivative and $\mathbf{u}$ is the fluid velocity. From (3.1) and (3.2), on a Lagrangian setting, the conformation tensor rate can be computed as:

$$\frac{D\mathbf{Q}}{Dt} = \mathbf{Q}\,\nabla\mathbf{u} + (\nabla\mathbf{u})^T\,\mathbf{Q} - \frac{1}{\tau}\left(\mathbf{Q} - \bar{\mathbf{Q}}\right). \tag{3.3}$$

The constitutive model of polymeric fluids is complete with the definition of the viscoelastic stress σ as:

$$\sigma = k\,c\,s\,\left(\mathbf{Q} - \bar{\mathbf{Q}}\right), \tag{3.4}$$

where s is a scale factor that depends on the geometry and structure of the polymer, c is the polymer concentration, and k is the polymer's stiffness, as mentioned above. This stiffness is typically a function of temperature.

By plugging the viscoelastic stress into the equation of fluid momentum conservation, we would obtain an upper convected Maxwell model for a polymeric fluid. Instead, we formulate viscoelasticity as a constraint on the conformation tensor, as we show next.

3.1.1 Implicit Conformation Constraint

To model high viscoelasticity, we propose a constraint that preserves the rest-state value of the conformation tensor, i.e.,

$$\mathbf{C}(\mathbf{Q}) = \mathbf{Q} - \bar{\mathbf{Q}} = \mathbf{Q} - \mathbf{I} = \mathbf{0}. \tag{3.5}$$

On each simulation step, we wish to enforce this constraint implicitly, i.e., on the simulation state at the end of the time step. We do this by combining the constraint equation (3.5) with implicit integration of the conformation tensor rate (3.3). We denote with a superscript 0 variables at the beginning of the time step, e.g., $\mathbf{Q}^0$ is the conformation tensor at the beginning of the time step. Then, the conformation tensor at the end of the time step can be obtained through implicit Euler integration of (3.3), by solving:

$$\frac{\mathbf{Q} - \mathbf{Q}^0}{\Delta t} = \mathbf{Q}\,\nabla\mathbf{u} + (\nabla\mathbf{u})^T\,\mathbf{Q} - \frac{1}{\tau}\left(\mathbf{Q} - \bar{\mathbf{Q}}\right). \tag{3.6}$$

Plugging the tensor constraint (3.5) in this equation, we obtain an implicit conformation constraint on fluid velocities:

$$\mathbf{C}(\mathbf{u}) = \mathbf{Q}^0 - \bar{\mathbf{Q}} + \Delta t \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right) = \mathbf{0}. \tag{3.7}$$

Our simulation loop for viscoelastic fluids proceeds on each time step by computing a dynamic update subject to the implicit velocity-based constraint (3.7). For this purpose, we use the constrained dynamics solver proposed in Section 3.2. At the end of each time step, we update the conformation tensor by solving for $\mathbf{Q}$ in (3.6). In the particular case when $\tau \to 0$, we simply set the tensor to be $\mathbf{Q} = \mathbf{I}$.

The relaxation time τ affects the viscoelasticity behavior of the fluid. As $\tau \to 0$, the internal elastic forces of the polymer dominate over friction forces with the fluid ($k \gg b$), and the polymer recovers quickly its structure. In the viscoelastic fluid simulation case, the conformation tensor $\mathbf{Q}$ barely changes over time, and the conformation constraint (3.7) becomes effectively a null-strain-rate constraint. The fluid appears highly viscous. Conversely, as $\tau \to \infty$, the friction forces of the polymer with the fluid dominate over its internal elastic forces ($b \gg k$), and the polymer fails to recover its structure. In the viscoelastic fluid simulation, the conformation tensor $\mathbf{Q}$ varies over time as a result of the strain rate according to (3.3), and the conformation constraint (3.7) acts on fluid velocities to remove the existing conformation change. The fluid appears elastic. In the constitutive model of polymer conformation, the viscoelastic stress depends also on a compliance $\alpha = \frac{1}{k c s}$ according to (3.4).

Later in Section 3.3, we show how to incorporate this compliance into our constrained dynamics solver as a relaxation coefficient for the conformation constraint (3.7). The compliance α defines the fluidity of the model. With two physically based parameters, τ and α, we obtain a palette of materials that spans elastoplastic, highly viscous, and inviscid fluids, as shown in Fig. 3.2. The influence of material parameters on the fluid's behavior is evidenced even on simple scenarios, such as impact (Fig. 3.3) or liquid rope coiling (Fig. 3.4). In these examples, material colors are chosen by interpolating the colors in the inset according to the parameter settings. Even on moderately complex scenes (more than $10\,000$ particles), the diverse materials can be simulated interactively.

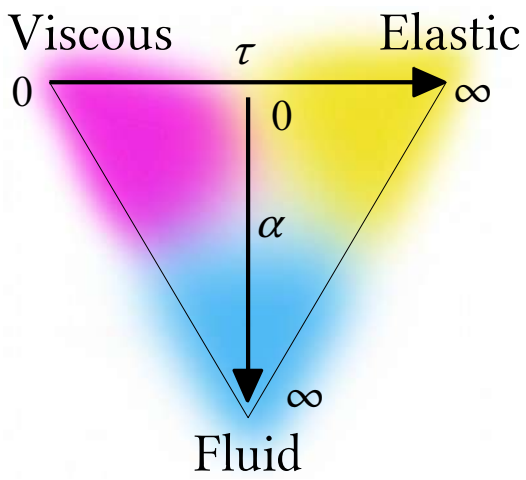


Fig. 3.2.: With two physically based parameters, τ and α, we obtain a palette of materials that spans elastoplastic, highly viscous, and inviscid fluids.

Fig. 3.3.: Interactive viscoelastic cubes are dropped on the ground. By varying two physics-based parameters, the conformation relaxation time constant τ and the compliance α, we achieve materials that bounce elastically **(yellow)**, appear highly viscous **(magenta)**, or splash as an inviscid liquid **(cyan)**.

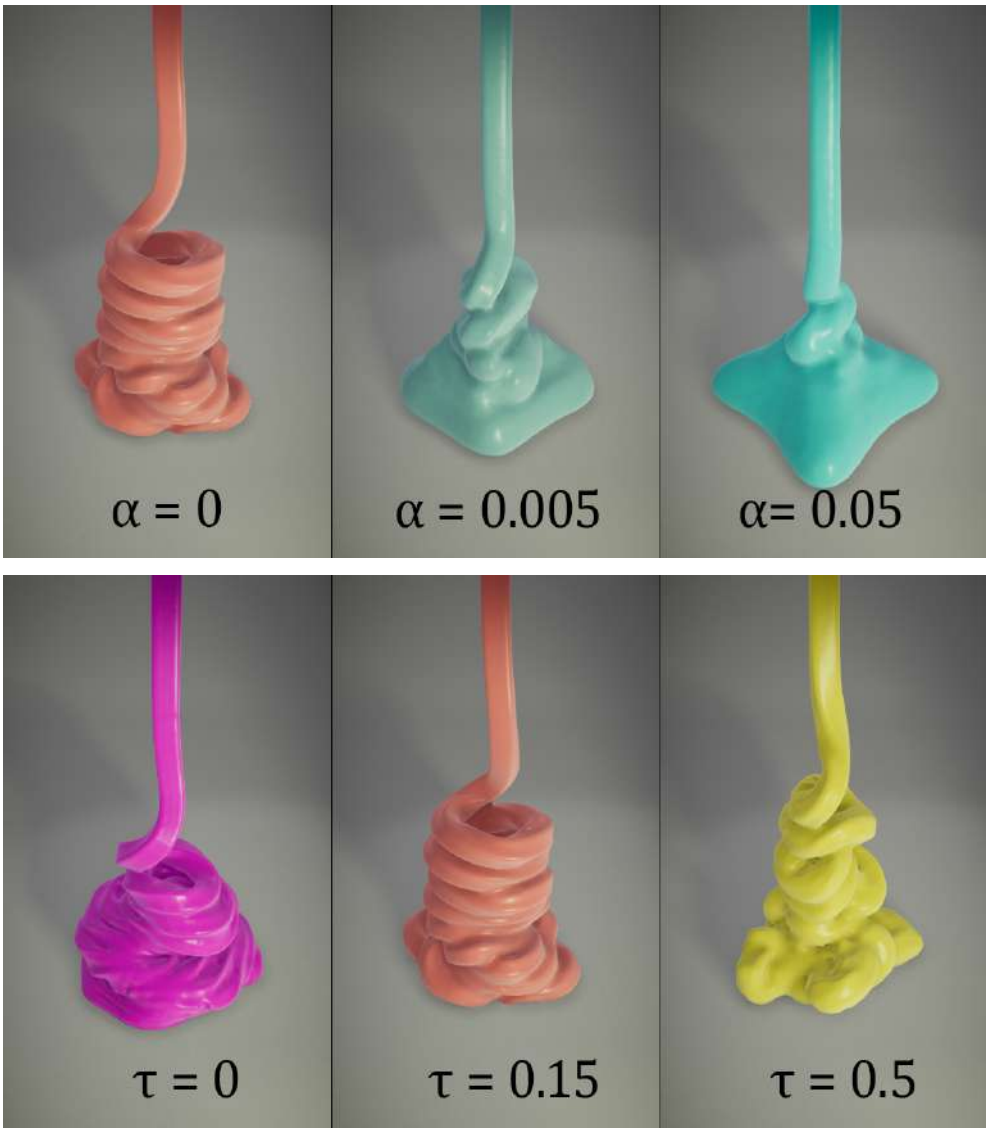


Fig. 3.4.: We compare liquid rope coiling with different material parameters. In the top row, varying the compliance α, with relaxation time constant $\tau = 0.15$ in all cases. In the bottom row, varying τ, with $\alpha = 0$ in all cases. The examples are colored by interpolating the material color palette shown in Fig. 3.3.

3.2 Constrained Dynamics Solver

As discussed in Chapter 2, PBD can be regarded as an integration scheme for constrained dynamics. In this section, we propose a doubly constrained PBD (DC-PDB) solver that handles also velocity-based constraints, such as those in our viscoelasticity model. DC-PBD projects both velocities and positions to the velocity-based constraints, and in this way it improves convergence and stability. We start the section with a summary of the regular PBD solver, and then we describe the differences in our DC-PDB approach.

3.2.1 PBD Solver

Given a dynamic system with mass $\mathbf{M}$, external forces $\mathbf{f}$, and position-based constraints $\mathbf{C}^{\mathbf{x}}$, the PBD method executes each simulation step as follows. Starting from positions and velocities $(\mathbf{x}^0, \mathbf{u}^0)$, it first computes a constraint-free state $(\mathbf{x}^*, \mathbf{u}^*)$ using symplectic Euler integration. Then, it projects the positions to the constraints, and computes velocities through finite differences between final and initial positions, to obtain the state $(\mathbf{x}, \mathbf{u})$ at the end of the time step. The PBD solver is summarized in Algorithm 1.

PBD succeeds to robustly and efficiently model stiff potentials as position-based constraints. Moreover, the recent XPBD extension adds relaxation to the constraint projection in order to model constraint compliance. However, PBD is not naturally designed to handle efficiently velocity-based constraints, such as the viscoelasticity constraint (3.7). In addition, constraint nonlinearity, such as the one introduced by the SPH kernels used in PBF, may complicate the convergence of Jacobi or Gauss-Seidel solvers.

3.2.2 Doubly Constrained PBD

Given the generic dynamic system described above, we propose a constrained dynamics algorithm that handles both position-based constraints $\mathbf{C}^{\mathbf{x}}$ and velocity-

ALGORITHM 1: PBD step

Input: Initial state $(\mathbf{x}^0, \mathbf{u}^0)$.
Output: Updated state $(\mathbf{x}, \mathbf{u})$.

Compute constraint-free state
$\mathbf{u}^* \leftarrow \mathbf{u}^0 + \Delta t\, \mathbf{M}^{-1}\, \mathbf{f}(\mathbf{x}^0)$
$\mathbf{x}^* \leftarrow \mathbf{x}^0 + \Delta t\, \mathbf{u}^*$

Project positions
$\mathbf{x} \leftarrow$ project $\mathbf{x}^*$ to $\mathbf{C}^{\mathbf{x}}(\mathbf{x}) = \mathbf{0}$
$\mathbf{u} \leftarrow \frac{\mathbf{x}-\mathbf{x}^0}{\Delta t}$

based constraints $\mathbf{C}^{\mathbf{u}}$ efficiently. The key difference is to project both positions and velocities to the constraints. A similar idea is applied by the RATTLE algorithm for molecular dynamics (Andersen, 1983), which is a constrained-dynamics version of the velocity-Verlet integrator. But, unlike RATTLE, we exploit the robustness of PBD by prioritizing the projection of positions, and computing velocity estimates through finite differences of safe positions.

We start the DC-PBD solver by computing a constraint-free state $(\mathbf{x}^*, \mathbf{u}^*)$ using symplectic Euler integration, same as in regular PBD. Then, we project the velocities to the velocity-based constraints $\mathbf{C}^{\mathbf{u}}$, and we update particle positions with the resulting velocity correction. Altogether, we obtain a velocity-safe state $(\mathbf{x}^{**}, \mathbf{u}^{**})$.

To conclude, we compute the final, position-safe state $(\mathbf{x}, \mathbf{u})$, by projecting the positions to both the position-based and velocity-based constraints. To do this, we need to transform the velocity-based constraints $\mathbf{C}^{\mathbf{u}}$ into position-based constraints, which are added to the regular position-based constraints $\mathbf{C}^{\mathbf{x}}$. We define the velocities through finite differences between the initial and final positions, i.e., $\mathbf{u} = \frac{\mathbf{x}-\mathbf{x}^0}{\Delta t}$, and thus we can turn the velocity-based constraints into position-based constraints of the form $\tilde{\mathbf{C}}^{\mathbf{u}}(\mathbf{x}) \equiv \mathbf{C}^{\mathbf{u}}\left(\frac{\mathbf{x}-\mathbf{x}^0}{\Delta t}\right) = \mathbf{0}$.

The DC-PBD solver is summarized in Algorithm 2. We compute the position projection using Fast Projection, just like in regular PBD. The velocity projection, on the other hand, is a positive semi-definite linear problem, similar in structure to the one tackled by Peer et al. (2015). Same as they do for generic cases, we solve it using Jacobi iterations. However, unlike the position projection, where Jacobians are recomputed after each Jacobi iteration, in the velocity projection the

Jacobians are constant and can be computed just once per time step. We provide full details of the application of the DC-PBD solver to viscoelasticity constraints in Section 3.3.

ALGORITHM 2: DC-PBD step

Input: Initial state $(\mathbf{x}^0, \mathbf{u}^0)$.
Output: Updated state $(\mathbf{x}, \mathbf{u})$.

Compute constraint-free state
$\mathbf{u}^* \leftarrow \mathbf{u}^0 + \Delta t\, \mathbf{M}^{-1}\, \mathbf{f}(\mathbf{x}^0)$
$\mathbf{x}^* \leftarrow \mathbf{x}^0 + \Delta t\, \mathbf{u}^*$

Project velocities
$\mathbf{u}^{**} \leftarrow$ project $\mathbf{u}^*$ to $\mathbf{C}^{\mathbf{u}}(\mathbf{u}^{**}) = \mathbf{0}$
$\mathbf{x}^{**} \leftarrow \mathbf{x}^* + \Delta t\ (\mathbf{u}^{**} - \mathbf{u}^*)$

Project positions
$\mathbf{x} \leftarrow$ project $\mathbf{x}^{**}$ to $\mathbf{C}^{\mathbf{x}}(\mathbf{x}) = \mathbf{0}$ and $\tilde{\mathbf{C}}^{\mathbf{u}}(\mathbf{x}) \equiv \mathbf{C}^{\mathbf{u}}\left(\frac{\mathbf{x}-\mathbf{x}^0}{\Delta t}\right) = \mathbf{0}$
$\mathbf{u} \leftarrow \frac{\mathbf{x}-\mathbf{x}^0}{\Delta t}$

To evaluate our DC-PBD solver, we have run a test where we drop on the ground a cube with full viscosity constraints and no compliance (Fig. 3.3, with $\tau = 0$ and $\alpha = 0$). We have computed the RMS error of particles in the cube w.r.t. their undeformed positions, as a global measure of constraint drift. We have compared constraint drift with our DC-PBD solver vs. position projection of both position-based and velocity-based constraints (which can be regarded as an improved version of regular PBD). Position projection alone requires a time step smaller than 30 ms to ensure stability. With our DC-PBD solver, on the other hand, the simulation remains stable with time steps twice as large, i.e., 60 ms. We have also found that the number of Jacobi iterations affects the amount of constraint drift (with lower drift in DC-PBD under the same total iteration count, as shown in the plots in Fig. 3.5), but it has little effect on stability.

Our conclusions about the reasons for the improved stability and robustness of DC-PBD are the following. Projection of particle positions is nonlinear, and nonlinear Jacobi may have trouble converging under large time steps. Projection of velocities, on the other hand, is linear, and linear Jacobi turns out more robust. The position correction in the "Project velocities" step in Algorithm 2 removes much of the position deviation, and further steps of nonlinear position projection are less prone to robustness problems induced by nonlinearity.

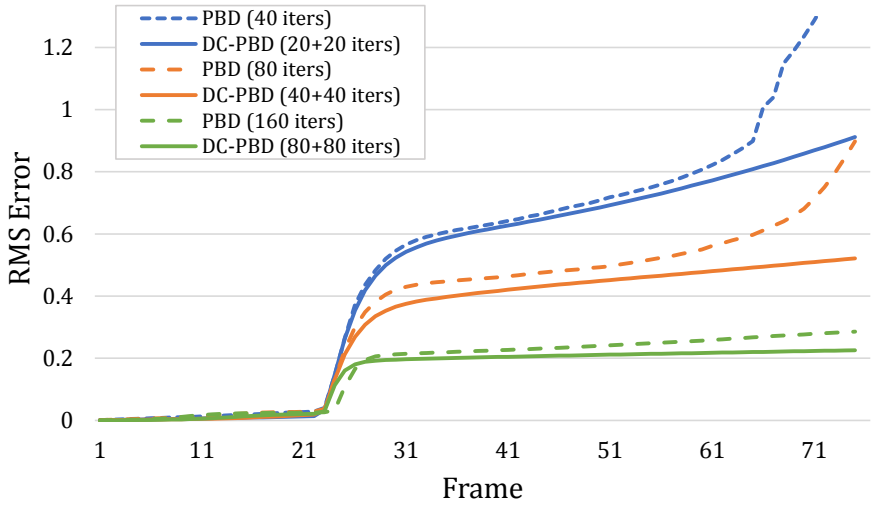


Fig. 3.5.: Comparison of constraint drift with our DC-PBD solver (with velocity and position projection) vs. position projection alone. We drop a fully viscous cube on the ground (Fig. 3.3), and we measure the RMS error of particle positions w.r.t. an undeformed cube as the simulation evolves. Position projection alone suffers higher error under the same total iteration count, and it requires a smaller time step to be stable (30 ms, vs. 60 ms in the case of DC-PBD).

3.3 Viscoelastic Position-Based Fluids

We integrate conformation constraints in a PBF model to simulate viscoelastic incompressible fluids. However, unlike the original PBF model, we employ our DC-PBD solver for improved convergence, and we adopt XPBD to support constraint compliance. We start this section by outlining the complete simulation model, and we then describe in detail how to handle conformation constraints for velocity and position projection respectively.

3.3.1 PBF Model

PBF (Macklin & Müller, 2013) simulates fluids using a SPH discretization (Monaghan, 1992), with incompressibility as a density constraint on particle positions. Following the SPH discretization, given a set of particles, each one with mass m_j,

attribute value a_j, and position $\mathbf{x}_j$, the value of the attribute a at an arbitrary position $\mathbf{x}_i$ is evaluated as:

$$a(\mathbf{x}_i) = \sum_j \frac{m_j}{\rho_j} a_j W_{ij}, \tag{3.8}$$

with $W_{ij} = W(\mathbf{x}_{ij}) = W(\mathbf{x}_i - \mathbf{x}_j)$ being the evaluation at $\mathbf{x}_i$ of a smoothing kernel with support radius h and centered at $\mathbf{x}_i$, and ρ_j the density field evaluated at $\mathbf{x}_j$. We employ the SPH-based attribute evaluation for the computation of the fluid velocity in conformation constraints (3.7).

Combining incompressibility and viscoelasticity, the DC-PBD solver proceeds as follows. For the position projection step of DC-PBD, we enforce both density and conformation constraints on particle positions. In Section 3.3.3, we describe how we formulate position-based conformation constraints per simulation particle. We have experimented with various strategies to combine incompressibility and viscoelasticity in the position projection, and we have observed best convergence by staggering one Jacobi iteration of incompressibility over all particles with one Jacobi iteration of viscoelasticity over all particles.

For the velocity projection step of DC-PBD, we enforce only conformation constraints on particle velocities, as we describe next in Section 3.3.2. Once this is done, we update the conformation tensor $\mathbf{Q}$ on each particle by solving the linear system (3.6).

Both for the position and velocity projection steps, we adopt the XPBD method (Macklin et al., 2016), which modifies the original PBD iterations to support constraint compliance, as also done in other constrained dynamics methods (Tournier et al., 2015). In our viscoelasticity model, constraint compliance allows us to account for the compliance α of viscoelastic stress defined in Section 3.1.1.

3.3.2 Discrete Velocity-Based Constraints

To implement the velocity-based, implicit conformation constraint (3.7) within the PBF framework, we express the constraint on each simulation particle. To do this,

we compute the fluid velocity $\mathbf{u}_i$ at the position of the ith particle using the SPH formulation (3.8) and, accordingly, we evaluate the SPH velocity gradient:

$$\nabla \mathbf{u}_i = \sum_j \frac{m_j}{\rho_j} \mathbf{u}_{ji} \nabla W_{ij}^T, \tag{3.9}$$

with $\mathbf{u}_{ji} = \mathbf{u}_j - \mathbf{u}_i$. Since the conformation tensor $\mathbf{Q}$ is symmetric, we rearrange it as a six-dimensional vector $\mathbf{q}$:

$$\mathbf{q} = (Q_{xx}, Q_{yy}, Q_{zz}, Q_{xy}, Q_{xz}, Q_{yz})^T; \qquad \bar{\mathbf{q}} = (1, 1, 1, 0, 0, 0)^T.$$

Rewriting the conformation constraint (3.7) using the vector notation, and plugging in the expression of the velocity gradient (3.9), we obtain the discrete version of the velocity-based constraint:

$$\mathbf{C}_i(\mathbf{u}) = \mathbf{q}_i^0 - \bar{\mathbf{q}} + \Delta t \sum_j \frac{m_j}{\rho_j} \mathbf{A}_{ij} \mathbf{u}_{ji} = \mathbf{0}, \tag{3.10}$$

$$\text{with } \mathbf{A}_{ij} = \begin{pmatrix} 2\partial_x W_{ij} & 0 & 0 \\ 0 & 2\partial_y W_{ij} & 0 \\ 0 & 0 & 2\partial_z W_{ij} \\ \partial_y W_{ij} & \partial_x W_{ij} & 0 \\ \partial_z W_{ij} & 0 & \partial_x W_{ij} \\ 0 & \partial_z W_{ij} & \partial_y W_{ij} \end{pmatrix}.$$

We apply this constraint to each simulation particle in the velocity-projection step of our DC-PBD solver (see Section 3.2.2). Each Jacobi iteration with XPBD yields the following update of Lagrange multipliers and particle velocities, respectively:

$$\Delta \lambda_i = \left(\text{diag}\left(\mathbf{J}_i \mathbf{M}^{-1} \mathbf{J}_i^T \right) + \frac{\alpha}{\Delta t} \mathbf{I} \right)^{-1} \left(-\mathbf{C}_i(\mathbf{u}) - \frac{\alpha}{\Delta t} \lambda_i \right), \tag{3.11}$$

$$\Delta \mathbf{u}_i = \frac{\beta}{n_i} \sum_j \mathbf{J}_{ji}^T \Delta \lambda_j. \tag{3.12}$$

We relax the Jacobi update with a factor $\frac{\beta}{n_i}$ to ensure convergence, where n_i is the size of the particle neighborhood and β is a scaling coefficient to avoid excessive

relaxation ($\beta = 5$ in our examples). In our implementation, we approximate the constraint Jacobians as:

$$\mathbf{J}_{ik} = \frac{\partial \mathbf{C}_i(\mathbf{u})}{\partial \mathbf{u}_k} = \begin{cases} \Delta t \, \frac{m_k}{\rho_k} \, \mathbf{A}_{ik} & \text{if } i \neq k \\ -\Delta t \, \sum_j \frac{m_j}{\rho_j} \, \mathbf{A}_{ij} & \text{if } i = k. \end{cases} \tag{3.13}$$

Note that these Jacobians remain constant during the whole velocity projection. The constraint response update (3.11) includes the compliance α defined in Section 3.1.1. In the original XPBD formulation (Macklin et al., 2016) the compliance is applied to constraints formulated on positions, and hence it is scaled by a factor $\frac{1}{\Delta t^2}$. In our setting, with constraints formulated on velocities, it is scaled by $\frac{1}{\Delta t}$ instead. Fig. 3.4 and Fig. 3.9 show the effect of varying the compliance on two different examples.

3.3.3 Discrete Position-Based Constraints

For the position-projection step of our DC-PBD solver, we wish to rewrite the discrete velocity-based conformation constraint (3.10) as a function of particle positions, i.e., in the form $\tilde{\mathbf{C}}_i(\mathbf{x}) = 0$. In principle, we could do this by approximating particle velocities through finite differences of particle positions, $\mathbf{u}_j = \frac{\mathbf{x}_j - \mathbf{x}_j^0}{\Delta t}$. However, this approach would fail to preserve angular momentum and would damp rotational motion of the fluid. At the core of the problem lies the inability of SPH to correctly reconstruct linear fields (e.g., uniform angular velocity). A similar observation was made by Becker et al. (2009) for the computation of deformation gradients and, similar to their corotational deformation gradient, we derive a corotational formulation of the velocity gradient (3.9) from particle positions. Fig. 3.6 shows an example simulation of a rotating block with and without our corotational discretization. The difference in rotational damping is evident.

Following Becker et al., 2009, we estimate for each particle the best-fit rotation $\mathbf{R}_i$ to the deformations of its neighbor particles:

$$\mathbf{R}_i = \arg\min \sum_j \left| \mathbf{R}_i \left(\mathbf{x}_j^0 - \mathbf{c}_i^0 \right) - \left(\mathbf{x}_j - \mathbf{c}_i \right) \right|^2, \tag{3.14}$$

where $\mathbf{c}_i$ is the center of mass of the neighbor particles. Then, for the estimation of the velocity gradient of the i^{th} particle, we define corotational finite-difference velocities for all particles in its neighborhood, by compensating for the rotation $\mathbf{R}_i$. Specifically:

$$\mathbf{u}_j = \frac{\mathbf{x}_j - \mathbf{x}_j^r}{\Delta t}, \qquad \text{with } \mathbf{x}_j^r = \mathbf{c}_i + \mathbf{R}_i \left(\mathbf{x}_j^0 - \mathbf{c}_i^0\right). \tag{3.15}$$

By substituting this velocity computation in the velocity gradient (3.9), we rewrite (3.10) to obtain the position-based expression of the discrete conformation constraint:

$$\tilde{\mathbf{C}}_i(\mathbf{x}) = \mathbf{q}_i^0 - \bar{\mathbf{q}} + \sum_j \frac{m_j}{\rho_j} \mathbf{A}_{ij} \left(\mathbf{x}_{ji} - \mathbf{x}_{ji}^r\right) = \mathbf{0}, \tag{3.16}$$

We apply this constraint to each simulation particle in the position-projection step of our DC-PBD solver (see Section 3.2.2). Each Jacobi iteration with XPBD yields the following update of Lagrange multipliers and particle positions, respectively:

$$\Delta\lambda_i = \left(\text{diag}\left(\tilde{\mathbf{J}}_i \mathbf{M}^{-1} \tilde{\mathbf{J}}_i^T\right) + \frac{\alpha}{\Delta t^2}\mathbf{I}\right)^{-1} \left(-\tilde{\mathbf{C}}_i(\mathbf{x}) - \frac{\alpha}{\Delta t^2}\lambda_i\right), \tag{3.17}$$

$$\Delta\mathbf{x}_i = \frac{\beta}{n_i} \sum_j \tilde{\mathbf{J}}_{ji}^T \Delta\lambda_j. \tag{3.18}$$

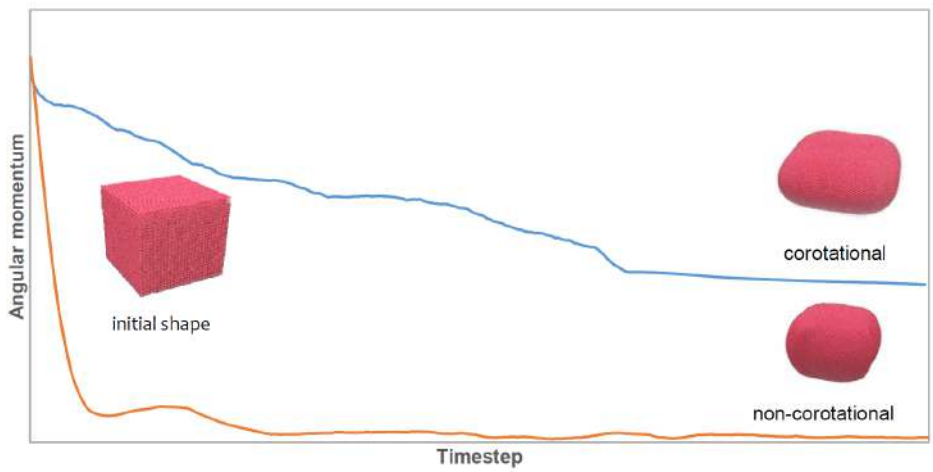


Fig. 3.6.: Plot of angular momentum of a rotating block, with and without our corotational formulation. With a non-corotational velocity gradient, rigid body motion is soon damped. With our approach, residual damping is due only to approximation errors.

The Jacobians are defined as $\tilde{\mathbf{J}}_i = \frac{1}{\Delta t}\mathbf{J}_i$, and they need to be recomputed after each Jacobi update of particle positions.

3.4 Results

We have tested our viscoelasticity model on multiple benchmarks. They were all executed on a Hexa-core Intel i7-3930K CPU with 32 GB of RAM, and a NVIDIA GeForce 1070 GTX GPU with 1920 CUDA Cores. The PBF model with the DC-PBD solver is programmed entirely on the GPU. Table 3.1 shows the major performance statistics and parameter settings for all the benchmarks (all rendered at 30 fps). Next, we discuss the results.

Scene & Fig	Particles	Time step	Steps/frame	Iters	Time/frame	τ	α
Blocks (Fig. 3.3)	10k	1/240	8	20	0.16s	See Fig.	
Coiling (Fig. 3.4)	89k	1/240	8	40	1.08s	See Fig.	
Interactive (Fig. 3.7, Fig. 3.8)	15k	1/200	3	8	0.03s	0.1	$0 \leq \alpha \leq 1$
Waffle (Fig. 3.9)	80k	1/300	10	40	1.42s	0.1	$0^{\dagger}$, $0.01^{\dagger\dagger}$
Honey (Fig. 3.11)	105k	1/300	10	10	0.15s	0	0.005
Cake (Fig. 3.1)	150k	1/750	25	15	1.13s	$0.1^{\dagger}$, $0.5^{\ddagger}$	$0^{\dagger}$, $0.015^{\ddagger}$
Armadillos (Fig. 3.10)	12M	1/150	5	15	19s	0.5	0.01

Tab. 3.1.: Parameter values and performance statistics for all our benchmarks (all rendered at 30 fps). The table lists: the total number of particles, the time step Δt, the amount of steps per frame, the number of iterations of the viscoelasticity constraint projection per step, the total time per frame (in seconds), and τ and α values. Some materials: † thick cream; †† runny cream; ‡ strawberry syrup.

Interactive scenes

The scenes shown in Fig. 3.7 and Fig. 3.8 demonstrate the suitability of the DC-PBD solver in interactive applications where very small time steps cannot be used. The improved convergence and stability enables even interactive performance on moderately complex scenarios. Both the hand-and-bowl scene and the ice cream scene consist of 15k particles each. In these examples, we use a Leap Motion™ device to track hand motions and move a virtual hand or other objects. The tests also show interactive modification of the material parameters, e.g., increasing the compliance α to model ice cream melting.

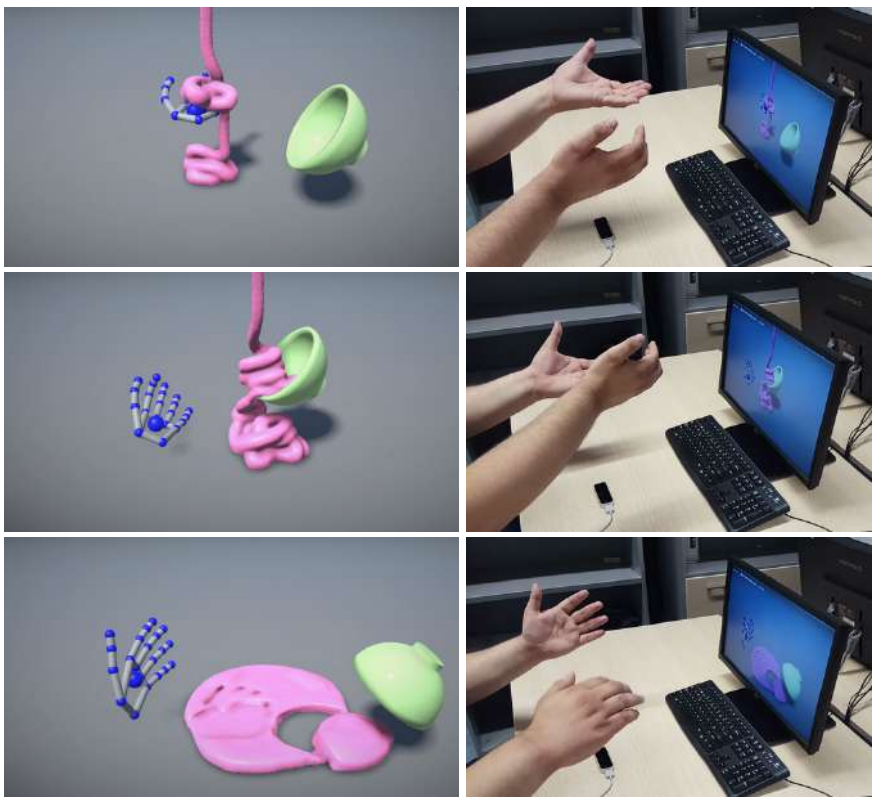


Fig. 3.7.: Three screen captures of interactive manipulation of a viscoelastic fluid, consisting of up to 15k particles, and simulated at 30 ms/frame. The motion of the hands is tracked using a Leap Motion™ device, and this motion is applied to a virtual hand and a bowl, which interact with the coiling fluid. We also demonstrate interactive changes to material parameters.

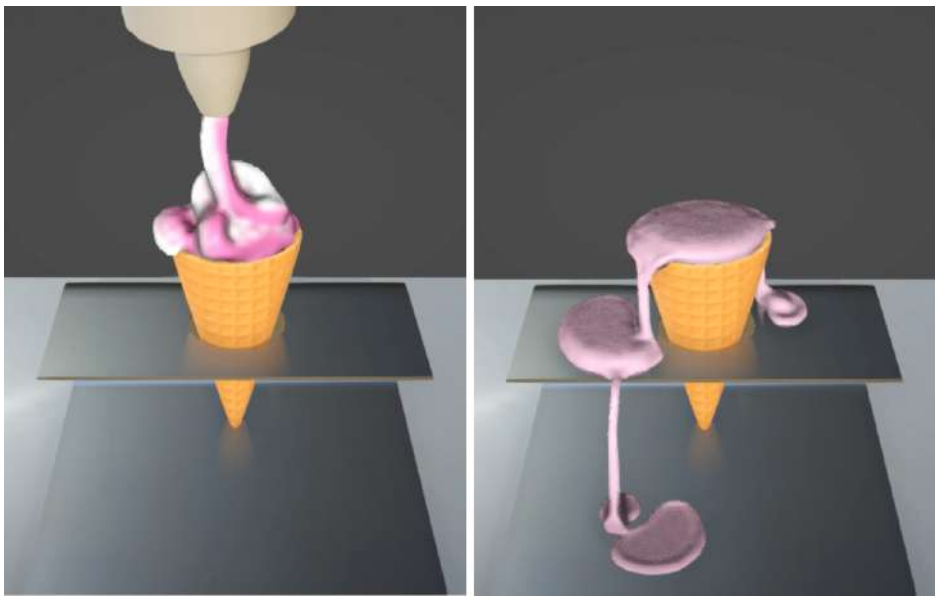


Fig. 3.8.: Screen captures of interactive ice cream simulation. The dispenser is controlled interactively through a Leap Motion™ device, and ice cream is poured into the cone. Increasing the compliance α, the ice cream melts. The scene consists of up to 15k particles, simulated at 30 ms/frame.

Fig. 3.9.: Two types of whipped cream are poured onto a waffle, with compliance ($\alpha = 0.01$) on the left, and without compliance on the right. High viscosity in the right causes a regular coiling effect.

Large-scale simulations

The scenes shown in Fig. 3.9, Fig. 3.10 and Fig. 3.11 represent several computationally demanding benchmarks with complex behaviors. The simulation of whipped cream poured onto a waffle (Fig. 3.9) runs up to 90k particles with a computation time of 1.42 seconds per frame. This simulation requires high particle density to correctly resolve the ridges on the cream's surface. We compare two scenarios, with the same value of relaxation time τ, but with different compliance α. With increased viscoelasticity (i.e., lower α), the cream coils in a regular manner.

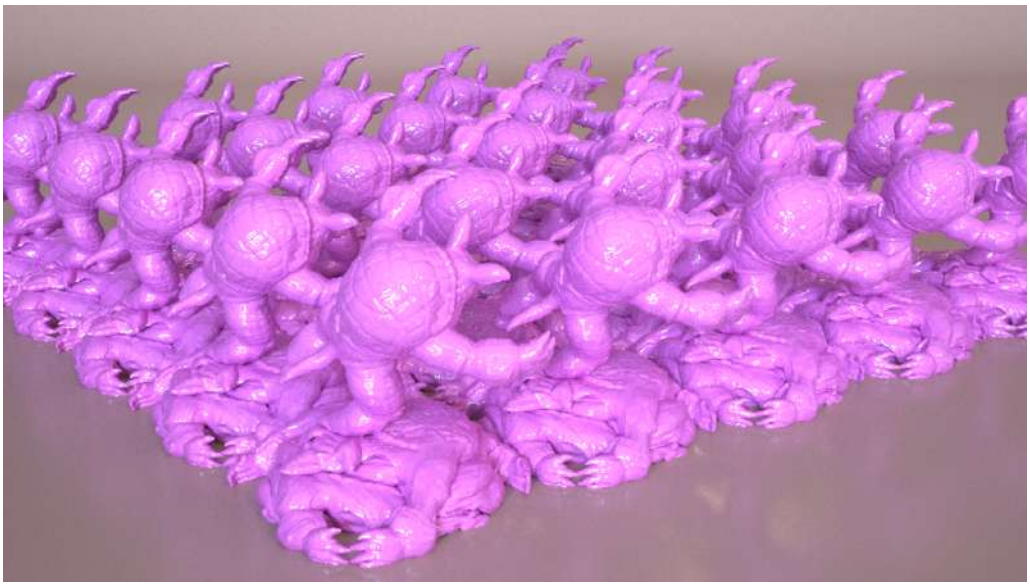


Fig. 3.10.: Massive viscoelastic simulation, with 12 million particles simulated at 19 seconds per frame. This example demonstrates that our viscoelasticity model achieves higher performance than previous methods, even on large-scale scenes.

The massive test shown in Fig. 3.10 involves the computation of high viscoelasticity on a large-scale scene, which replicates a benchmark tested by Peer et al., 2015. They simulated 11 million particles at 144 seconds per frame and 50 fps. We simulate 12 million particles at 19 seconds per frame and 30 fps. Prorating particle count and frame-rate, our method achieves a speed-up of more than 13x, showing that our viscoelasticity model provides superior performance to previous approaches even on large-scale scenes. Furthermore, the simultaneous treatment of density and viscoelasticity constraints within our solver reduces the possibility of

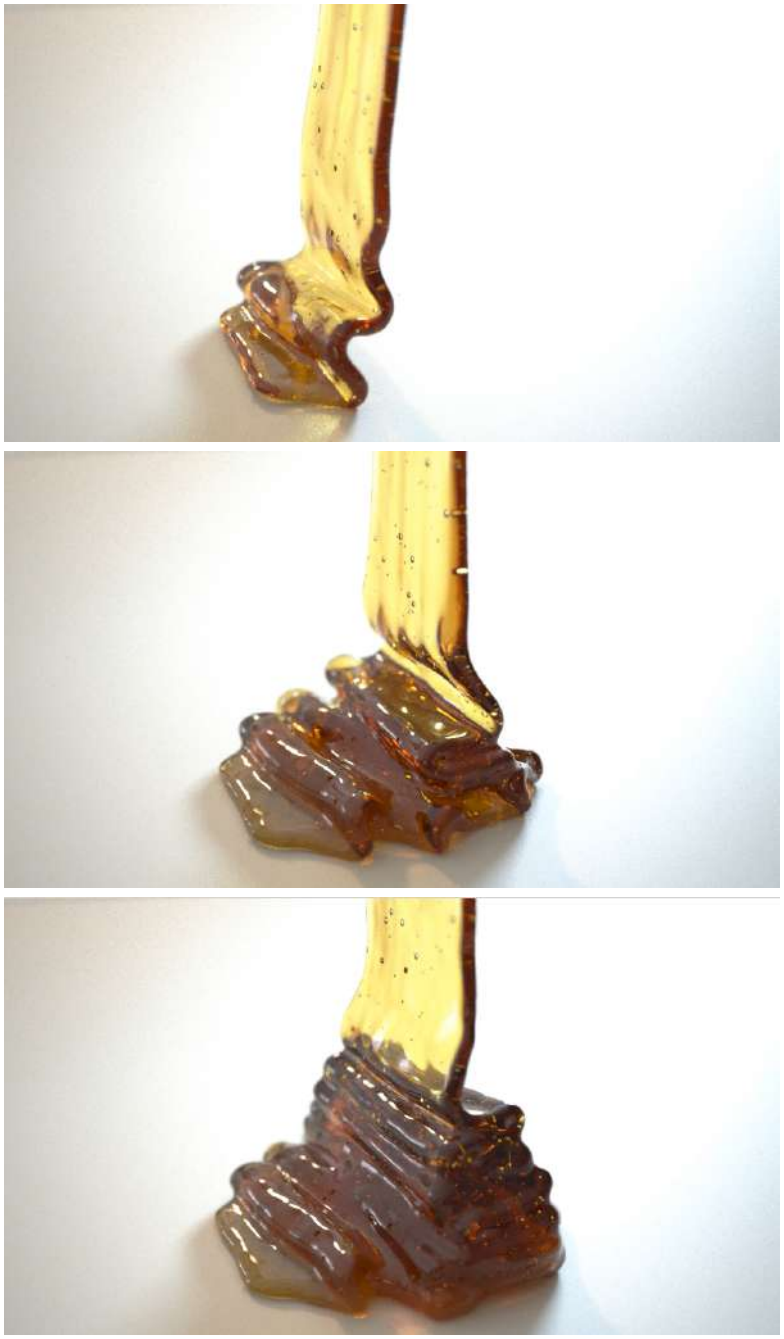


Fig. 3.11.: Simulation of honey using 105k particles of viscous fluid. This test runs at 0.14 seconds per frame on average, and exhibits the characteristic buckling of highly viscous materials.

incompressibility drift (cf. Fig. 3.12). Note that the iterative Jacobi or Gauss-Seidel solvers of PBD-type methods become particularly slow at such high resolutions, but our method achieves competitive performance. Performance would suffer more, both with our method and with the one by Peer et al., 2015, on taller hydrostatic columns.

The pouring honey in Fig. 3.11 is simulated using 105k particles of viscous fluid, rendered with translucent material. This test runs at 0.14 seconds per frame on average, and exhibits the characteristic buckling of highly viscous materials.

Multiphysics simulation

One of the features of the PBD-type constrained dynamics solvers is that they can easily accommodate objects and materials with diverse properties. In the scene shown in Fig. 3.1, we simulate two types of viscoelastic materials (whipped cream and strawberry syrup) using our conformation constraints, rigid chocolate letters using shape-matching constraints, and soft flowers using distance constraints. The complete scene consists of up to 150k particles and is simulated in 1.13 seconds per frame.

Discussion

In our examples, we have demonstrated that the proposed viscoelasticity model covers efficiently a broad set of simulation scenarios, from interactive scenes to large-scale scenes.

To the best of our knowledge, we have shown unprecedented viscous and viscoelastic interactive simulations, with a combination of high viscosity and scene complexity (i.e., particle count) not possible before. Our solution also outperforms previous methods on large-scale scenes, even though the type of constraint solver may not be a priori best suited for such scenes. A key feature for the performance of our solution is the efficiency and robustness of the constraint solver, which allows time steps of moderate size, few iterations per time step, and massive parallelization within each iteration.

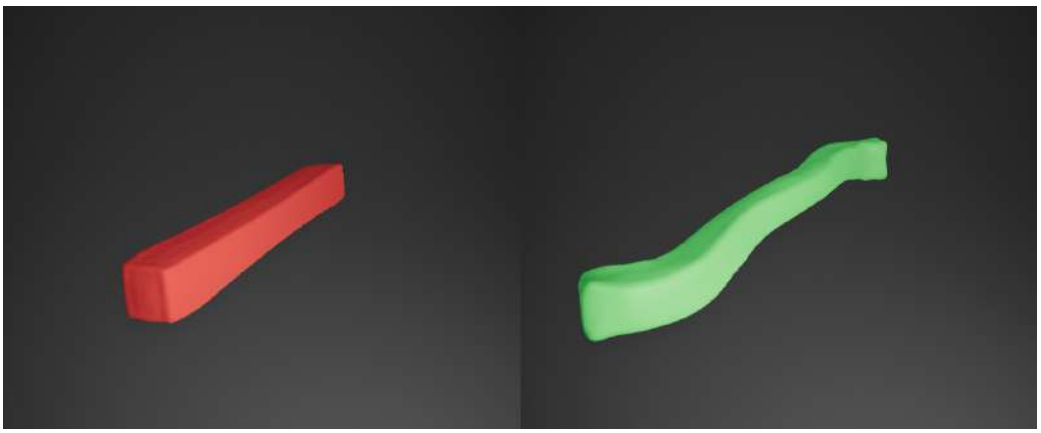


Fig. 3.12.: A beam of highly viscous material, rotating around the vertical axis in absence of external forces, using our method **(left)** and a PBF simulation with a viscosity method based on that of Peer et al., 2015 **(right)**. The method of Peer et al. constrains velocities successfully, but cannot remove position drift in a PBF simulation.

Some viscosity models advocate defining viscous stress as a function of shear rate (Peer et al., 2015; B. Zhu et al., 2015). In the purely viscous case, our model converges instead to a null-strain-rate constraint, as discussed in Section 3.1.1. The difference w.r.t. a null-shear-rate constraint produces a hydrostatic stress, which acts against density changes. We have tested using shear rate constraints in our model, and we have validated that the hydrostatic stress acts as a damping term on density, and it helps the convergence of the incompressibility constraint. For compressible liquids, we could perhaps add higher compliance to the hydrostatic part of the conformation constraint, but we have not explored this avenue.

3.5 Discussion and Future Work

In this chapter, we have presented a novel model of viscoelasticity for fluid simulation. Our model formulates viscoelasticity using constraints, and can be solved efficiently within the PBD constrained dynamics framework. We have designed viscoelasticity constraints inspired by a constitutive model of viscoelasticity for polymeric fluids, which employs a conformation tensor with simple treatment of purely viscous vs. elastic effects. To enable efficient and robust simulation,

we formulate the constraints implicitly, and we describe their integration in the state-of-the-art XPBD solver, with further improvements.

Our DC-PBD solver might be applicable to other types of constraints, beyond those handled in our work. One such example is friction, which shares a dissipative nature with viscosity, but incorporates constraints on the deviatoric stress. Another example is incompressibility. Similar to the divergence-free SPH method by Bender and Koschier, 2017, incompressibility constraints could be applied on both positions and velocities within our DC-PBD solver.

Despite the rich effects achieved with our method and the range of materials that can be simulated, there are still some limitations that suggest interesting future work. Our work inherits some of the generic limitations of the PBD and PBF approach, in particular the convergence limitations of Jacobi or Gauss-Seidel solvers. It would be interesting to take advantage of the connection between PBD and minimization formulations of implicit integration, to explore efficient optimization algorithms, as done by others after the projective dynamics method (Bouaziz et al., 2014). However, those optimization algorithms cannot be trivially extended to fluids as they make strong connectivity assumptions (Weiler et al., 2016).

Our method cannot handle large elastic deformations accurately, which would require storing some explicit measure of rest state. Additionally, while some of our examples demonstrate the simulation of fine features, the ability to resolve such fine features is eventually limited by the particle resolution of the simulation. Fusing codimensional representations (B. Zhu et al., 2015) with constraint-based viscoelasticity would enable even richer effects under manageable computational cost.

To conclude, even though our method is parameterized using two physics-based parameters, it is difficult to design them purely from measurable physical quantities in a discretization-independent manner. Our model is derived from a constitutive model for polymeric fluids, and the parameters could be set from geometric and physical quantities only for such fluids. However, we apply the model to other types of viscoelastic fluids too, and in that case the model can be regarded as empirical or phenomenological, and parameters could be estimated from measurements. In our examples, we have opted for an artist-driven parameter design, which nevertheless proves effective thanks to the narrow set of parameters. Another problem of the

parameterization is that many details of the constitutive model are reduced to just one parameter, the constraint compliance. This limits the ability to represent non-Newtonian fluids with complex dependence on any of these material parameters, e.g., some pseudoplastic fluids.

4

Ultrasound Rendering of Fluids through Amplitude Modulation

The advent of non-contact haptic displays has introduced new forms of interaction, allowing users to experience tactile sensations in mid-air without the need for holding or wearing a device. Ultrasound haptic devices are notable examples of these displays.

They employ arrays of ultrasonic transducers as actuators, which produce high-frequency pressure waves in the space around the device. By modulating the activation of the transducers, it is possible to aggregate the pressure waves at specific points in space, and thus create focal pressure points (Iwamoto et al., 2008). Pressure reaches perceivable values at such focal points, and produces a touch sensation in mid air.

By eliminating the need to hold or wear a haptic device, ultrasound haptics promises the possibility of a more immersive and scalable virtual touch experience. Ultrasound haptics also suffers evident limitations, however. The forces they exert on skin are subtle and cannot impose constraints on the user's motion.

Fluids appear as an interesting phenomenon to be rendered using ultrasound haptics, as they can be moved around without constraining the user's motion. When we interact with fluids, e.g., with our hands, we experience a temporally and spatially varying pressure field on our skin. This pressure field is the combined result of our own motion, the inherent properties of the fluid (i.e., density and viscosity), and the dynamic state of the fluid (i.e., velocity).

Rendering the tactile interaction with fluids using ultrasound haptics can be formulated as a problem of dynamically reproducing the pressure field on the user's skin. This problem comes with two major challenges. One is to perform a real-time fluid

simulation with moving objects (i.e., the user's hands) and dynamically extract the pressure field on the skin. The other one is to dynamically optimize the actuation of the transducers to approximate the pressure field on the skin.

The generation of tactile percepts using ultrasound devices is still a largely unknown process. Pressure waves elicit complex mechanical interaction on skin, both in space and time, and this mechanical interaction produces an at least equally complex activation and aggregation of mechanoreceptor signals to form tactile percepts. In the absence of a computational model that maps activation patterns of transducers to tactile percepts, previous works have explored different high-level metaphors to command ultrasound devices.

To date, two control metaphors prevail: amplitude modulation (AM) and spatiotemporal modulation (STM). AM controls the position and pressure intensity of focal points to produce pressure distributions on skin, while STM controls paths of focal points to *draw* shapes on skin. In both cases, the high-level control metaphor of focal points is translated into low-level control of transducer activation patterns through well-established optimization methods (Hoshi et al., 2010; Long et al., 2014).

In this chapter, we study the problem of rendering interactions with virtual environments using ultrasound haptics. We approach the problem as a dynamic mapping of virtual interactions to the control metaphors of ultrasound devices. Most previous works have simplified this problem by displaying contact locations at maximum intensity, either through AM or STM. On the other hand, we believe that richer display is possible if we account for the force distributions in virtual interactions, and not just contact locations.

We propose an algorithm for ultrasound rendering of tactile interaction with fluids based in the AM control metaphor. We characterize the actuation of ultrasound haptics using a set of focal points, and we optimize the location and intensity of such focal points to best approximate the pressure field on the skin. We devise efficient methods to extract the skin pressure field from the fluid simulation and optimize the focal points at high update rates, and hence produce a responsive experience while dynamically interacting with a virtual fluid.

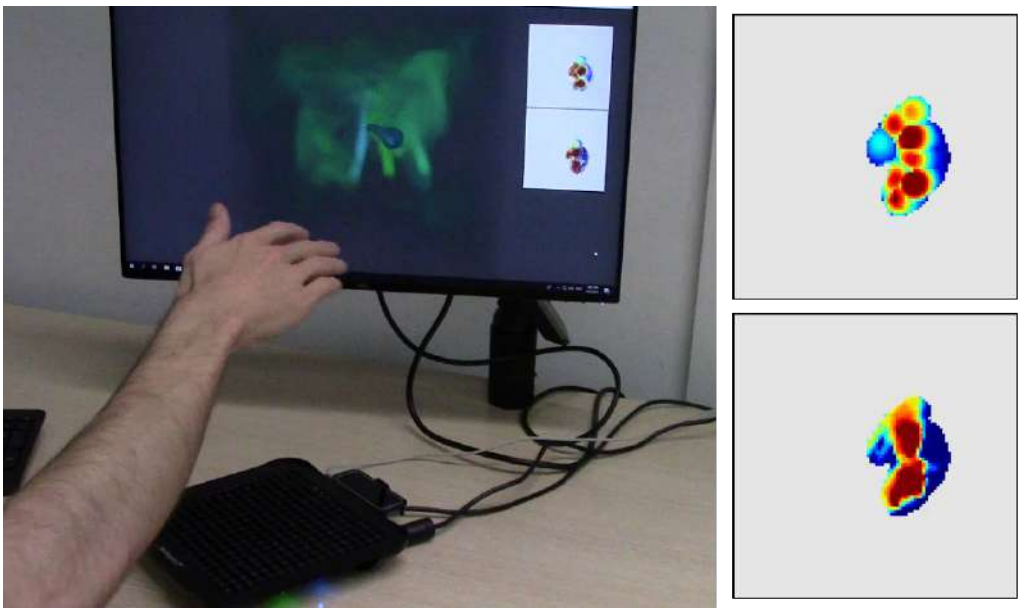


Fig. 4.1.: On the left image, a user interacts with a fluid simulation. His hand is tracked and mapped to a virtual hand that stirs the simulated fluid. We propose a novel tactile rendering algorithm that extracts the pressure field on the virtual hand **(bottom right)**, and optimizes a pressure field **(top right)** that is rendered to the user with the ultrasound phased array shown in the left image.

We have implemented our ultrasound tactile rendering method on an Ultrahaptics STRATOS device (Fig. 4.1). We demonstrate example interactions where the user interacts with a fluid container of $100 \times 100 \times 100$ cells, with the fluid simulation running at 90Hz and tactile rendering at 30Hz.

4.1 Rendering Based on Pressure Field Optimization

Without loss of generality, we assume that the user interacts with the simulated fluid using one hand. Then, the hand of the user is tracked in real-time, and a virtual replica of the hand is moved within the fluid simulation. The motion of the virtual hand, together with the properties and the motion of the fluid, produce a temporally and spatially varying pressure field on its surface. Ultrasound phased arrays are capable of producing a spatial pressure field. Therefore, we choose to

render the tactile interaction between the user and the simulated fluid by matching the pressure field produced by the ultrasound device to the pressure field on the user's virtual hand.

We start this section by characterizing the pressure field produced by the device, according to the control method of choice. Then, we define the target pressure field, which is constrained to the surface of the hand visible from the device. Finally, we formulate an optimization problem to compute the rendering output, and we provide an efficient solution algorithm.

4.1.1 Command and Stimuli of Ultrasound Haptic Devices

As discussed in Section 2.2, ultrasound phased arrays can be commanded following two high-level command metaphors: amplitude modulation (AM) and spatiotemporal modulation (STM). In our rendering problem, we wish to modulate a pressure field in space. One approach would be to extend emitter phase modulation to support arbitrary target regions with spatiotemporally varying pressure; another could be to modulate high-velocity spatiotemporal control points to maximally cover the target field. However, these approaches require the solution to a complex optimization of high dimensionality (i.e., the activation pattern of each transducer, or long-term point trajectories), running at high update rates.

Instead, we opt for an approach based on AM, which allows direct control of pressure values, albeit at a small number of points. We leverage the observation that, in reality, pressure is not concentrated at focal points, but exhibits a smooth fall-off determined by the wavelength of the ultrasound signal (e.g., 8.6mm for the 40kHz of our test device). This fall-off can be well approximated by a Gaussian function ϕ (Hoshi et al., 2010). Then, given a focal point at position $\mathbf{x}_i$ with nominal pressure p_i, and a pressure fall-off with standard deviation σ, the pressure at position $\mathbf{x}$ can be characterized as

$$p(\mathbf{x}) = p_i \, \phi(\|\mathbf{x} - \mathbf{x}_i\|) = p_i \, \mathrm{e}^{-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\,\sigma^2}}. \tag{4.1}$$

If focal points are distant enough, we can safely assume that the pressure at every location depends only on the closest control point. Then, given a set of focal points $\{\mathbf{x}_i\}$, each one of them defines the pressure field over its Voronoi region R_i according to (4.1).

4.1.2 Target Pressure Field

Given a fluid simulation defined on a volume domain $\mathbb{R}$, we are interested only in the pressure field on the surface of the user's hand. Moreover, parts of the surface of the hand may be occluded from the ultrasound device, hence it is pointless to try to match their simulated pressure. Consequently, we define the target pressure field $p^*(\mathbf{x})$ on a target domain $\mathbf{R} = \{\mathbf{x} \in \mathbb{R}^3\}$ formed by the portion of the surface of the virtual hand that is visible from the ultrasound device. Since the ultrasound phased array is capable of producing pressure intensities, we clamp to zero the negative target pressure values that occur when the hand moves away from the flow.

In practice, we sample the target domain $\mathbf{R}$ with a set of points. In Section 4.2.2 we describe an efficient GPU-based algorithm to extract the target domain $\mathbf{R}$ and the target pressure field $p^*(\mathbf{x})$ from a fluid simulation.

4.1.3 Pressure Field Optimization

Based on all the ingredients described thus far, we define the rendering problem as the search of N focal points and their pressure magnitudes, such that the difference between rendered and target pressures, $p(\mathbf{x}) - p^*(\mathbf{x})$, is minimized over the target domain $\mathbf{R}$. Following the assumption that the focal points are distant and hence the rendered pressure at each point is defined only by the closest focal point, we partition the target domain into the Voronoi regions of the focal points, i.e., $\mathbf{R} = \bigcup_i R_i$. Within each Voronoi region, the summed pressure difference depends only on the pressure magnitude of the corresponding focal point.

We propose an approximate solution to the pressure optimization problem that works in two steps. First, we compute the positions of focal points following a clustering approach. Second, we estimate the pressure magnitude of each focal point to best match the target pressure within its Voronoi region.

Optimization of Focal Points

We formulate a clustering problem using target pressure values as weights. Formally, this amounts to minimizing the following objective function:

$$f(\{\mathbf{x}_i\}) = \sum_i \sum_{\mathbf{x} \in R_i} p^*(\mathbf{x}) \, \|\mathbf{x} - \mathbf{x}_i\|^2. \tag{4.2}$$

This objective function corresponds to a weighted k-means clustering problem, which can be solved efficiently using Lloyd's algorithm (Lloyd, 2006). The algorithm iterates steps where it computes the weighted centroids of Voronoi regions and then updates those Voronoi regions, until convergence.

At every render frame, we initialize the iterative algorithm by placing the N focal points at distant high-pressure locations. In practice, we search for the N points in $\mathbf{R}$ with highest target pressure, such that they are separated by a distance larger than σ.

Optimization of Pressure Magnitudes

Once the focal points and hence the Voronoi partition are fixed, we optimize the pressure magnitude for each focal point independently. For each Voronoi region, we formulate an objective function based on the summed pressure difference between the target pressure $p^*(x)$ and the actual pressure rendered by the device, accounting for its fall-off as described in (4.1). This is a simple quadratic function of the form

$$f(p_i) = \sum_{\mathbf{x} \in R_i} \left(p_i \, \phi(\|\mathbf{x} - \mathbf{x}_i\|) - p^*(\mathbf{x})\right)^2. \tag{4.3}$$

By the optimality condition $\nabla f = 0$, we compute the pressure magnitude of each focal point as

$$p_i = \frac{\sum_{\mathbf{x} \in R_i} p^*(\mathbf{x})\, \phi(\|\mathbf{x} - \mathbf{x}_i\|)}{\sum_{\mathbf{x} \in R_i} \phi(\|\mathbf{x} - \mathbf{x}_i\|)^2}. \tag{4.4}$$

Once focal points and their pressure magnitudes are computed, we output them to the driver of the ultrasound device. The driver then executes internally the optimization of transducer waves (Long et al., 2014).

4.2 Fluid Simulation and Rendering Pipeline

In this section, we describe our fluid simulation solver, as well as the method for extracting the pressure field on the hand's surface.

4.2.1 Fluid Simulation

We apply our algorithm to the rendering of gaseous media such as smoke. As in the work of Fedkiw et al., 2001, we assume that the simulated fluid is inviscid and incompressible. Hence, its motion is described by the incompressible Euler equations:

$$\begin{aligned} \nabla \cdot \mathbf{u} &= 0, \\ \frac{\partial \mathbf{u}}{\partial t} &= -(\mathbf{u} \cdot \nabla)\, \mathbf{u} - \frac{1}{\rho} \nabla p + \mathbf{f}, \end{aligned} \tag{4.5}$$

where $\mathbf{u}$ corresponds to the velocity of the fluid, p is the pressure, ρ is the (constant) density, and $\mathbf{f}$ accounts for external body forces such as gravity. We solve these equations following the standard advection-projection scheme used in computer graphics (Crane et al., 2007; Fedkiw et al., 2001; Stam, 1999; Yang et al., 2009).

First, we compute an intermediate velocity field $\mathbf{u}^*$ by solving the self-advection equation in (4.5) using a semi-Lagrangian advection scheme (Stam, 1999) and integrating the external body forces $\mathbf{f}$ with explicit Euler. Next, to ensure mass

conservation, the velocity field is projected onto a divergence-free state by solving the pressure Poisson equation

$$\nabla^2 \cdot p = \frac{\rho}{\Delta t} \nabla \cdot \mathbf{u}^*, \tag{4.6}$$

with Neumann boundary conditions $\left(\frac{\partial p}{\partial \mathbf{n}} = 0\right)$ at boundaries with normal $\mathbf{n}$. Prior to the projection to the divergence-free state, we explicitly enforce free-slip boundary conditions ($\mathbf{u} \cdot \mathbf{n} = \mathbf{u}^* \cdot \mathbf{n}$), i.e., we set the normal velocities at fluid boundaries equal to those of the obstacles $\mathbf{u}^*$.

Following the state of the art, we discretize the simulation domain $\mathbb{R}$ using a staggered grid (see Section 2.1.1), with fluid pressure defined at cell centers, and fluid velocities defined component-wise at the centers of cell faces. We achieve interactive simulation times by implementing the entirety of the fluid solver on the GPU. As in the work of Crane et al., 2007, we enable massive parallelism by solving the pressure Poisson equation using Jacobi relaxation.

Due to semi-Lagrangian advection, numerical dissipation might dampen interesting features of the flow, such as vortices and eddies, which could be perceptually relevant. We use vorticity confinement (Fedkiw et al., 2001; Steinhoff & Underhill, 1994) to inject additional kinetic energy at existing vortices, and thus alleviate the effects of numerical dissipation.

For visualization purposes, we define immersed media, such as smoke, through secondary density fields. We advect these density fields at every frame using the same semi-Lagrangian advection method.

4.2.2 Adding the Hand and Extracting the Target Pressure Field

In the fluid simulation, the user's hand is treated as a moving obstacle with known velocity. On every simulation step, we rasterize a signed-distance representation of the hand, along with its corresponding velocity field. This representation enables an efficient classification of interior/exterior points of the domain, as well as the extraction of the surface normal, all under the same compact storage.

As outlined in Section 4.1.2, we wish to extract the fluid pressure on the portion of the boundary of the hand $\mathbf{R}$ visible from the ultrasound device. To do this, we assume that the ultrasound device is placed on one of the walls of the domain $\mathbb{R}$. Then, for every cell of this wall, we march inward into the domain until we hit the first cell that is within one unit of the hand, and we store this cell's position $\mathbf{x}$ and pressure $p^*(\mathbf{x})$.

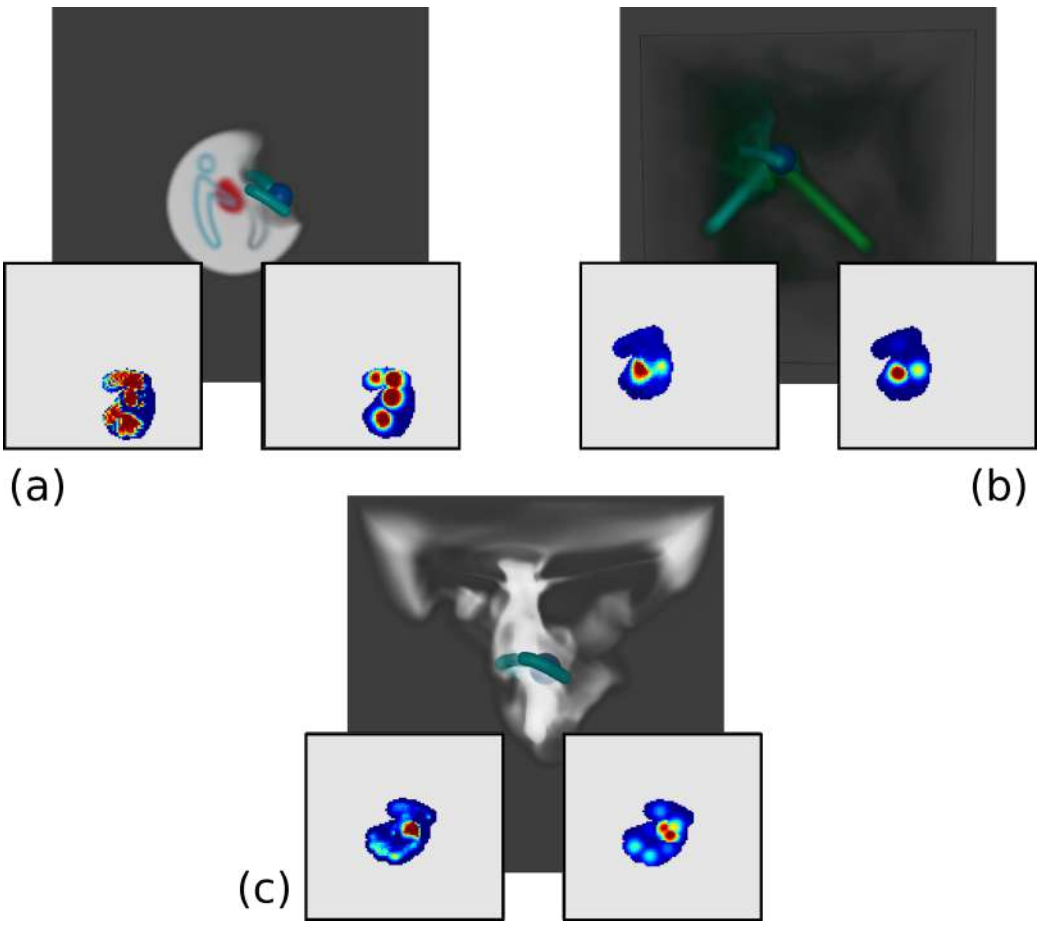


Fig. 4.2.: Examples of fluid interaction showing target pressure values extracted from the device's view of the hand from below (lower left inset) compared with reconstruction (lower right inset). Examples are **(a)** stirring the fluid, **(b)** hand in a smoke jet, and **(c)** creating smoke plumes.

4.3 Results

4.3.1 Summary of the Complete Rendering Pipeline

In our runtime pipeline, we set different refresh rates for the haptic rendering loop and for the fluid simulation loop. We set the haptic refresh rate at 30Hz for smooth rendering. At this rate, we sample the hand tracker, update the location of the hand in the fluid simulation, extract the target pressure field, optimize the focal pressure points and magnitudes, and output these commands to the haptic device driver.

We set the fluid simulation rate at the highest possible multiple of 30Hz; in our implementation, 90Hz. In this way, we maximize the hand speed that can be robustly handled by the simulation. With a domain $\mathbb{R}$ of $0.5 \times 0.5 \times 0.5$m discretized by $100 \times 100 \times 100$ cells, the CFL condition translates into a maximum hand velocity of 0.45m/s. In practice, due to the numerical dissipation of the simulation, we support even higher hand speed.

We run the fluid simulation and the haptic rendering on the same thread, with 3 fluid simulation steps per haptic update. We add a one-frame delay to the tracked hand positions, and interpolate them at in-between fluid simulation steps.

We have executed our rendering algorithm on an AMD Ryzen 7 2700 8-core 3.20 GHz PC with 32 GB of RAM and a Nvidia GeForce GTX 1080 Ti GPU with 11 GB of RAM. For ultrasound rendering, we have used an Ultrahaptics STRATOS Explore (USX) device, running at 40kHz, which also features a Leap Motion device used for hand pose tracking.

Fig. 4.2 shows screen-captures of example interactions. In all the captures, we compare the target pressure fields (lower left insets) and the pressure fields resulting from our optimization algorithm and reconstructed according to the model described in Section 4.1.1 (lower right insets). The results have been produced with $N = 8$ focal points and a pressure fall-off with $\sigma = 16$mm.

(a) With emitter

	$\sigma = 8$ mm	16 mm	24 mm	32 mm
$N = 2$	0.357	0.283	0.220	0.197
$N = 4$	0.334	0.242	0.193	0.186
$N = 8$	0.315	0.227	0.189	0.157

(b) Without emitter

	$\sigma = 8$ mm	16 mm	24 mm	32 mm
$N = 2$	0.335	0.268	0.206	0.180
$N = 4$	0.313	0.228	0.180	0.170
$N = 8$	0.296	0.210	0.173	0.168

Tab. 4.1.: The RMSE between target and reconstructed pressure for different numbers of points N and different radii σ of the pressure model (columns) for two simulation conditions. Lower values indicate higher reconstruction quality. In **(a)** the hand interacts with a smoke emitter; in **(b)** the hand stirs the smoke and there is no emitter.

4.3.2 Algorithm Evaluation

According to the specification of the STRATOS device, it supports amplitude modulation of up to 8 focal points and a frequency of 40kHz, which corresponds to a focal diameter of 8.6mm. Based on these values, we have evaluated the rendering quality of our algorithm with different numbers of focal points N, as well as different fall-off distances σ (which could be related to the focal radius). We used the manufacturer-recommended setting of 200 Hz for amplitude modulation, and typically settled on $N = 4$ focal points, since the power of individual points diminishes as more points are used. Using pre-recorded hand trajectories for the smoke stirring scene and the smoke jet scene, we have computed the root mean square error (RMSE) of the pressure field reconstruction under the different parameter settings.

Table 4.1 summarizes the RMSE results. A clear trend is visible towards a better reconstruction of the target values for an increasing number of points and larger radii. However, in practice, reconstruction radii must be based on the area of the hand affected by an individual focal point and thus selected based on the device capabilities.

Fig. 4.3 compares the reconstruction quality under the various parameter settings for one particular target pressure field. The images show that as focal point radius increases with respect to the Voronoi region size, the reconstructed values flatten and thus the edges between boundaries decrease in realism with respect to rendering. Thus, the focal point radii should be selected to give good results with respect to their distribution as well as separation.

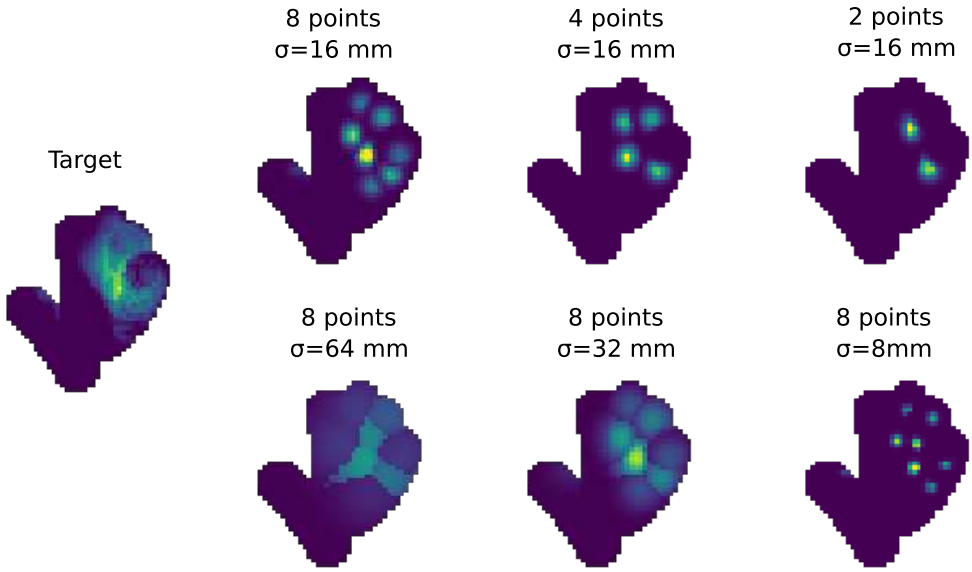


Fig. 4.3.: How pressure reconstruction changes with differing numbers of focal points **(top)** and focal radii **(bottom)**.

4.3.3 Timings

Timing information for the algorithm can be found in Table 4.2. Timing data was collected and averaged over one minute of simulation while looping the same trajectory used in the evaluation described in the previous subsection. Each simulation step takes roughly 6.21ms, and each optimization step 8.5ms. Recall that the simulation runs at 90Hz and the optimization and haptic rendering at 30Hz. The total computation time per haptic update is then 27.13ms.

(a) Timings per simulation step

Step	Time
Obstacles rasterization	0.44 ms
Advection	0.42 ms
Pressure	3.58 ms
Boundaries and vorticity	1.77 ms

(b) Timings per optimization step

Step	Time
Pressure extraction	1.50 ms
Transferring to host	2.65 ms
Optimization	4.35 ms

Tab. 4.2.: Timing data of the full pipeline, including both **(a)** the fluid simulation and **(b)** the optimization.

4.4 Discussion and Future Work

In this chapter we introduce the ability to interact with fluid simulations using ultrasound haptics. To achieve this, we have designed a novel method to display a pressure field on the surface of the hand. We propose to find an optimal set of focal points, minimizing the difference between the reconstructed pressure field and the target pressure obtained from the fluid simulation.

Choices of fall-off radius and number of focal points were explored with respect to the expected reconstruction accuracy. However, to fully evaluate the display method, a perceptual evaluation is required, not only in these variables, but also with regards to spatiotemporal aspects that may lead to experience these moving focal points as a representation of a dynamic field. We expect research in this direction to lead to formulations that bridge the amplitude modulation and spatiotemporal modulation control methods for ultrasound haptics.

5 Ultrasound Rendering of Fluids through Spatiotemporal Modulation

In Chapter 4 we address the rendering of interactions with virtual environments using ultrasound haptics, putting an special emphasis on fluid interaction. To this end, we extracted a target pressure field from the interaction of a virtual hand with a dynamic fluid simulation, and then found the optimal amplitude modulation (AM) commands that induce a best-matching pressure field on the user. However, AM suffers some limitations; probably the most evident that the intensity of focal points modulates a pressure wave that induces a perceivable vibration on skin (typically at 200 Hz).

Alternatively to AM, spatiotemporal modulation (STM) promises the ability to cover larger skin areas with higher perceived intensity, by leveraging constructive interference of focal point paths with the skin waves they induce. To date, all previous work commands STM with focal point paths of constant intensity; no previous method computes STM paths of varying intensity to best match dynamic interactions. Not surprisingly, STM poses a more complex challenge than AM. While AM rendering can be regarded as a quasi-static problem, STM requires the solution to a spatial and temporal problem.

In this chapter, we study the problem of rendering interactions with virtual environments targeting the STM command metaphor. For this, we propose *path routing optimization for STM* (PRO-STM), the first method that commands ultrasound STM to render the force distribution resulting from a dynamic virtual interaction. As we discuss in Section 5.1, a key aspect of our method is to pose STM rendering as a quasi-static problem. Thanks to careful approximations, we can eliminate the temporal variable on each dynamic rendering update. As a result, given a target

pressure field, we pose STM rendering as the computation of focal point paths that produce the best-matching quasi-static pressure field.

Then, given a target pressure field, we propose an optimization algorithm to compute focal point paths, as described in Section 5.2. Our algorithm works at two scales. First, on a coarse scale, it initializes paths over the target domain to optimize coverage weighted by pressure intensity. Then, on a finer scale, it refines the paths to maximize the similarity to the target pressure.

We have applied PRO-STM to the interaction with gaseous fluid media, as shown in Fig. 5.1. In such interactions, haptic perception is dictated by a spatially and temporally varying pressure field on skin, which is used as target for our algorithm. We have compared the reconstruction quality of PRO-STM vs. our previous AM rendering method, observing that PRO-STM succeeds to provide larger and smoother coverage than the AM-based method. In particular, AM produces ambiguous results when rendering interaction with one wide plume or with multiple thin plumes, while STM does not suffer such ambiguity. We have conducted an experiment that confirms this observation.

5.1 Principles of Spatiotemporal Modulation

To design an ultrasound rendering algorithm based on STM, it is important to understand the properties that best balance the capabilities of the ultrasound device with the quality and richness of tactile stimuli. We pay attention to the speed and frequency at which focal points traverse skin, and we extract desiderata for our algorithm. In addition, we devise a model of the radiation pressure produced by focal point paths, which helps design our algorithm. Specific values of the parameters and constraints of the algorithm depend on the choice of ultrasound device, an Ultrahaptics STRATOS device in our case.

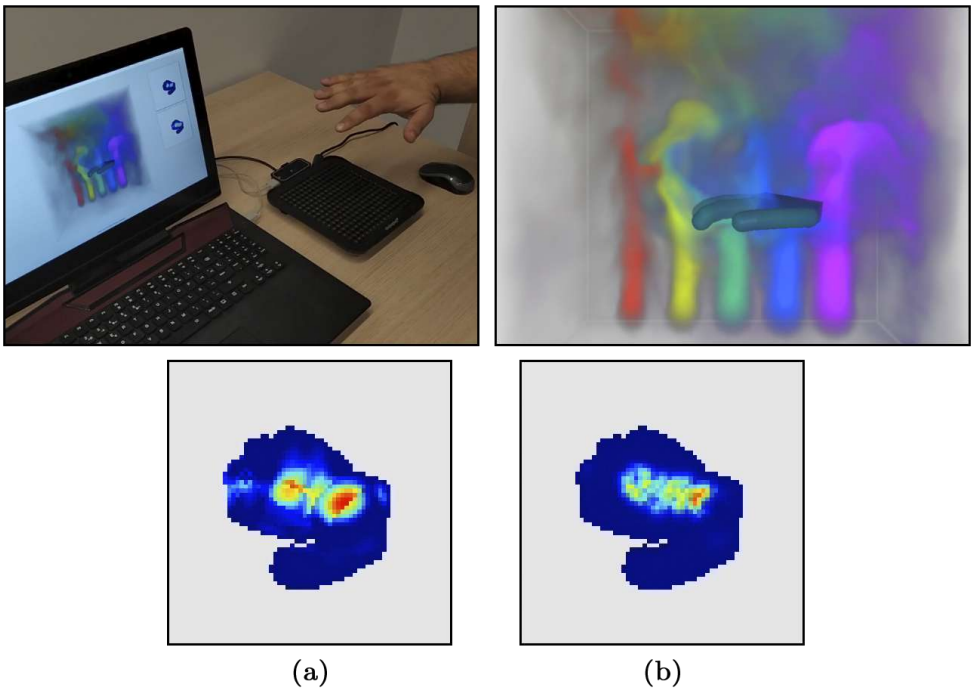


Fig. 5.1.: Example scene rendered using our novel PRO-STM method. From left to right: Physical setup with the ultrasound device and a view of an interactive fluid simulation; screen-capture of the fluid simulation, showing the virtual representation of the user's hand interacting with colored smoke plumes; pressure field **(a)** produced by the smoke on the surface of the hand, which sets the target for our algorithm; reconstructed pressure field **(b)** produced by STM rendering of dynamically optimized focal point paths.

5.1.1 Rendering Parameters and Constraints

In STM, a focal point traverses a path in space. We formally describe the path as a time-dependent position: $\mathcal{P} = \mathbf{x}(t) \in \mathbb{R}^3$. As noted in Section 2.2.3, Frier et al., 2018 concluded that the perceived intensity of a focal point path is maximized under constructive interference between the motion of the focal point and the propagation of skin waves. This happens for focal point speeds between 5 and 8 m/s; therefore, we select a reference speed $v = 7$ m/s for our rendering algorithm. Furthermore, to ensure constructive interference on the complete path, we design closed paths, i.e., $\mathcal{P}$ is a closed 3D curve.

Frier et al. assumed that the length of the path is given; therefore, the frequency at which the path is repeated cannot be independently controlled, and depends on

the traversal speed and the length of the path. In our preliminary experiments, we observed that this is acceptable up to a maximum path length. Beyond that length, the frequency at which the path is repeated is too low, and the stimulus is no longer perceived as continuous. To determine the minimum acceptable frequency, i.e., the maximum acceptable path length, we informally experimented rendering circles of different radii at the reference traversal speed of 7 m/s. We found that a minimum frequency of 50 Hz, i.e., a maximum path length $L = 140$ mm is a safe bound to ensure that the stimulus is perceived as continuous.

Our test device allows STM of multiple focal points simultaneously. Each focal point can traverse a different path, with all focal points traveling at the reference speed, and all paths satisfying the maximum length constraint. From our preliminary experiments, we have concluded to limit the number of simultaneous focal points, i.e., the number of simultaneous paths, to four. More focal points may reach larger coverage, but at the price of notable degradation of perceived intensity.

5.1.2 Quasi-Static Pressure Field

In previous works, STM was used to render 3D curves, hence the intensity of the radiation pressure of the moving focal point was kept constant along such curves. In this work, we render a temporally and spatially varying pressure field, hence the intensity of the radiation pressure along the path should adapt locally to the intensity of the pressure field. Based on this consideration, we characterize a path with a position-dependent pressure intensity $p(\mathbf{x})$.

As a focal point cycles multiple times through the same position $\mathbf{x}_i$, the rendered pressure intensity $p(\mathbf{x}_i) = p_i$ is the same on all cycles. If the path is repeated frequently enough (i.e., at more than 50 Hz), the rendered radiation pressure produces a persistent tactile perception. We consider that this is equivalent to applying a time-invariant pressure at each position along the path, with its effective magnitude a fraction of the rendered pressure. Thanks to this assumption, during a time window we can consider that the effective pressure is a spatially varying but temporally invariant field, i.e., a *quasi-static* pressure field.

Determining accurately the effective pressure of STM rendering is a complex subject that requires further research. It is not a simple time-average of the rendered pressure. In our work, we follow a perceptual heuristic to approximate its magnitude. We render the same stimuli using AM and STM, and we ask subjects to tune the gain γ of STM until the peak perceptual intensity is similar. In practice, we have used a gain $\gamma = 1.4$.

Focal points exhibit a smooth fall-off determined by the wavelength of the ultrasound signal (e.g., 8.6 mm for the 40 kHz of our test device). As shown by Hoshi et al., 2010, this fall-off can be approximated well by a Gaussian function ϕ. Based on this finding, together with the heuristic gain γ, we approximate the effective quasi-static pressure field produced by a focal point path as

$$p(\mathbf{x}) = \frac{1}{\gamma}\, p_i\, \phi(\|\mathbf{x} - \mathbf{x}_i\|) = \frac{1}{\gamma}\, p_i\, \mathrm{e}^{-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\,\sigma^2}}, \tag{5.1}$$

where $\mathbf{x}_i$ is the closest position to $\mathbf{x}$ in the path. We set the standard deviation σ of the Gaussian fall-off to the same value as the wavelength of the ultrasound signal (i.e., 8.6 mm)

The quasi-static pressure field assumption simplifies the design of our rendering algorithm. Given a target pressure field obtained from a fluid simulation, we pose each rendering step as the search for the focal point paths whose quasi-static pressure field best reconstructs the target field. This search must fulfill two constraints to ensure that the quasi-static pressure field assumption is valid, namely that each focal point travels at the reference speed v and the length of each path is not longer than the maximum length L. In the next section, we describe our path optimization algorithm.

5.2 Path Routing Optimization

Given a target pressure field, we seek focal point paths that produce a best-matching quasi-static pressure field. We solve this problem in two steps, at two different resolutions. Both steps search for paths that maximize coverage and integrated pressure intensity subject to the path length constraint, but the first coarse step

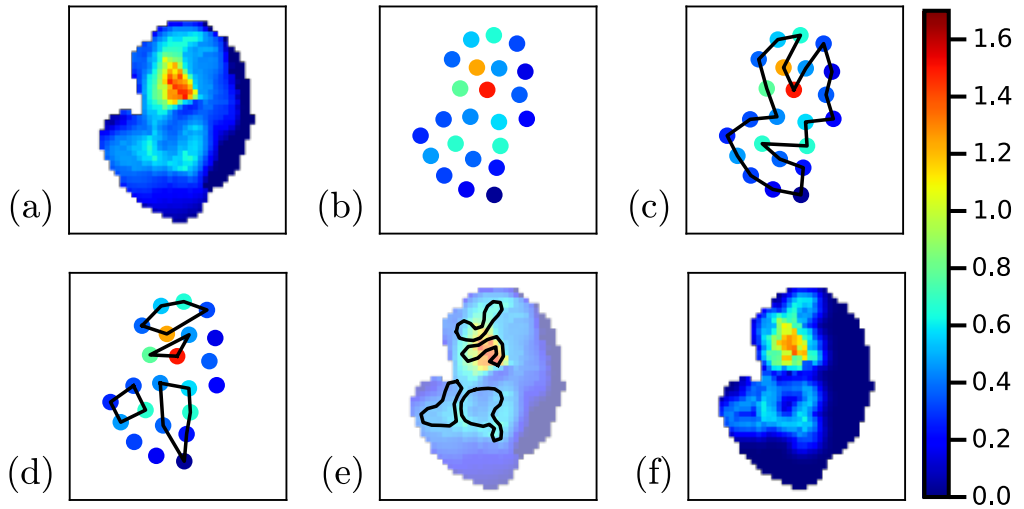


Fig. 5.2.: Steps of our PRO-STM algorithm: **(a)** input target pressure field, **(b)** clustering, **(c)** initial path, **(d)** split into multiple paths to satisfy length constraints, **(e)** refinement of the paths to maximize pressure intensity, **(f)** resulting reconstructed pressure field.

performs a global search, while the second fine step performs a local search. Before describing these two steps in detail, we describe how we obtain the target pressure field from a fluid simulation. And we conclude the section with implementation details to render the paths on our test device.

5.2.1 Target Pressure Field

We adopt the approach described in Section 4.2.1 to compute an interactive fluid simulation and extract the target pressure field. We track the user's hand and model it as a moving obstacle in a 3D simulation of a gaseous medium. We model fluid dynamics using incompressible Euler equations discretized on a 3D Eulerian grid, with semi-Lagrangian advection and massively parallel Jacobi relaxation for the pressure solve. The fluid simulation is executed on a GPU for maximum performance. Please refer to Section 4.2.1 for full details.

To define the target pressure field, we voxelize the hand, and select the voxel positions that are visible from the side of the domain that corresponds to the location of the ultrasound device. To simplify the path optimization problem, we

fit a plane to the voxel positions and we project them onto the plane, making path optimization a 2D problem. Formally, the target pressure field is described by a set of 2D positions and their corresponding target pressure values, $\mathcal{T} = \{\Big(\mathbf{x}_i \in \mathbb{R}^2, p^*(\mathbf{x}_i)\Big)\}$.

To ensure high computational performance, the initialization of the paths uses only a representative subset of the target pressure points $\mathcal{T}$. We apply weighted k-means clustering to $\mathcal{T}$, to produce a downsampled target pressure field with pressure points $\mathcal{D}$. In Fig. 5.2-a we show $\mathcal{T}$, the target pressure points on the user's hand extracted from a fluid simulation, and in Fig. 5.2-b we show $\mathcal{D}$, the downsampled pressure points.

5.2.2 Path Initialization

Given the pressure points $\mathcal{D}$, we seek a set of closed paths that visit all the points, subject to the maximum path length L. The optimal solution to this problem may require an arbitrarily large number of paths; however, as noted in Section 5.1.1, we concluded to limit the number of paths to four in practice. Consequently, the resulting paths may fail to visit all the pressure sample points, and an optimal set must be selected.

We solve this problem iteratively. We first compute the optimal path that visits all the points. If the path is too long, we split the set of points into two subsets and we compute separate optimal paths. We split the subsets of points recursively until all paths satisfy the maximum length constraint. Fig. 5.2-c shows the optimal path for the full set of pressure sample points, while Fig. 5.2-d shows the optimal paths after splitting the points to satisfy the maximum path length constraint. If the number of resulting paths is larger than four (as in the figure), we retain the four paths with highest integrated pressure, and we pass them to the refinement step described in the next section. But first, we describe in detail the operations to compute an optimal path for a set of points and to split a set of points.

Given a set of points, finding the shortest path that visits all the points corresponds to the *traveling salesman* problem. We solve this problem using the 2-opt algorithm (Croes, 1958), which admits closed paths. The computational cost of

2-opt sets an upper bound on the size of $\mathcal{D}$. In our implementation, we set it to a maximum of 50 points. Thus we run the weighted k-means clustering step above with 50 or fewer clusters.

To split a set of points, we find the direction of maximum spread, we bound the points along this direction, and we place a splitting plane orthogonal to the direction at the midpoint of the two bounds. To find the direction of maximum spread, we compute the covariance matrix of the points, weighted by their pressure value. The direction of maximum spread corresponds to the eigenvector with highest eigenvalue.

5.2.3 Path Refinement

After initialization, the paths pass through pressure clusters and fulfill the maximum length constraint. However, due to their coarse sampling, they are not optimally aligned with pressure peaks and ridges. We execute path refinement at a higher resolution, hence we start by upsampling each path to N points. $N = 20$ in our implementation, which sets points 7 mm apart from each other, i.e., the distance traveled by a focal point in 1 ms.

During refinement, the goal is to move path samples locally toward locations with higher pressure, while ensuring that paths preserve the following properties: **(i)** they satisfy the maximum length constraint; **(ii)** to reach maximum coverage, they do not (self-)intersect; and **(iii)** they do not bend at sharp angles, as the design decisions of our algorithm stem from perceptual observations on smooth paths, and paths with sharp corners reach smaller coverage. To implement refinement, we formulate the goal and the properties as cost terms of an objective function, and we execute a minimization algorithm.

Given paths with samples $\{\mathbf{x}_i \in \mathbb{R}^2\}$, we formulate a pressure intensity cost term as:

$$c_p = -\sum_i p^*(\mathbf{x}_i). \tag{5.2}$$

This term is minimized as the samples move to locations with higher pressure.

With a target path length L, and N samples per path, the target length is obtained if the length of each path segment is L/N. Then, we formulate a length cost term as:

$$c_l = \sum_i (\|\mathbf{x}_{i+1} - \mathbf{x}_i\| - L/N)^2, \tag{5.3}$$

where $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$ are two consecutive path samples.

If two paths or two portions of a path get closer than the fall-off distance of focal points, σ, they stimulate overlapping skin areas. The result can be considered inefficient, as the covered skin area is larger if the paths move away. We introduce a (self-)intersection cost term that prevents path samples from getting too close:

$$c_i = \sum_{i,j} \max(\sigma - \|\mathbf{x}_j - \mathbf{x}_i\|, 0)^2, \tag{5.4}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are two non-consecutive path samples.

Finally, to favor low-curvature paths, we introduce a bending cost term:

$$c_b = \sum_i \arctan^2 \frac{(\mathbf{x}_{i+1} - \mathbf{x}_i) \times (\mathbf{x}_i - \mathbf{x}_{i-1})}{(\mathbf{x}_{i+1} - \mathbf{x}_i)^T (\mathbf{x}_i - \mathbf{x}_{i-1})}, \tag{5.5}$$

where $\mathbf{x}_{i-1}$, $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$ are three consecutive path samples.

We optimize the paths by iterating steps of gradient descent of the four cost terms. For the pressure intensity term, we set a 2D grid with the target pressure values $\mathcal{T}$, and we use bicubic interpolation to evaluate the pressure at subgrid resolution and to compute robust gradients. Furthermore, we apply a line search to ensure that the step along the gradient reduces the cost. Fig. 5.2-e shows example paths after refinement.

To account for the effective magnitude of the pressure field, we must incorporate the heuristic gain γ discussed in Section 5.1.2. We set the rendered pressure of a point on a path as $p_i = \gamma\, p^*(\mathbf{x}_i)$, based on the target pressure field p^*. Fig. 5.2-f shows the reconstructed pressure field according to the quasi-static pressure model of Section 5.1.2.

5.2.4 STM Rendering

Once 2D paths are fully computed, we lift them back to 3D for rendering on the device. We achieve this by undoing the projection of the hand voxels discussed in Section 5.2.1.

Each path is 140 mm long and is traversed in 20 ms at 7 m/s. The STM rendering API of the Ultrahaptics STRATOS Explore (USX) device updates every 1 ms a burst of 40 consecutive focal point positions. Therefore, we linearly upsample each path to 800 points spaced 0.175 mm, and feed them in groups of 40 points to the API.

5.3 Experiments and Results

Here we describe some details of implementation and characteristics of its run-time performance. Following this, we describe a user study we performed to compare PRO-STM to the AM method.

5.3.1 Implementation Details

All experiments were performed, and timing data was collected, on a Lenovo laptop featuring an NVidia GeForce GTX 1070 GPU and Intel Core i7-6820HK CPU at 2.7 GHz, with 24 GB of RAM and 8 GB of video RAM. Software was written in C++ and CUDA for GPU kernels.

We have used an Ultrahaptics STRATOS Explore (USX) device for ultrasound rendering. The USX employs an array of 16×16 transducers running at 40 KHz, thus producing focal points of 8.6mm diameter. The device supports a maximum of 8 simultaneous focal points, but we have used 4 in our experiments. Hand pose tracking was achieved through the Leap Motion tracker device bundled with the USX.

The fluid simulation ran at 50 Hz on the GPU while PRO-STM was executed in parallel on the CPU at 25 Hz. At this rendering rate, each path was traversed twice per update.

5.3.2 Performance

We evaluated computational performance of PRO-STM on 500 frames recorded during interaction with two scenarios, namely 1 and 4 plumes of smoke, as these have qualitatively different distributions of pressure fields. Timings were broken down by segment of the algorithm, and are presented in Table 5.1. These data show that the algorithm runs at interactive rates (typically above 40 Hz in this implementation) without problems on commodity hardware, and may do so alongside a computationally heavy load driving a time-varying target field. In practice we run PRO-STM at 25 Hz in order to attain integer division of the path traversal frequency of 50 Hz, as discussed above.

We have also compared performance with our previous AM rendering method (Chapter 4). AM shows superior performance, with update rates of roughly 250 Hz on both test scenarios.

Step	1-plume Mean (μs)	StdDev	4-plumes Mean (μs)	StdDev
Init	285	106	142	59
K-means	1548	386	948	338
Initial path (2-Opt)	2550	646	1224	554
Splitting (recur. 2-Opt)	744	235	324	179
Resampling	12	4	10	4
Refinement	17896	1664	11120	3344
Total:	**24303**	**2256**	**14640**	**4281**

Tab. 5.1.: Timing for each step of PRO-STM on two scenarios.

5.3.3 Qualitative Comparisons

Currently we restrict the comparison between the current work and the previous AM method (Chapter 4) to a qualitative one because the respective reconstructed fields are not numerically comparable. This is because the AM field reconstruction is very sparse compared to the STM field reconstruction, but it is not clear from current knowledge of these haptic display methods how to properly consider the reconstructed fields from a perceptual point of view. In particular, the relation between the rendered pressure intensity and the perceived intensity is different for these two methods, and this variable strongly affects computed reconstruction error. The reconstruction model does not describe well the effects of parasitic tactile artifacts either, such as AM's constant vibration.

However, to demonstrate qualitatively the difference in coverage and detail reproduction, in Fig. 5.3 we visualize reconstructions of rendered pressure fields with STM and AM. For STM, we use the model of quasi-static pressure field of Eq. (5.1), which includes the heuristic gain between rendered and effective pressure. Both for STM and AM, we use $\sigma = 8.6$ mm, corresponding to the focal point fall-off. Since our model of quasi-static pressure reconstruction is based on heuristics, it is not suited for a quantitative (e.g., RMSE) comparison of STM and AM.

The renderings are taken from the scenarios used in the comparison study described later in this section. It is apparent that the field expected from STM does a better job at differentiating between these two conditions, while AM suffers from ambiguity, making it difficult to tell the difference between one large smooth shape and four smaller ones. In the following section, we describe a study intended to investigate this ambiguity and whether PRO-STM might overcome it.

5.3.4 Perceptual Study

To evaluate a difference in rendering between the proposed algorithm and the AM method for fluid rendering described in Chapter 4, we explored whether PRO-STM could provide more details to the user in a situation where the AM approach leads to ambiguity in the display.

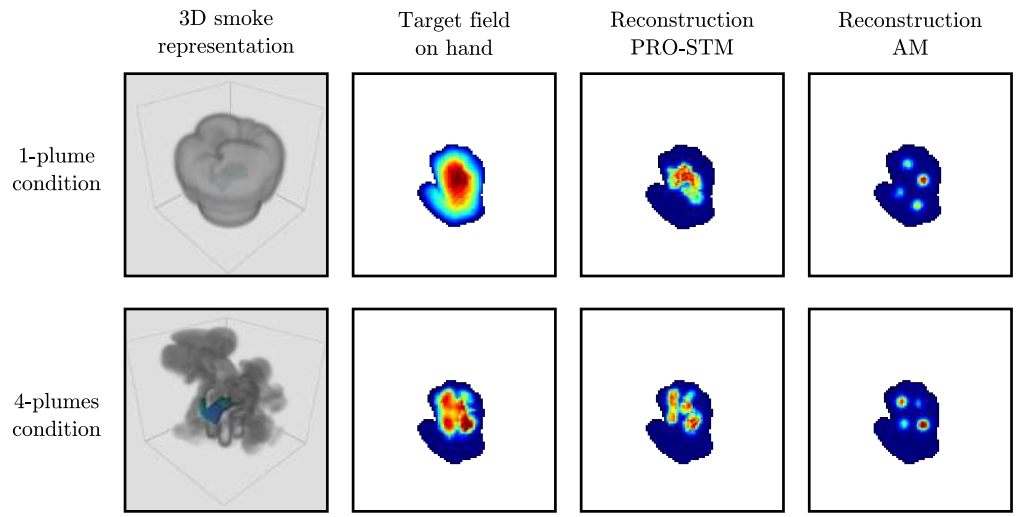


Fig. 5.3.: Experimental scenarios and representative examples of target and reconstructed fields using the proposed method (PRO-STM) and our previous AM method.

Specifically, we noticed that when a large, smooth pressure field is present, such as in the case of a single, large plume of smoke rising towards the hand, the AM approach places four control points in relatively stable positions. Correspondingly, it can be difficult to discriminate this display from a situation of four individual pressure concentrations, as in the case of four smaller columns of smoke. These situations are depicted in Fig. 5.3.

In the study, we asked participants to discriminate between 1-plume and 4-plume scenarios. The study featured 8 participants evaluated with AM, and 8 separate participants evaluated with PRO-STM. Initial testing revealed some confounding effects of mixing the two methods in the same experiment, as participants would use detectable differences between the two methods as erroneous cues to attempt to distinguish the two scenarios. Therefore, we found it most reliable to examine average performance across participant groups each evaluating different methods. No participants from earlier testing participated in the final study.

Participants were seated in front of the STRATOS device and a laptop running the fluid simulation. They were asked to hold their hand out flat above the device and feel the column or columns of smoke. First, an initial training period of two minutes was given where they were allowed to explore freely the two scenarios while seeing the simulation's 3D visual representation. Next, each participant performed 16

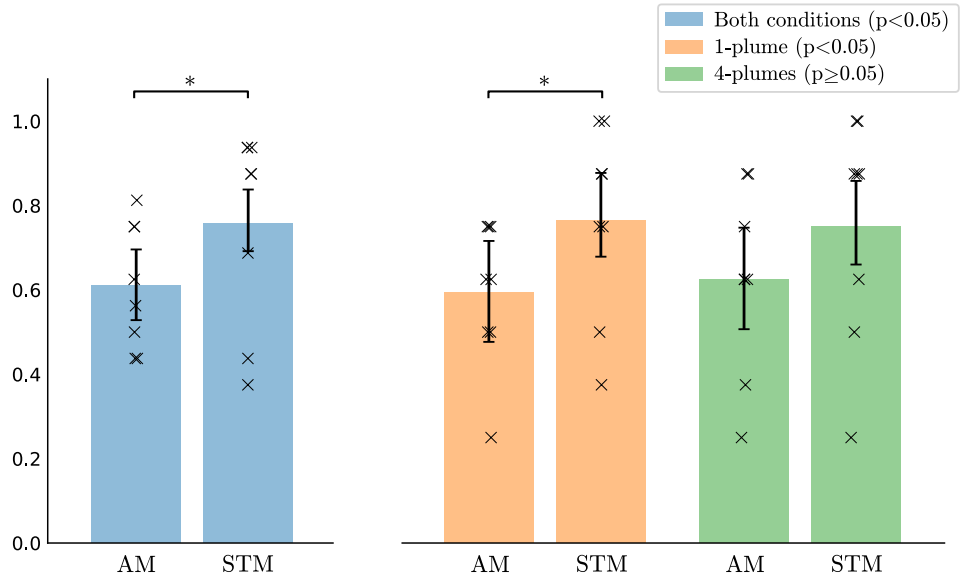


Fig. 5.4.: Proportions of correct responses for the two methods overall, and by scenario. Asterisks represent significant differences, determined by a pooled probability z-test across the cumulative samples. Error bars indicate standard error of participant means, with individual proportions marked by an 'x'.

trials, 8 with one column and 8 with four columns, in random order. On each trial, the participant experienced the scenario while looking at a static image along with a textual question, chosen randomly, "*Is there one plume of smoke?*" or "*Are there four plumes of smoke?*" The experimenter also repeated this text out loud for each trial, and was responsible for entering the information. The participant responded with "*true*" or "*false*" by voice. Participants did not have an explicit time limit to experience the scenarios and respond, but they took on average 5 seconds per trial.

Demographics: participants were between the ages of 20 and 50, with two participants above 40. Seven participants were female, and 9 were male. The majority of participants interacted with the experiment in their native language, Spanish. Two participants performed the experiment in English. One participant was left-handed and performed the experiment with his dominant hand. No participants reported health difficulties, abnormal skin conditions, or sensitivity issues.

A significant difference was found between AM and STM, with higher proportion of correct responses using STM, see Fig. 5.4, indicating that PRO-STM delivers

more detailed information about the target in both scenarios. No effect of the scenario was found, and since participants tried only one method each, no order of presentation effect is possible. Additionally no significant difference was found between genders or due to question correctness. The same experimenter performed all trials, which took place in an office environment.

Notable in the figure is that some participants could not perform the task, whereas others performed it exceptionally well. This would suggest a possible bimodal distribution, and we could exclude poorly performing individuals, but we elected not to as it did not affect the significance of the cumulative result. However, we noted that during informal testing and in demo conditions some people had difficulty detecting or discriminating using either AM or STM or both; this appears to be a limitation of ultrasound haptics for which we do not have enough data to correlate with explanatory variables such as skin type, hydration, or sensitivity, and remains an open question for the field.

5.4 Discussion and Future Work

We have described a new method, PRO-STM, for dynamically determining the paths of STM ultrasound focal points for an arbitrary target pressure field, and demonstrated improved coverage and smoothness over previous work based on AM ultrasound.

We account for current knowledge of STM to design algorithm features such as optimal traversal speed, maximum path length, or minimum inter-path distance. However, several perceptual factors related to STM remain unknown, and they can spur further algorithm improvements.

(i) We currently treat path routing as a 2D problem, whose solution is subsequently un-projected to a 3D path, and we ignore the change of path length due to this un-projection. The problem admits a more accurate formulation, accounting for the surface of the hand as a 2D manifold embedded in 3D; however, the effect of focal point velocity in 3D is an unknown aspect of STM perception. **(ii)** We introduce a bending cost term in the optimization, as we know that path corners are difficult to

distinguish; however, we do not know how to best weigh this cost, as the sensitivity of STM perception to changes in direction is unknown. **(iii)** We optimize paths for each rendering update independently, without special treatment to path transitions. This approach induces transients with unknown effects, as moving focal point paths have not been studied yet. **(iv)** We map the rendered pressure to the effective perceived pressure using a heuristic gain. We use a constant gain, but in reality the mapping depends on the spatiotemporal characteristics of each path and the specific effect produced on the subject's skin; effective skin deformation might be a more accurate metric to design a mapping. **(v)** The implementation of the algorithm relies on the choices of optimal traversal speed of 7 m/s and minimal frequency of 50 Hz. While these values are derived from perceptual studies, they can be refined through further experiments focused on the algorithm.

Although recent works are beginning to address perception of STM and its relation to wave propagation in the skin (Reardon et al., 2019), we lack knowledge of the effect on perceived intensity of fundamental STM parameters such as rendered intensity, traversal speed, or path frequency. This knowledge can improve our quasi-static pressure reconstruction model, and allow a more considered path optimization. Overall, as our method depends on hand tracking, it is in a good position to take into account new knowledge about hand skin mechanics and its connection to ultrasound-based stimulation.

6 Natural Interaction with Virtual Clay

One of the ambitions of virtual reality (VR) is to let people create 3D forms without the constraints of real-world objects, materials and procedures (Bouzbib et al., 2021). Today, VR provides commodity immersive display and hand tracking, two of the major requirements for effective VR-based 3D modeling applications. However, simulation models and interaction techniques have not reached the maturity necessary for virtually modeling complex materials such as clay in a natural way.

Virtual modeling of clay-like materials should include the following features: realistic behavior of the material, natural hands-on interaction, and tactile feedback. In this chapter, we present a simulation and interaction model that includes all these features, and therefore enables natural and tangible VR-based modeling of clay. We rely on an existing natural hand simulation model (Verschoor et al., 2018) to enable bidirectional coupling between the user and the virtual clay simulation. We use hand tracking to command the simulated virtual hand, we simulate the physical interaction between this hand and the clay-like material, and we feed the interaction forces back to the hand simulation. We also use these forces to command a tactile rendering algorithm. The two major technical contributions in the work presented in this chapter, which enable the overall system, are the simulation model of clay and the tactile rendering algorithm.

Clay is a complex material that combines properties of solids (permanent shape) with properties of fluids (plastic flow). To date, no interactive simulation method represents well this complex behavior. We present a particle-based model, as an extension to the one presented in Chapter 3, which captures well and efficiently the main features of clay-like materials. Our model, described in Section 6.1, includes novel formulations of constraints for viscosity, elastoplasticity, and friction, which together produce clay-like behavior.

Fig. 6.1.: Arts & crafts in the virtual classroom. We introduce a novel model of clay that allows interactive and highly realistic deformations, merging, and splitting. We also introduce an ultrasound rendering algorithm that enables a tangible interactive experience.

Clay is modeled with bare hands, and contact may occur both at finger pads as well as on large areas of the palm. Ultrasound rendering, despite its power limitations, offers the ability to stimulate the full hand in free air, with a good trade-off between coverage and resolution (Otaduy et al., 2016). In Section 6.2, we present a rendering algorithm that takes as input the contact forces between the virtual hand model and the clay material, and outputs focal pressure point commands for an ultrasound array. Our algorithm extends the one presented Chapter 4, and solves an optimization formulation that can accommodate perceptual weight maps.

We demonstrate the effectiveness of our simulation model and rendering algorithm on several examples of creative experiences with complex and rich clay material, such as the one shown in Fig. 6.1.

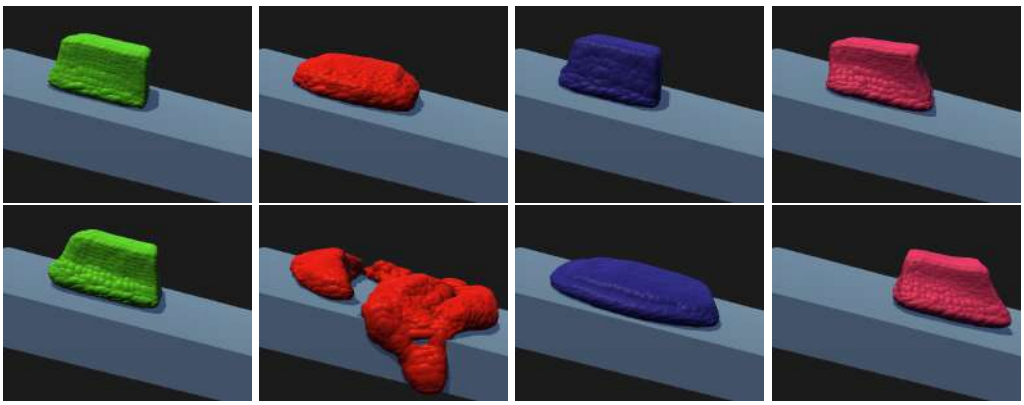


Fig. 6.2.: Ablation study of the novel constraints in our PBF clay model. We drop a block of clay on an incline, and we show its deformation during the impact (top) and one second later (bottom). From left to right: our full clay model (green); without viscosity constraints, the material flows fast and fractures (red); without elastoplasticity constraints, the material drifts (blue); and without friction constraints the block slides (magenta).

6.1 Viscoplastic Model of Clay

Particle-based Lagrangian representations offer the best compromise for the simulation of complex viscoplastic materials such as clay. They support robust strain metrics for elasticity through particle bonds, and they also support efficient plastic flow by dynamically (de)activating such bonds. Following the method presented in Chapter 3, we build on the PBF simulation method (Macklin & Müller, 2013). PBF formulates particle dynamics as a constrained minimization problem, and constraints are resolved using a relaxation algorithm. In PBF, solid and fluid constraints can be seamlessly handled, and the main difference is whether they use (semi-)permanent or temporary particle bonds. We start this section with a recap of the PBF algorithm for the simulation of incompressible fluids, which sets the baseline for our method.

To apply the PBF algorithm to highly viscoplastic materials such as clay, we propose novel constraint formulations that capture the major effects of the material. First, we model viscosity by expressing a constraint on strain rate. To this end, we must turn the velocity-based formulation into position-based constraints. Second, we model elastoplasticity using semi-permanent distance constraints. We achieve plasticity by integrating a hysteresis threshold on the (de)activation of the constraints.

And third, we model frictional contact using anchor constraints. All in all, these constraint formulations produce the characteristic clay behavior of extreme viscoplasticity. Fig. 6.2 compares the effect of each of the constraints on the behavior of clay.

6.1.1 PBF Simulation Model

As discussed in Section 2.1.2, PBF discretizes the material by a set of particles, and the state of each particle is defined by its position $\mathbf{x}_i$ and velocity $\mathbf{v}_i$. Particles also have mass m_i. PBF uses a smoothed particle hydrodynamics (SPH) Monaghan, 1992 approach to define variables in the continuum, and hence the value of a variable a at an arbitrary position $\mathbf{x}_i$ is evaluated as:

$$A(\mathbf{x}_i) = \sum_j \frac{m_j}{\rho_j} A_j W_{ij}, \tag{6.1}$$

with $W_{ij} = W(\mathbf{x}_{ij}) = W(\mathbf{x}_i - \mathbf{x}_j)$ the evaluation at $\mathbf{x}_i$ of a smoothing kernel centered at $\mathbf{x}_j$ with support radius h, A_j the value of the variable for the j^{th} particle, and ρ_j the density field evaluated at $\mathbf{x}_j$.

As noted in Section 2.1.2, in PBF the mechanical behavior of the material is defined and solved using constraints. In the following subsections we detail the formulation of the different types of constraints of our viscoplastic material model. First, let us define some general notation, where C_k represents each one of k types of constraints, and $\mathbf{J}_{k,i} = \frac{\partial C_k}{\partial \mathbf{x}_i}$ is its Jacobian with respect to the position of a particle.

Constraints are solved one at a time, projecting the particle positions such that linearized constraints are fulfilled. The particle projection for a constraint C_k is computed as Bender et al., 2014:

$$\Delta \mathbf{x}_i = -\frac{\frac{1}{m_i} \mathbf{J}_{k,i}^T C_k}{\sum_j \frac{1}{m_j} \|\mathbf{J}_{k,j}\|^2}. \tag{6.2}$$

The overall PBF algorithm proceeds as outlined in Algorithm 3. First, it computes unconstrained motion given by inertial and gravity forces. Then, it iterates over the

ALGORITHM 3: PBF step

Input: Previous particle states $\{\mathbf{x}_i^0\}, \{\mathbf{v}_i^0\}$.
Output: Updated particle states $\{\mathbf{x}_i\}, \{\mathbf{v}_i\}$.

```
/* Compute unconstrained positions                                    */
```
foreach *particle* i **do**
    $\mathbf{x}_i = \mathbf{x}_i^0 + \Delta t\, \mathbf{u}_i^0 + \frac{\Delta t^2}{m_i}\, \mathbf{f}_i(\mathbf{x}_i^0)$
end

```
/* Relaxation iterations                                              */
```
foreach *iteration* **do**
```
    /* Loop over constraint types                                     */
```
    foreach *constraint type* k **do**
```
        /* Solve constraints in parallel                              */
```
        foreach *constraint* j **do**
            project $\{\mathbf{x}_i\}$ such that $\mathbf{C}_{k,j}(\{\mathbf{x}_i\}) = \mathbf{0}$
        end
    end
end

```
/* Update velocities                                                  */
```
foreach *particle* i **do**
    $\mathbf{v}_i = \frac{\mathbf{x}_i - \mathbf{x}_i^0}{\Delta t}$
end

constraints, projecting the particles to the closest valid configuration. As in the work presented in Chapter 3, we maximize the efficiency of a GPU implementation by iterating over constraint types in Gauss-Seidel fashion, but executing all constraints of the same type in fully parallel Jacobi fashion. To conclude, the PBF algorithm computes particle velocities using finite differences.

We characterize clay as an incompressible extremely viscoplastic material. To model incompressibility, we use a constraint on the particle density, as done regularly in PBF methods (Bodin et al., 2012; Macklin & Müller, 2013). Next, we discuss the novel types of constraints used in our model.

6.1.2 Viscosity

Viscosity damps the differences in local velocities within a medium, and these velocity differences can be best characterized by the symmetric part of the strain

rate tensor (Bender & Koschier, 2017). Then, we design a viscosity constraint as:

$$\mathbf{C}_{viscosity}(\mathbf{v}_i) = \nabla \mathbf{v}_i + (\nabla \mathbf{v}_i)^T = \mathbf{0}. \tag{6.3}$$

Our constraint can be regarded as a simplified version of the viscoelasticity constraint presented in Chapter 3. However, we care only about viscosity, not viscoelasticity, and we model the elastic behavior of the material through an explicit elastoplastic constraint discussed later.

The velocity gradient in (6.3) is obtained by differentiating the SPH kernel (6.1) to obtain:

$$\nabla \mathbf{v}_i = \sum_j \frac{m_j}{\rho_j} \left(\mathbf{v}_j - \mathbf{v}_i\right) \nabla W_{ij}^T. \tag{6.4}$$

To avoid damping rotational motion, we define particle velocities using a corotational finite-difference approximation as $\mathbf{v}_i = \frac{\mathbf{x}_i - \mathbf{x}_i^r}{\Delta t}$, where $\mathbf{x}_i^r$ represents the rotational part of the displacement.

In Fig. 6.2 we compare the simulation with (green) and without (red) the viscosity constraint. With viscosity, the material becomes rigid soon after an impact, and without viscosity it flows fast and quickly fractures.

6.1.3 Elastoplasticity

Viscosity alone cannot represent the behavior of materials such as clay. The material exhibits a resistance to deviate from its stable configuration (i.e., elasticity), although it soon flows into a different configuration. We model this behavior using an elastoplastic model at low velocities, combined with the previous viscous model at larger velocities.

In PBF, elasticity can be handled using distance constraints between pairs of particles $\mathbf{x}_i$ and $\mathbf{x}_j$. The constraint is formulated as:

$$\mathbf{C}_{elasticity}(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\| - d_{ij} = \mathbf{0}, \tag{6.5}$$

where d_{ij} is the rest length of the constraint.

To model plastic flow, we (de)activate elasticity constraints based on the relative velocity of particle pairs projected to their connecting segment. If the projected relative velocity of a particle pair falls below a threshold, we activate an elasticity constraint. On the other hand, if the relative velocity grows over a threshold, we deactivate the elasticity constraint to allow the particles to flow. We achieve stable behavior through the use of distinct (de)activation thresholds (i.e., hysteresis).

In Fig. 6.2 we compare the simulation with (green) and without (purple) the elastoplasticity constraint. Without elastoplaticity the material drifts slowly, and with elastoplasticity it soon stops deforming.

6.1.4 Contact and Friction

We model frictional contact with arbitrary objects, such as the dynamically deforming hand, by rasterizing ghost particles on these objects, and then setting particle-based constraints. We use two types of constraints, to model impenetrability and friction, as we detail next.

For each solid object, we rasterize ghost particles in its rest configuration, and transform these particles according to the motion of the object. For the hand, we mesh its interior with tetrahedra, and transform the particles using the barycentric coordinates of the enclosing tetrahedra.

We execute collision detection at the beginning of each time step. For each ghost particle that falls within a radius R of a clay particle, we set an impenetrability constraint with the same formulation of the distance constraint (6.5), with $d_{ij} = R$. As done by Macklin et al., 2014, we ensure that impenetrability only repels and does not attract particles, by designing one-sided contact normals.

To handle Coulomb friction, we use a sliding-anchor model, inspired by spring-based frictional contact models (Yamane & Nakamura, 2006), but adapted to the constraint-based PBF simulation algorithm. Every time a particle $\mathbf{x}_i$ suffers a new contact, we set an anchor $\mathbf{a}_i$ at an offset R from the location of the corresponding ghost particle. Then, we add a zero-distance constraint between $\mathbf{x}_i$ and $\mathbf{a}_i$. However, when applying the particle projection (6.2) due to the friction constraint, we

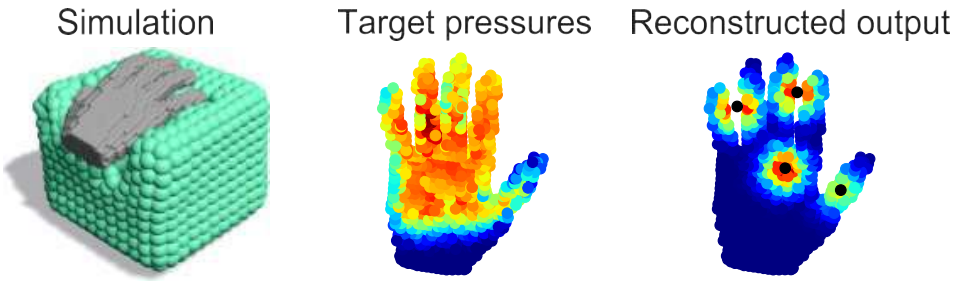


Fig. 6.3.: Our tactile rendering algorithm proceeds according to these steps, from left to right: (i) The particle-based simulation computes forces and deformation on clay particles due to the interaction with ghost particles on the hand (Section 6.1). (ii) We compute a pressure field on the ghost particles on the hand (Section 6.2.1). (iii) We compute the location and pressure of focal points such that the reconstructed pressure field is optimal, and these focal points are commanded to an ultrasound array for amplitude-modulation rendering (Section 6.2.2).

limit it based on the friction coefficient μ and the particle projection due to the impenetrability constraint, i.e., $\|\Delta \mathbf{x}_{friction,i}\| \leq \mu \|\Delta \mathbf{x}_{impenetrability,i}\|$. Furthermore, at the end of each step we slide each anchor, such that the distance to the colliding particle is given by the total particle projection of the friction constraint.

In Fig. 6.2 we compare the simulation with (green) and without (magenta) the friction constraint. Without friction the material slides down the incline.

6.2 Ultrasound Haptic Rendering

Based on the interactive simulation of clay presented in the previous section, in this section we describe how we provide feedback to the user, in the form of ultrasound-based tactile rendering. We divide this task in two steps, outlined in Fig. 6.3. First, we obtain a target pressure field from the contact forces between the hand model and the clay material. Second, we design an amplitude-modulation rendering algorithm to compute focal pressure points that are commanded to the ultrasound device. Next, we describe these two steps in detail.

6.2.1 Interaction Pressure Field

Similar to previous works on ultrasound rendering of fluids, we extract a pressure field from the interaction between the simulated hand and the clay material. However, since our PBF simulation model differs from the Eulerian model employed in Chapters 4 and 5 or the SPH model of Jang and Park, 2020, we require a different method to compute the pressure field on the simulated hand.

As described in Section 6.1.4, we set ghost particles on the hand to handle contact with the clay material. We use these same ghost particles to recover the pressure field on the hand. On every projection step due to an impenetrability contact constraint, we compute the contact force on the corresponding ghost particle. The force $\Delta\mathbf{f}_i$ of a projection step (6.2) can be obtained as:

$$\Delta\mathbf{f}_i = -\frac{\frac{1}{\Delta t^2}\,\mathbf{J}_{k,i}^T\, C_k}{\sum_j \frac{1}{m_j}\,\|\mathbf{J}_{k,j}\|^2}, \tag{6.6}$$

with $k = impenetrability$. By adding up the forces from all constraint iterations in a time step, we obtain the total collision force $\mathbf{f}_i$ on each ghost particle. Then, same as Jang and Park, 2020, we obtain the target pressure p^* by computing the component of the force normal to the surface at every ghost particle $\mathbf{x}_i$:

$$p^*(\mathbf{x}_i) = \max\left(-\mathbf{n}_i^T\,\mathbf{f}_i, 0\right), \tag{6.7}$$

where $\mathbf{n}_i$ is the outward surface normal. Note that we only consider ghost particles that are facing the location of the ultrasound array.

The middle image in Fig. 6.3 shows a sample interaction, with ghost particles in the hand color-coded according to their pressure.

6.2.2 Amplitude-Modulation Rendering

Once the target pressure field is defined, we optimize a set of focal pressure points and their corresponding pressure, which are commanded to the ultrasound device for amplitude-modulation rendering. We opt for amplitude modulation instead of spatiotemporal modulation to accommodate a user-defined perceptual weighting

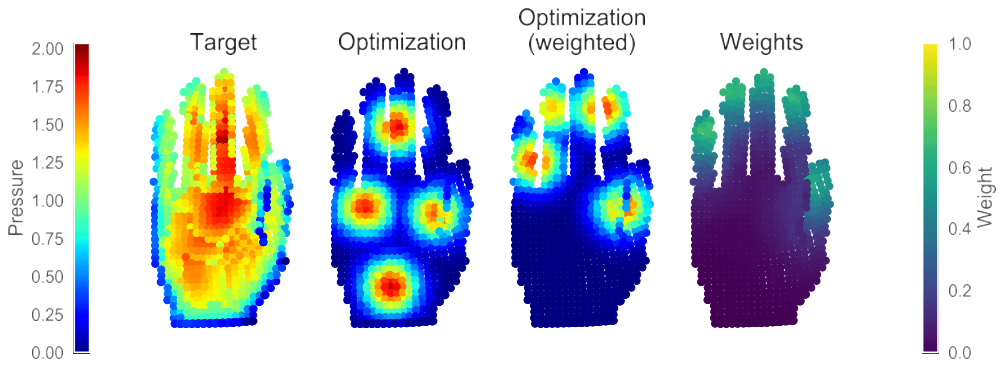


Fig. 6.4.: Our rendering algorithm supports perceptual weight maps to favor higher accuracy on more sensitive areas of the hand, such as the finger pads. On the left, we show particles color-coded according to their target pressure; on the right, we show the weight map. The two optimizations indicate the reconstructed pressure with and without weight map.

function $\alpha(\mathbf{x})$, as we see next. This function allows us to focus attention on more sensitive areas of the hand, as shown in Fig. 6.4.

We follow the clustering algorithm presented in Chapter 4, augmented with the perceptual weighting function. Specifically, we set weights $\alpha = 1$ at finger pads, and smoothly decay the weight function based on the distance to the closest finger pad, following roughly mechanoreceptor densities (Corniani & Saal, 2020).

Our algorithm first finds the locations of focal points through a k-means clustering problem. We search for the $\{\mathbf{x}_k\}$ focal point locations, using the weighted target pressure $\alpha(\mathbf{x})\, p^*(\mathbf{x})$:

$$\{\mathbf{x}_k\} = \arg\min \sum_k \sum_{\mathbf{x}_i \in Vor(\mathbf{x}_k)} \alpha(\mathbf{x}_i)\, p^*(\mathbf{x}_i)\, \|\mathbf{x}_i - \mathbf{x}_k\|^2, \tag{6.8}$$

where $Vor(\mathbf{x}_k)$ is the Voronoi region of focal point $\mathbf{x}_k$. Then, the algorithm computes the rendered pressures $\{p_k\}$ of the focal points, by minimizing the difference between the target pressure and the reconstructed pressure. This reconstructed pressure assumes a Gaussian fall-off with standard deviation σ given by the wavelength of the ultrasound signal.

$$p_k = \arg\min \sum_{\mathbf{x}_i \in Vor(\mathbf{x}_k)} \left(p_k \, \mathrm{e}^{-\frac{\|\mathbf{x}_i - \mathbf{x}_k\|^2}{2\,\sigma^2}} - p^*(\mathbf{x}_i) \right)^2 \Rightarrow$$

$$p_k = \frac{\sum_{\mathbf{x}_i \in Vor(\mathbf{x}_k)} p^*(\mathbf{x}_i) \, \mathrm{e}^{-\frac{\|\mathbf{x}_i - \mathbf{x}_k\|^2}{2\,\sigma^2}}}{\sum_{\mathbf{x}_i \in Vor(\mathbf{x}_k)} \mathrm{e}^{-\frac{\|\mathbf{x}_i - \mathbf{x}_k\|^2}{\sigma^2}}}, \forall k. \tag{6.9}$$

The computed focal point positions and pressures are commanded to the ultrasound device on each rendering frame, resulting on tangible feedback of the user's interaction.

6.3 Experiments

We have tested our simulation and rendering methods on two interactive modeling scenarios. Fig. 6.5 shows a kindergarten table with colorful blocks of modeling clay. The user deforms, splits and merges the blocks as shown in the images and the video. Fig. 6.6, on the other hand, shows a potter's wheel with a block of clay. In this case, the user interacts with both hands to create a solid of revolution as the wheel rotates. Both scenes demonstrate the robustness of the clay-like material, which exhibits the extreme viscoplasticity of real-world clay. We render clay graphically using a screen-space ellipsoid splatting method. The depth buffer is smoothed to produce a continuous surface, but some bumpiness and interpenetrations may appear.

The ultrasound rendering algorithm provides tangible feedback of the interaction. As shown in Fig. 6.7, this feedback depends on the properties of the clay material. With a more plastic material, contact forces are smaller under the same user actions, which turn into lower rendered pressures.

In the kindergarten and pottery scenes, the number of particles is respectively $2\,114$ and $4\,169$. The full simulation and tactile rendering runs at approximately 25 fps. The cost is dominated by the simulation of the clay material (27 ms per frame in the kindergarten scene, 30 in the pottery scene), while the rendering algorithm takes only 2 ms per frame. The simulation of the hand takes 6 ms per frame, but it

is executed in parallel on a different thread. Within the clay simulation, the cost is dominated by the elastoplasticity constraint. We use a time step of $1/60$, which is smaller than the actual update rate. This, together with the numerical damping introduced by the PBF solver, makes the simulated physics appear slightly slower than real-world physics. All examples were executed on an AMD Ryzen 7 2700 8-core 3.20 GHz PC with 32 GB of RAM and a Nvidia GeForce GTX 1080 Ti GPU with 11 GB of RAM. For ultrasound rendering, we have used a STRATOS Explore (USX) device from Ultraleap, running at 40kHz.

6.4 Discussion and Future Work

In this chapter, we have demonstrated a solution for tangible interactive clay modeling, based on a clay simulation method and an ultrasound-based rendering algorithm. This work enables novel rich creative experiences, thanks to the robust handling of extreme viscoplasticity, and to the optimization of the rendering output

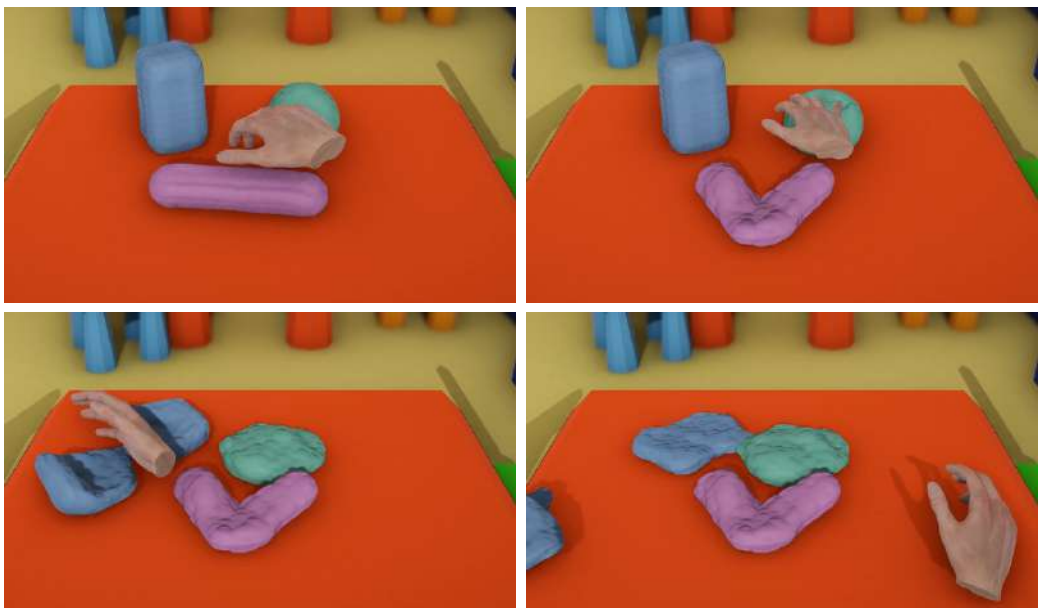


Fig. 6.5.: Interactive modeling of colorful clay figures. Our clay model supports robust viscoplastic deformations, which are key for generating arbitrary stable shapes, and for merging and splitting material.

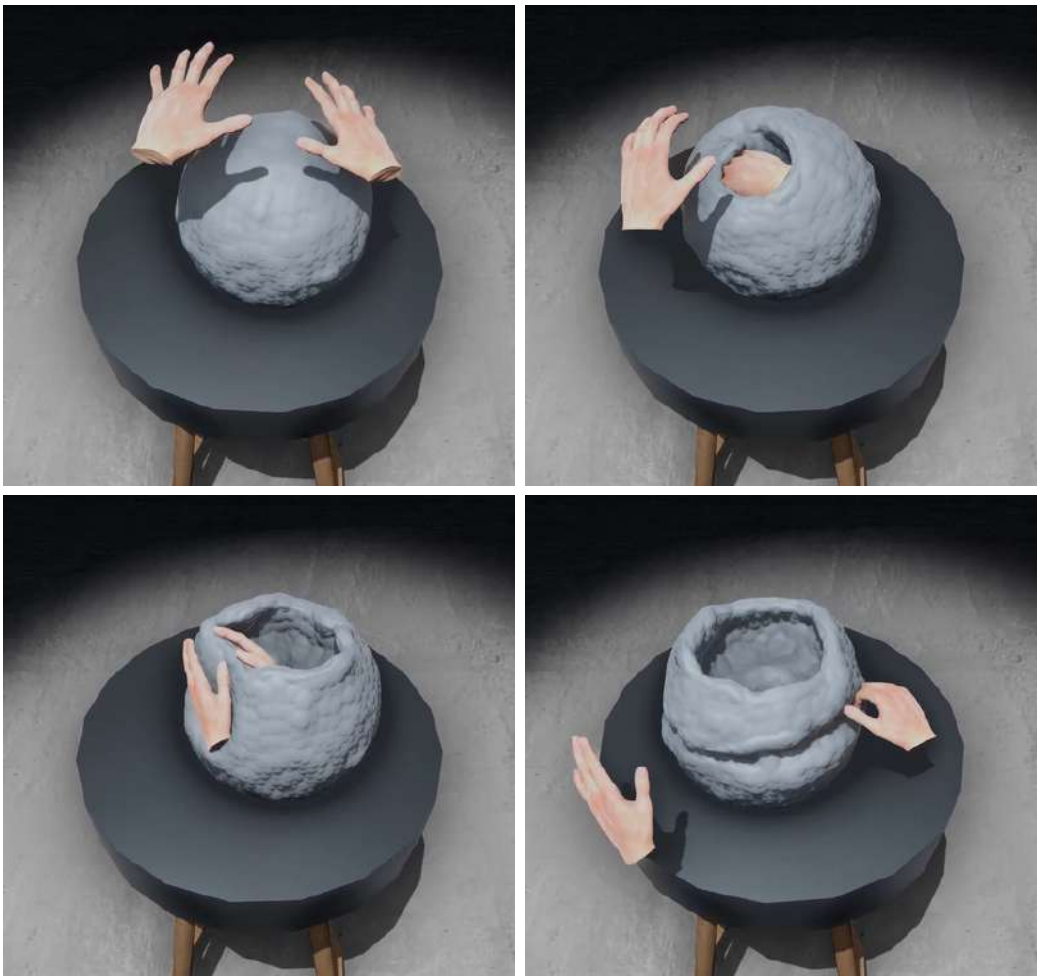


Fig. 6.6.: Tangible modeling of clay on a potter's wheel. We simulate two-handed interaction with clay, providing a natural hands-on experience.

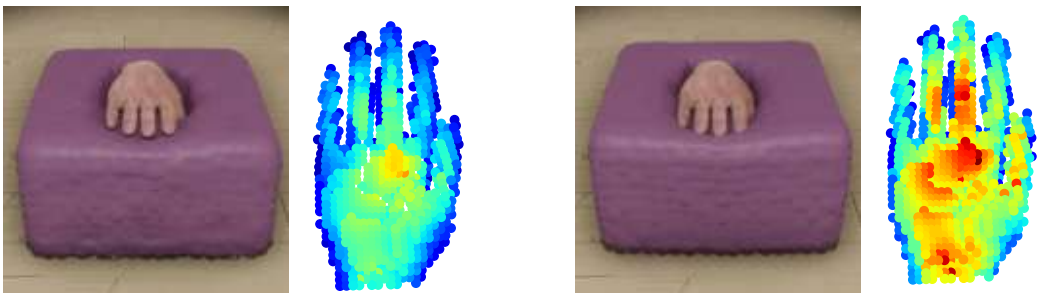


Fig. 6.7.: Rendered pressures vary depending on the properties of the material. With less plasticity **(right)**, the material flows less and imposes a higher pressure field on the hand.

for arbitrary interactions. Given the working solution, now it is possible to turn the attention to the investigation of choices and improvements.

We identify three limitations, which could motivate future work. One is the dominant cost of the elastoplasticity constraint, which may produce 10 to 20 constraints per particle. This cost limits the number of particles in the scene, and therefore the resolution of the clay material.

Another limitation is the coverage of the amplitude-modulation rendering method. As evidenced in Fig. 6.4, the quick decay of the focal points prevents matching large pressure areas on the hand. Nevertheless, the weight map allows us to focus on perceptually relevant locations. A possible alternative is to investigate spatiotemporal rendering methods to reach larger coverage (Kappus & Long, 2018; Korres & Eid, 2016).

Last but not least, the dexterity of the interaction is limited when trying to execute detailed modeling with the fingers. The reasons are multiple, including the low resolution of the particles, but probably also the limited fidelity of the tactile stimulation. We chose ultrasound-based stimulation due to its free-air operation and ease of use, but it is worth investigating what type of tactile feedback is best suited to enable dexterous modeling with the fingers.

7 Conclusions

In the previous chapters, we have presented the technical contributions of this thesis. This chapter concludes the discussion of the presented works by providing a general summary of the proposed ideas, their limitations, potential applications and possible avenues of future work.

7.1 General conclusions

The overall goal of this thesis was to enable physical interaction with rich and complex virtual phenomena such as fluids. We achieved this in two ways. First, we developed a novel fluid simulation model that is extremely efficient. This enables the representation of a wide range of materials in virtual environments. Second, we developed two rendering methods for the tactile interaction with such media using modern haptic technology, namely ultrasonic phased arrays.

Chapter 3 focused on the design of the aforementioned model for the efficient simulation of viscoelastic fluids. Our formulation is based in a constitutive model for polymeric fluids, which describes the elastic and viscous properties of the fluid as a function of the time evolution of a conformation tensor. As result, our method permits the representation of a wide range of materials (ranging from inviscid to highly viscous or viscoelastic) under one single formulation. Moreover, our constraint-based approach enables the use of high-performance constrained dynamics solvers, thus allowing its implementation into interactive scenes. The development of this work culminated in the publication of the article *Conformation Constraints for Efficient Viscoelastic Fluid Simulation* (Barreiro et al., 2017) in the **ACM Transactions on Graphics** journal (JCR Q1).

Chapters 4 and 5 are devoted to the development of novel tactile interaction algorithms for ultrasonic phased arrays. In fluid interaction, the activation patterns

of these displays must be controlled in order to produce a tactile sensation resembling the interactions with the virtual fluid. We achieved this goal by casting device actuation as a numerical optimization problem, targeting two well established high-level control metaphors commonly employed in ultrasound haptics: amplitude modulation (AM) and spatiotemporal modulation (STM). Given a target pressure field, we optimize the intensity and locations/trajectories over time of the ultrasonic foci to find the commands that induce a best-matching pressure field on the user skin. Thanks to our optimization-based approach, we are also able to incorporate knowledge of the technical and perceptual limitations of both control metaphors to maximize the efficacy of our solutions. To our knowledge, no prior work had addressed the problem of reproducing arbitrary target pressure distributions following a similar scheme.

This research culminated with a conference presentation and a journal publication. The AM-based rendering algorithm was presented at the **IEEE World Haptics Conference 2019** under the title *Ultrasound Rendering of Tactile Interaction with Fluids* (Barreiro et al., 2019), whereas the STM-based algorithm was published in the **IEEE Transactions on Haptics** journal (JCR Q2) under the title *Path Routing Optimization for STM Ultrasound Rendering* (Barreiro et al., 2020). The novelty of the latter work was recognized with the best paper award at the **IEEE Haptics Symposium 2020** conference.

Finally, in Chapter 6 we have proved the applicability of the approaches presented in this thesis by proposing a computational solution for the interactive simulation of clay-like materials. For this end, we augmented the model presented in Chapter 3 to incorporate the characteristic elastoplastic behavior that clay-like materials exhibit. This model, coupled with a real-time hand simulation and a haptic rendering algorithm based in the one presented in Chapter 4, enables natural hands-on manipulation of virtual clay-like materials with unprecedented degree of realism. This work culminated with a presentation at the **IEEE World Haptics Conference 2021** titled *Natural Tactile Interaction with Virtual Clay* (Barreiro et al., 2021).

7.2 Discussion and Future Work

Naturally, the approaches presented in this thesis are not devoid of limitations. However, these reveal the scope of applicability of our methods and unveil new challenges and possible avenues for future research that might further serve towards the realization of the virtual reality dream. In this section, we provide a recap of the limitations of the different methods presented in this thesis and highlight possible lines of future work.

7.2.1 Simulation of Viscoelastic Fluids

Despite the rich effects achieved with the method presented in Chapter 3 and the range of materials that can be simulated, there are still some limitations that suggest interesting future work.

Our work inherits some of the generic limitations of the PBD and PBF approach, particularly the convergence limitations of Jacobi or Gauss-Seidel solvers. These convergence limitations are related to the speed at which information propagates across the simulation domain. Given that relaxation methods are key for an efficient GPU implementation, it would be interesting to explore approaches offering improved convergence properties.

Recent works (Sommer et al., 2020; H. Wang, 2015) found success in the application of more sophisticated methods such as Chebyshev iterations to fluid simulation. Scheduled Relaxation Jacobi (Islam & Wang, 2020) methods are also interesting candidates, having gained popularity in recent years due to their simplicity and efficiency. However, these methods are still in their infancy, and additional research is necessary to ascertain their suitability to such applications.

Alternatively, it would be interesting to take advantage of the connection between PBD and minimization formulations of implicit integration to explore higher efficiency optimization algorithms, as done by others after the projective dynamics method (Bouaziz et al., 2014). However, those optimization algorithms cannot be

trivially extended to fluids as they make strong connectivity assumptions (Weiler et al., 2016).

Our method cannot handle large elastic deformations accurately, which would require storing some explicit measure of rest state. Additionally, while some of our examples demonstrate the simulation of fine features, the ability to resolve such fine features is eventually limited by the particle resolution of the simulation. Fusing codimensional representations (B. Zhu et al., 2015) with constraint-based viscoelasticity would enable even richer effects under manageable computational cost.

Furthermore, the stability of our simulation is highly dependent on the choice of smoothing kernel, as well as how the deficiencies on the free surface are handled. In practice, we maintain the stability of the simulation by making use of a regularization term that depends on the number of neighboring particles. However, its parameterization is material-dependent. Exploring the properties of different smoothing kernels as well as alternative strategies permitting robust treatment of such deficiencies (e.g. ghost particles or schemes based in neighbourhood prototypes in the spirit of Solenthaler and Pajarola, 2009 and Alduán et al., 2017) could also constitute an interesting line of future work.

Our DC-PBD solver might be applicable to other types of constraints beyond those handled in our work. One such example is friction, which shares a dissipative nature with viscosity, but incorporates constraints on the deviatoric stress. Another example is incompressibility. Similar to the divergence-free SPH method by Bender and Koschier, 2017, incompressibility constraints could be applied on both positions and velocities within our DC-PBD solver.

Even though our method is parameterized using two physics-based parameters, it is difficult to design them purely from measurable physical quantities in a discretization-independent manner. Our model is derived from a constitutive model for polymeric fluids, and the parameters could be set from geometric and physical quantities only for such fluids. However, we apply the model to other types of viscoelastic fluids too, and in that case the model can be regarded as empirical or phenomenological, and parameters could be estimated from measurements. In this context, video-guided parameter estimation (T. Takahashi & Lin, 2019) could prove a useful tool for finding such parameterization. In our examples, we have

opted for an artist-driven parameter design, which nevertheless proves effective thanks to the narrow set of parameters.

Finally, another problem of the parameterization is that many details of the constitutive model are reduced to just one parameter, the constraint compliance. This limits the ability to represent non-Newtonian fluids with complex dependence on any of these material parameters, e.g., some pseudoplastic fluids.

7.2.2 Ultrasound Rendering of Fluids through Amplitude Modulation and Spatiotemporal Modulation

While the algorithms presented in Chapters 4 and 5 address the rendering problem from distinct command metaphor perspectives, both share some limitations that are consequence of the choices made in modeling the device actuation, as well as perceptual unknowns of the actuation principle itself.

Both methods assume that the pressure distribution exerted by a focal point is isotropic and can be approximated by a Gaussian function for the reconstruction of the resulting pressure field. However, focal points actually exhibit anisotropic distributions, and are dependent on both their position in space, as well as the transducer alignment within the ultrasonic phased array. Furthermore, our actuation model does not account for the occurrence of other effects caused by the device's technical limitations, such as maximum power distribution across the working space (and how it is reduced in presence of multiple focal points), the appearance of grating lobes, or the occurrence of spurious pressure maxima at locations other than the focal points.

Although researchers are beginning to identify the variables associated with the perception of AM and STM stimuli, as well as their relationship to wave propagation in the skin (Reardon et al., 2019), there is still a lack of knowledge regarding the perceptual effects of fundamental parameters in terms of perceived intensity and discrimination performance. This is specially important when it comes to STM, as spatiotemporal characteristics of the stimulus such as traversal speed, phase and frequency heavily influence perception. Incorporating this knowledge could

substantially increase the quality of our quasi-static pressure reconstruction model, allowing for a more careful optimization while improving resource allocation.

Overall, as our methods depend on hand tracking, they are in a good position to take into account new knowledge about hand skin mechanics and its connection to ultrasound-based stimulation.

Amplitude Modulation Algorithm

Concerning the limitations of each method, our AM-based algorithm relies on the premise that pressures at a particular point in space are determined solely by the closest focal point. However, a more detailed understanding of the device actuation would require discarding this assumption in order to consider the aggregate effects of the focal points. Determining the extent to which this knowledge can be incorporated into an optimization method and developing a solver scheme capable of fulfilling the performance constraints imposed by haptic applications remains as future work.

Spatiotemporal Modulation Algorithm

As for our STM-based algorithm, we currently treat path routing as a 2D problem, whose solution is subsequently un-projected to a 3D path, and we ignore the change of path length due to this un-projection. The problem admits a more accurate formulation, accounting for the surface of the hand as a 2D manifold embedded in 3D space; however, the effect of focal point velocity in 3D is an unknown aspect of STM perception.

The morphology of the paths routed by our algorithm is significantly dependent on the weights associated with each optimization term. We introduce a bending cost term in the optimization, as we know that path corners are difficult to distinguish; however, we do not know how to best weigh this cost, as the sensitivity of STM perception to changes in direction is unknown.

We optimize paths for each rendering update independently, without special treatment to path transitions. This approach induces transients with unknown effects, as moving focal point paths have not been studied yet. Furthermore, we disregard the impact of relative traversal phases across the different paths. Although their effect has not been studied yet, we suspect that they have a noticeable effect on perceived intensity as a result of constructive interference.

We map the rendered pressure to the effective perceived pressure using a heuristic gain. We use a constant gain, but in reality the mapping depends on the spatiotemporal characteristics of each path and the specific effect produced on the subject's skin; effective skin deformation might be a more accurate metric to design a mapping.

Finally, the implementation of the algorithm relies on the choices of optimal traversal speed of 7 m/s and minimal frequency of 50 Hz. While these values are derived from perceptual studies, they can be refined through further experiments focused on the algorithm.

7.2.3 Natural Interaction with Virtual Clay

To conclude, our solution for tangible interactive clay modeling is susceptible to improvements. We identify three limitations, which could motivate future work.

One is the dominant cost of the elastoplasticity constraint, which may produce 10 to 20 constraints per particle. This cost limits the number of particles in the scene, and therefore the resolution of the clay material.

Another limitation is the coverage of the amplitude-modulation rendering method. The quick decay of the focal points prevents matching large pressure areas on the hand. Nevertheless, the weight map allows us to focus on perceptually relevant locations. Incorporating our solution based in spatiotemporal modulation would help us reach larger coverage.

Last but not least, the dexterity of the interaction is limited when trying to execute detailed modeling with the fingers. The reasons are multiple, including the low

resolution of the particles, but probably also the limited fidelity of the tactile stimulation. We chose ultrasound-based stimulation due to its free-air operation and ease of use, but it is worth investigating what type of tactile feedback is best suited to enable dexterous modeling with the fingers.

7.3 Final Remarks

In this thesis, we have presented several highly innovative models and algorithms that substantially contribute towards the advancement of the state of the art in fluid simulation and haptic interaction. Their suitability has been demonstrated through multiple practical cases, enabling user interaction with a variety of interesting fluid materials.

Given the rapid evolution of VR technologies, its popularization and the ever increasing demand for realistic and sophisticated interaction models, we are confident that the proposed solutions will be well received both by industry and practitioners, and therefore will serve as a starting point for further exploration in the field.

Our contributions, however, are not limited to the field of VR; but reach other industries as well. Such is the case of *Next Limit Technologies*, a pioneer in the development of fluid simulation solutions for VFX. As a result of our direct collaboration with them for the development of the method presented in Chapter 3, our solution has been integrated into the *Dyverso* particle solver (Alduán et al., 2017) included in the *RealFlow* package, receiving extraordinarily positive feedback from artists and technical directors for its efficiency and ability to model diverse materials.

Finally, we are excited about the potential for engagement that future technologies will provide. The standardization of mechanical models capable of accurately capturing the behavior of biological soft tissues such as the hand (Verschoor et al., 2018) will enable a significant improvement in the quality of virtual interactions and their coupling to other physical phenomena. When this occurs, researchers will be in a privileged position to develop perceptual models that provide greater and more reliable information for driving the choices of future tactile rendering methods.

This, together with continued advancements in the VR ecosystem's processing capabilities and the introduction of new and more powerful machine learning algorithms capable of handling the ever growing stream of sensory information, will shape the future of VR and enable previously inconceivable degrees of detail and realism.

Bibliography

Aanjaneya, M., Gao, M., Liu, H., Batty, C., & Sifakis, E. (2017). Power diagrams and sparse paged grids for high resolution adaptive liquids. *ACM Transactions on Graphics (TOG)*, *36*(4), 1–12.

Alduán, I., & Otaduy, M. A. (2011). Sph granular flow with friction and cohesion. *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 25–32.

Alduán, I., Tena, A., & Otaduy, M. A. (2017). Dyverso: A versatile multi-phase position-based fluids solution for vfx. *Computer Graphics Forum*, *36*(8), 32–44.

Andersen, H. C. (1983). Rattle: A "velocity" version of the shake algorithm for molecular dynamics calculations. *Journal of Computational Physics*, *52*(1), 24–34.

Andrade, L. F. d. S., Sandim, M., Petronetto, F., Pagliosa, P., & Paiva, A. (2015). Particle-based fluids for viscous jet buckling. *Comput. Graph.*, *52*(100), 106–115.

Bailey, D., Biddle, H., Avramoussis, N., & Warner, M. (2015). Distributing liquids using openvdb. *Acm siggraph 2015 talks* (pp. 1–1).

Baraff, D. (1989). Analytical methods for dynamic simulation of non-penetrating rigid bodies. *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*, 223–232.

Baraff, D. (1996). Linear-time dynamics using lagrange multipliers. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, 137–146.

Barreiro, H., García-Fernández, I., Alduán, I., & Otaduy, M. A. (2017). Conformation constraints for efficient viscoelastic fluid simulation. *ACM Transactions on Graphics (TOG)*, *36*(6), 1–11.

Barreiro, H., Sinclair, S., & Otaduy, M. A. (2020). Path routing optimization for stm ultrasound rendering. *IEEE transactions on haptics*, *13*(1), 45–51.

Barreiro, H., Sinclair, S., & Otaduy, M. A. (2019). Ultrasound rendering of tactile interaction with fluids. *2019 IEEE World Haptics Conference (WHC)*, 521–526.

Barreiro, H., Torres, J., & Otaduy, M. A. (2021). Natural tactile interaction with virtual clay.

Batty, C., & Bridson, R. (2008). Accurate viscous free surfaces for buckling, coiling, and rotating liquids. *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 219–228.

Batty, C., Uribe, A., Audoly, B., & Grinspun, E. (2012). Discrete viscous sheets. *ACM Trans. Graph.*, *31*(4), 113:1–113:7.

Baxter, W., & Lin, M. C. (2004). Haptic interaction with fluid media. *Proceedings of graphics interface 2004*, 81–88.

Becker, M., Ihmsen, M., & Teschner, M. (2009). Corotated sph for deformable solids. *Proceedings of the Fifth Eurographics Conference on Natural Phenomena*, 27–34.

Becker, M., & Teschner, M. (2007). Weakly compressible sph for free surface flows. *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 209–217.

Bender, J., & Koschier, D. (2017). Divergence-free sph for incompressible and viscous fluids. *IEEE Transactions on Visualization and Computer Graphics*, *23*(3), 1193–1206.

Bender, J., Müller, M., Otaduy, M. A., Teschner, M., & Macklin, M. (2014). A survey on position-based simulation methods in computer graphics. *Computer Graphics Forum*, *33*(6), 228–251.

Bergou, M., Audoly, B., Vouga, E., Wardetzky, M., & Grinspun, E. (2010). Discrete viscous threads. *ACM Trans. Graph.*, *29*(4).

Berndt, I., Torchelsen, R., & Maciel, A. (2017). Efficient surgical cutting with position-based dynamics. *IEEE Computer Graphics and Applications*, *37*(3), 24–31.

Bhatia, H., Norgard, G., Pascucci, V., & Bremer, P.-T. (2012). The helmholtz-hodge decomposition—a survey. *IEEE Transactions on visualization and computer graphics*, *19*(8), 1386–1404.

Bird, R. B., Armstrong, R. C., Hassager, O., & Curtiss, C. F. (1977). *Dynamics of polymeric liquids* (Vol. 1). Wiley New York.

Bodin, K., Lacoursiere, C., & Servin, M. (2012). Constraint fluids. *IEEE Transactions on Visualization and Computer Graphics*, *18*(3), 516–526.

Bouaziz, S., Martin, S., Liu, T., Kavan, L., & Pauly, M. (2014). Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph.*, *33*(4), 154:1–154:11.

Bouzbib, E., Bailly, G., Haliyo, S., & Frey, P. (2021). "can i touch this?": Survey of virtual reality interactions via haptic solutions. *Proceedings of IHM Conference (To appear)*.

Bridson, R. (2015). *Fluid simulation for computer graphics*. CRC press.

Carlson, M., Mucha, P. J., III, R. B. V. H., & Turk, G. (2002). Melting and flowing. *Proceedings of SIGGRAPH'02*.

Carter, T., Seah, S. A., Long, B., Drinkwater, B., & Subramanian, S. (2013). Ultrahaptics: Multi-point mid-air haptic feedback for touch surfaces. *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, 505–514.

Chang, Y., Bao, K., Liu, Y., Zhu, J., & Wu, E. (2009). A particle-based method for viscoelastic fluids animation. *Proc. 16th ACM sympos. Virtual Reality Software and Technology*.

Chaudhury, S., & Chaudhuri, S. (2014). Volume preserving haptic pottery. *2014 IEEE Haptics Symposium (HAPTICS)*, 129–134.

Chentanez, N., & Müller, M. (2011). Real-time eulerian water simulation using a restricted tall cell grid. *Acm siggraph 2011 papers* (pp. 1–10).

Chorin, A. J. (1968). Numerical solution of the navier-stokes equations. *Mathematics of computation*, *22*(104), 745–762.

Cirio, G., Marchal, M., Lécuyer, A., & Cooperstock, J. R. (2013). Vibrotactile rendering of splashing fluids. *IEEE transactions on haptics*, *6*(1), 117–122.

Cirio, G., Marchal, M., Otaduy, M. A., & Lécuyer, A. (2013). Six-dof haptic interaction with fluids, solids, and their transitions. *World Haptics Conference (WHC), 2013*, 157–162.

Clavet, S., Beaudoin, P., & Poulin, P. (2005). Particle-based viscoelastic fluid simulation. *Proc. of 2005 ACM SIGGRAPH/ Eurographics Symposium on Computer Animation*.

Colin, F., Egli, R., & Lin, F. Y. (2006). Computing a null divergence velocity field using smoothed particle hydrodynamics. *Journal of Computational Physics*, *217*(2), 680–692.

Corniani, G., & Saal, H. P. (2020). Tactile innervation densities across the whole body. *Journal of Neurophysiology*, *124*(4).

Crane, K., Llamas, I., & Tariq, S. (2007). Real-time simulation and rendering of 3d fluids. *GPU gems*, *3*(1).

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, *6*(6), 791–812.

Daviet, G., & Bertails-Descoubes, F. (2016). A semi-implicit material point method for the continuum simulation of granular materials. *ACM Trans. Graph.*, *35*(4), 102:1–102:13.

Desbrun, M., & Gascuel, M.-P. (1996). Smoothed particles: A new paradigm for animating highly deformable bodies. *Computer animation and simulation'96* (pp. 61–76). Springer.

Deshpande, A. P., Krishnan, J. M., & Kumar, P. B. S. (Eds.). (2010). *Rheology of complex fluids*. Springer New York.

Dewaele, G., & Cani, M.-P. (2003). Interactive global and local deformations for virtual clay. *Proc of Pacific Graphics Conf*.

Dobashi, Y., Sato, M., Hasegawa, S., Yamamoto, T., Kato, M., & Nishita, T. (2006). A fluid resistance map method for real-time haptic interaction with fluids. *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, 91–99.

Fang, Y., Li, M., Gao, M., & Jiang, C. (2019). Silly rubber: An implicit material point method for simulating non-equilibrated viscoelastic and elastoplastic solids. *ACM Transactions on Graphics (TOG)*, *38*(4), 1–13.

Fedkiw, R., Stam, J., & Jensen, H. W. (2001). Visual simulation of smoke. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 15–22.

Foster, N., & Fedkiw, R. (2001). Practical animation of liquids. *Proc. of SIGGRAPH*, 23–30.

Foster, N., & Metaxas, D. (1996). Realistic animation of liquids. *Graph. Models Image Process.*, *58*(5), 471–483.

Frier, W. T. A. (2020). *Rendering spatiotemporal mid-air tactile patterns* (Doctoral dissertation). University of Sussex.

Frier, W. T. A., Ablart, D., Chilles, J., Long, B., Giordano, M., Obrist, M., & Subramanian, S. (2018). Using spatiotemporal modulation to draw tactile patterns in mid-air. In D. Prattichizzo, H. Shinoda, H. Z. Tan, E. Ruffaldi, & A. Frisoli (Eds.), *Haptics: Science, technology, and applications* (pp. 270–281). Springer International Publishing.

Frier, W. T. A., Pittera, D., Ablart, D., Obrist, M., & Subramanian, S. (2019). Sampling strategy for ultrasonic mid-air haptics. *CHI Conference on Human Factors in Computing Systems Proceedings*.

Fu, C., Guo, Q., Gast, T., Jiang, C., & Teran, J. (2017). A polynomial particle-in-cell method. *ACM Transactions on Graphics (TOG)*, *36*(6), 1–12.

Gerszewski, D., Bhattacharya, H., & Bargteil, A. W. (2009). A point-based method for animating elastoplastic solids. *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.

Goktekin, T. G., Bargteil, A. W., & O'Brien, J. F. (2004). A method for animating viscoelastic fluids. *ACM Trans. Graph.*, *23*(3), 463–468.

Goldenthal, R., Harmon, D., Fattal, R., Bercovier, M., & Grinspun, E. (2007). Efficient simulation of inextensible cloth. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, *26*(3).

Hairer, E., Lubich, C., & Wanner, G. (2002). *Geometric numerical integration* (Vol. 31). Springer-Verlag.

Han, G., Hwang, J., Choi, S., & Kim, G. J. (2007). Ar pottery: Experiencing pottery making in the augmented space. *Virtual Reality*.

Harlow, F. H., & Welch, J. E. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The physics of fluids*, *8*(12), 2182–2189.

Hasegawa, K., & Shinoda, H. (2018). Aerial vibrotactile display based on multiunit ultrasound phased array. *IEEE Transactions on Haptics*, *11*(3), 367–377.

He, X., Liu, N., Li, S., Wang, H., & Wang, G. (2012). Local poisson sph for viscous incompressible fluids. *Computer Graphics Forum*, *31*(6), 1948–1958.

Hoetzlein, R. K. (2016). Gvdb: Raytracing sparse voxel database structures on the gpu. *Proceedings of high performance graphics* (pp. 109–117).

Hoshi, T., Takahashi, M., Iwamoto, T., & Shinoda, H. (2010). Noncontact tactile display based on radiation pressure of airborne ultrasound. *IEEE Transactions on Haptics*, *3*(3), 155–165.

Howard, T., Gallagher, G., Lécuyer, A., Pacchierotti, C., & Marchal, M. (2019). Investigating the recognition of local shapes using mid-air ultrasound haptics.

Ihmsen, M., Cornelis, J., Solenthaler, B., Horvath, C., & Teschner, M. (2013). Implicit incompressible sph. *IEEE transactions on visualization and computer graphics*, *20*(3), 426–435.

Inoue, S., Makino, Y., & Shinoda, H. (2015). Active touch perception produced by airborne ultrasonic haptic hologram. *2015 IEEE World Haptics Conference (WHC)*, 362–367.

Iodice, M., Frier, W. T. A., Wilcox, J., Long, B., & Georgiou, O. (2018). Pulsed schlieren imaging of ultrasonic haptics and levitation using phased arrays. *arXiv preprint arXiv:1810.00258*.

Irving, G., Schroeder, C., & Fedkiw, R. (2007). Volume conserving finite element simulations of deformable models. *ACM Trans. Graph.*, *26*(3).

Islam, M. S., & Wang, Q. (2020). A data driven heuristic for rapid convergence of general scheduled relaxation jacobi (srj) schemes. *arXiv preprint arXiv:2011.06636*.

Iwamoto, T., Tatezono, M., & Shinoda, H. (2008). Non-contact method for producing tactile sensation using airborne ultrasound. *Proceedings of the 6th International Conference on Haptics: Perception, Devices and Scenarios*, 504–513.

Jang, J., & Park, J. (2020). Sph fluid tactile rendering for ultrasonic mid-air haptics. *IEEE Transactions on Haptics*, *13*(1).

Jiang, C., Schroeder, C., Selle, A., Teran, J., & Stomakhin, A. (2015). The affine particle-in-cell method. *ACM Trans. Graph.*, *34*(4), 51:1–51:10.

Kappus, B., & Long, B. (2018). Spatiotemporal modulation for mid-air haptic feedback from an ultrasonic phased array. *The Journal of the Acoustical Society of America*, *143*(3), 1836–1836.

Kaufman, D. M., Sueda, S., James, D. L., & Pai, D. K. (2008). Staggered projections for frictional contact in multibody systems. *ACM Trans. Graph. (SIGGRAPH Asia)*, *27*(5), 164:1–164:11.

Kelager, M. (2006). Lagrangian fluid dynamics using smoothed particle hydrodynamics. *Camb. Monogr. Mech.*, *77*.

Kim, B., Liu, Y., Llamas, I., & Rossignac, J. R. (2005). *Flowfixer: Using bfecc for fluid simulation* (tech. rep.). Georgia Institute of Technology.

Kim, D., Song, O.-y., & Ko, H.-S. (2008). A semi-lagrangian cip fluid solver without dimensional splitting. *Computer Graphics Forum*, *27*(2), 467–475.

Klár, G., Gast, T., Pradhana, A., Fu, C., Schroeder, C., Jiang, C., & Teran, J. (2016). Drucker-prager elastoplasticity for sand animation. *ACM Trans. Graph.*, *35*(4), 103:1–103:12.

Korres, G., & Eid, M. (2016). Haptogram: Ultrasonic point-cloud tactile stimulation. *IEEE Access*, *4*, 7758–7769.

Krause, F.-L., & Lüddemann, J. (1997). Virtual clay modelling. *Proceedings of the Fifth IFIP TC5/WG5.2 International Workshop on Geometric Modeling in Computer Aided Design on Product Modeling for Computer Integrated Design and Manufacture*, 162–175.

Larionov, E., Batty, C., & Bridson., R. (2017). Variational stokes: A unified pressure-viscosity solver for accurate viscous liquids. *To appear in ACM Transactions on Graphics*.

Lee, J., Han, G., & Choi, S. (2008). Haptic pottery modeling using circular sector element method. *Haptics: Perception, Devices and Scenarios*.

Lenaerts, T., & Dutré, P. (2009). Mixing fluids and granular materials. *Computer Graphics Forum (Proceedings of Eurographics 2009)*, *28*(2), 213–218.

Lloyd, S. (2006). Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, *28*(2), 129–137.

Long, B., Seah, S. A., Carter, T., & Subramanian, S. (2014). Rendering volumetric haptic shapes in mid-air using ultrasound. *ACM Transactions on Graphics (TOG)*, *33*(6), 181.

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

Losasso, F., Gibou, F., & Fedkiw, R. (2004). Simulating water and smoke with an octree data structure. *Acm siggraph 2004 papers* (pp. 457–462).

Macklin, M., & Müller, M. (2013). Position based fluids. *ACM Trans. Graph.*, *32*(4), 104:1–104:12.

Macklin, M., Müller, M., & Chentanez, N. (2016). Xpbd: Position-based simulation of compliant constrained dynamics. *Proceedings of the 9th International Conference on Motion in Games*, 49–54.

Macklin, M., Müller, M., Chentanez, N., & Kim, T.-Y. (2014). Unified particle physics for real-time applications. *ACM Trans. Graph.*, *33*(4), 153:1–153:12.

McAdams, A., Sifakis, E., & Teran, J. (2010). A parallel multigrid poisson solver for fluids simulation on large grids. *Symposium on Computer Animation*, 65–73.

McCormick, S. F. (1987). *Multigrid methods*. SIAM.

McDonnell, K. T., Qin, H., & Wlodarczyk, R. A. (2001). Virtual clay: A real-time sculpting system with haptic toolkits. *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, 179–190.

Meagher, D. (1982). Geometric modeling using octree encoding. *Computer graphics and image processing*, *19*(2), 129–147.

Molemaker, J., Cohen, J. M., Patel, S., Noh, J., et al. (2008). Low viscosity flow simulations for animation. *Symposium on Computer Animation*, *2008*.

Monaghan, J. J. (1992). Smoothed particle hydrodynamics. *Annu. Rev. Astronomy and Astrophysics*, *30*, 543–574.

Monnai, Y., Hasegawa, K., Fujiwara, M., Yoshino, K., Inoue, S., & Shinoda, H. (2014). Haptomime: Mid-air haptic interaction with a floating virtual screen. *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, 663–667.

Mora, J., & Lee, W.-S. (2008). Real-time 3d fluid interaction with a haptic user interface. *3D User Interfaces, 2008. 3DUI 2008. IEEE Symposium on*, 75–81.

Morales, R., Marzo, A., Subramanian, S., & Martínez, D. (2019). Leviprops: Animating levitated optimized fabric structures using holographic acoustic tweezers. *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, 651–661.

Morales González, R., Freeman, E., & Georgiou, O. (2020). Levi-loop: A mid-air gesture controlled levitating particle game. *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, 1–4.

Morales González, R., Marzo, A., Freeman, E., Frier, W., & Georgiou, O. (2021). Ultrapower: Powering tangible & wearable devices with focused ultrasound. *Proceedings of the Fifteenth International Conference on Tangible, Embedded, and Embodied Interaction*, 1–13.

Müller, M., Keiser, R., Nealen, A., Pauly, M., Gross, M., & Alexa, M. (2004). Point based animation of elastic, plastic and melting objects. *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 141–151.

Müller, M., Charypar, D., & Gross, M. (2003). Particle-based fluid simulation for interactive applications. *ACM SIGGRAPH/Eurographics Sympos. on Comput. Anim.*

Müller, M., Heidelberger, B., Hennix, M., & Ratcliff, J. (2006). Position based dynamics. *Proc. of 3rd workshop on Virtual Reality Interaction and Physical Simulation*.

Museth, K. (2013). Vdb: High-resolution sparse volumes with dynamic topology. *ACM transactions on graphics (TOG)*, *32*(3), 1–22.

Museth, K., Lait, J., Johanson, J., Budsberg, J., Henderson, R., Alden, M., Cucka, P., Hill, D., & Pearce, A. (2013). Openvdb: An open-source data structure and toolkit for high-resolution volumes. *Acm siggraph 2013 courses* (pp. 1–1).

Nguyen, D. Q., Fedkiw, R., & Jensen, H. W. (2002). Physically based modeling and animation of fire. *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 721–728.

Ochiai, Y., Kumagai, K., Hoshi, T., Rekimoto, J., Hasegawa, S., & Hayasaki, Y. (2016). Fairy lights in femtoseconds: Aerial and volumetric graphics rendered by focused femtosecond laser combined with computational holographic fields. *ACM Transactions on Graphics (TOG)*, *35*(2), 1–14.

Osher, S., & Sethian, J. A. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *Journal of computational physics*, *79*(1), 12–49.

Otaduy, M. A., Okamura, A., & Subramanian, S. (2016). Haptic technologies for direct touch in virtual reality. *ACM SIGGRAPH 2016 Courses*, 13:1–13:123.

Paiva, A., Petronetto, F., Lewiner, T., & Tavares, G. (2006). Particle-based non-newtonian fluid animation for melting objects. *Brazilian Symposium on Computer Graphics and Image Processing*, 78–85.

Peer, A., & Teschner, M. (2017). Prescribed velocity gradients for highly viscous sph fluids with vorticity diffusion. *IEEE Transactions on Visualization and Computer Graphics*, *to appear*.

Peer, A., Ihmsen, M., Cornelis, J., & Teschner, M. (2015). An implicit viscosity formulation for sph fluids. *ACM Trans. Graph.*, *34*(4), 114:1–114:10.

Pihuit, A., Kry, P. G., & Cani, M. (2008). Hands on virtual clay. *2008 IEEE Intl Conf on Shape Modeling and Applications*.

Ram, D., Gast, T., Jiang, C., Schroeder, C., Stomakhin, A., Teran, J., & Kavehpour, P. (2015). A material point method for viscoelastic fluids, foams and sponges. *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 157–163.

Reardon, G., Shao, Y., Dandu, B., Frier, W., Long, B., Georgiou, O., & Visell, Y. (2019). Cutaneous wave propagation shapes tactile motion: Evidence from air-coupled ultrasound. *IEEE World Haptics Conference (WHC)*, 628–633.

Ribeiro, P. C., de Campos Velho, H. F., & Lopes, H. (2016). Helmholtz–hodge decomposition and the analysis of 2d vector field ensembles. *Computers & Graphics*, *55*, 80–96.

Robert, A. (1981). A stable numerical integration scheme for the primitive meteorological equations. *Atmosphere-Ocean*, *19*(1), 35–46.

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

Shakeri, G., Freeman, E., Frier, W., Iodice, M., Long, B., Georgiou, O., & Andersson, C. (2019). Three-in-one: Levitation, parametric audio, and mid-air haptic feedback. *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–4.

Solenthaler, B., & Pajarola, R. (2009). Predictive-corrective incompressible sph. *ACM transactions on graphics (TOG)*, *28*(3), 40.

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

Sommer, A., Schwanecke, U., & Schoemer, E. (2020). Chebyshev's method on projective fluids.

Spelmezan, D., Sahoo, D. R., & Subramanian, S. (2016). Sparkle: Towards haptic hover-feedback with electric arcs. *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 55–57.

Stam, J. (2009). Nucleus: Towards a unified dynamics solver for computer graphics. *Computer-Aided Design and Computer Graphics, 2009. CAD/Graphics '09. 11th IEEE International Conference on*, 1–11.

Stam, J. (1999). Stable fluids. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 121–128.

Steinhoff, J., & Underhill, D. (1994). Modification of the euler equations for "vorticity confinement": Application to the computation of interacting vortex rings. *Physics of Fluids*, *6*(8), 2738–2744.

Stomakhin, A., Schroeder, C., Chai, L., Teran, J., & Selle, A. (2013). A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, *32*(4), 102.

Stomakhin, A., Schroeder, C., Jiang, C., Chai, L., Teran, J., & Selle, A. (2014). Augmented mpm for phase-change and varied materials. *ACM Trans. Graph.*, *33*(4), 138:1–138:11.

Sulsky, D., Zhou, S.-J., & Schreyer, H. L. (1995). Application of a particle-in-cell method to solid mechanics. *Computer physics communications*, *87*(1-2), 236–252.

Takahashi, R., Hasegawa, K., & Shinoda, H. (2018). Lateral modulation of midair ultrasound focus for intensified vibrotactile stimuli. *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*, 276–288.

Takahashi, T., Dobashi, Y., Fujishiro, I., & Nishita, T. (2016). Volume preserving viscoelastic fluids with large deformations using position-based velocity corrections. *The Visual Computer*, *32*(1), 57–66.

Takahashi, T., Dobashi, Y., Fujishiro, I., Nishita, T., & Lin, M. C. (2015). Implicit formulation for sph-based viscous fluids. *Computer Graphics Forum*, *34*(2), 493–502.

Takahashi, T., & Lin, M. C. (2019). Video-guided real-to-virtual parameter transfer for viscous fluids. *ACM Transactions on Graphics (TOG)*, *38*(6), 1–12.

Takahashi, T., Nishita, T., & Fujishiro, I. (2014). Fast simulation of viscous fluids with elasticity and thermal conductivity using position-based dynamics. *Computers & Graphics*, *43*, 21–30.

Tampubolon, A. P., Gast, T., Klar, G., Fu, C., Teran, J., Jiang, C., & Museth, K. (2017). Multi-species simulation of porous sand and water mixtures. *ACM Transactions on Graphics*, *36*(4), To appear.

Terzopoulos, D., & Fleischer, K. (1988). Modeling inelastic deformation: Viscolelasticity, plasticity, fracture. *SIGGRAPH Comput. Graph.*, *22*(4), 269–278.

Tournier, M., Nesme, M., Gilles, B., & Faure, F. (2015). Stable constrained dynamics. *ACM Trans. Graph.*, *34*(4), 132:1–132:10.

Tsalamlal, M. Y., Issartel, P., Ouarti, N., & Ammi, M. (2014). Hair: Haptic feedback with a mobile air jet. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2699–2706.

Verschoor, M., Lobo, D., & Otaduy, M. A. (2018). Soft hand simulation for smooth and robust natural interaction. *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 183–190.

Wang, H. (2015). A chebyshev semi-iterative approach for accelerating projective and position-based dynamics. *ACM Transactions on Graphics (TOG)*, *34*(6), 1–9.

Wang, H., O'Brien, J., & Ramamoorthi, R. (2010). Multi-resolution isotropic strain limiting. *ACM Trans. Graph.*, *29*(6), 156:1–156:10.

Weiler, M., Koschier, D., & Bender, J. (2016). Projective fluids. *Proceedings of the 9th International Conference on Motion in Games*, 79–84.

Weiler, M., Koschier, D., Brand, M., & Bender, J. (2018). A physically consistent implicit viscosity solver for sph fluids. *Computer Graphics Forum*, *37*(2), 145–155.

Wilson, G., Carter, T., Subramanian, S., & Brewster, S. A. (2014). Perception of ultrasonic haptic feedback on the hand: Localisation and apparent motion. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1133–1142.

Wojtan, C., & Turk, G. (2008). Fast viscoelastic behavior with thin features. *ACM Trans. Graph.*, *27*(3), 47:1–47:8.

Wu, K., Truong, N., Yuksel, C., & Hoetzlein, R. (2018). Fast fluid simulations with sparse volumes on the gpu. *Computer Graphics Forum*, *37*(2), 157–167.

Xu, L., Lu, Y., & Liu, Q. (2018). Integrating viscoelastic mass spring dampers into position-based dynamics to simulate soft tissue deformation in real time. *R Soc Open Sci*, *14*(5).

Yamane, K., & Nakamura, Y. (2006). Stable penalty-based model of frictional contacts. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 1904–1909.

Yang, M., Lu, J., Safonova, A., & Kuchenbecker, K. J. (2009). Gpu methods for real-time haptic interaction with 3D fluids. *Haptic Audio visual Environments and Games, 2009. HAVE 2009. IEEE International Workshop on*, 24–29.

Yue, Y., Smith, B., Batty, C., Zheng, C., & Grinspun, E. (2015). Continuum foam: A material point method for shear-dependent flows. to appear. *ACM Trans Graph*, *2*, 25–35.

Zhu, B., Lee, M., Quigley, E., & Fedkiw, R. (2015). Codimensional non-newtonian fluids. *ACM Trans. Graph.*, *34*(4), 115:1–115:9.

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

Zhu, Y., & Bridson, R. (2005). Animating sand as a fluid. *ACM Transactions on Graphics (TOG)*, *24*(3), 965–972.

Resumen

A

Las tecnologías de realidad virtual (RV) están comprometidas con el desarrollo de soluciones que permitan hacer realidad la anhelada visión de crear mundos sintéticos inmersivos que trasciendan los límites de la realidad. Aunque la RV se encuentra en el punto más álgido de su historia gracias a la confluencia de varias tecnologías desarrolladas durante décadas, la diversidad de interacciones que pueden representarse en las experiencias de RV sigue siendo limitada.

Mientras que la interacción con objetos sólidos y cuerpos deformables ha atraído una gran atención por parte de los investigadores, otros medios interesantes, como los fluidos, han sido ignorados en gran medida. En el caso particular de los fluidos, esto se debe principalmente a una combinación de dos factores. En primer lugar, los métodos interactivos para la simulación de fluidos son incapaces de representar materiales que no sean fluidos no viscosos. Esto limita mucho la capacidad de reproducir elementos cotidianos e interesantes como la miel, la nata montada, la pintura o la arcilla. En segundo lugar, los dispositivos hápticos convencionales tienen dificultades para proporcionar un contacto convincente con los medios fluidos, especialmente en aplicaciones que requieren una manipulación directa.

En esta tesis, investigamos estrategias para superar las limitaciones actuales del estado del arte con el fin de permitir el contacto físico con fenómenos virtuales ricos y complejos como los fluidos. Para ello, abordamos este problema desde dos perspectivas: la primera, mediante el desarrollo de un novedoso modelo de simulación de fluidos extremadamente eficiente, capaz de representar una amplia gama de materiales en entornos virtuales. La segunda, mediante el desarrollo de métodos capaces de representar la interacción táctil con dichos medios utilizando tecnología háptica moderna, como los dispositivos de ultrasonido multielemento. Finalmente, demostramos la practicalidad de los métodos propuestos y abordamos el reto de la simulación virtual de la interacción con materiales tipo arcilla, alcanzando un grado de realismo sin precedentes en la manipulación de materiales que presentan un comportamiento viscoplástico extremo.

Proporcionamos un resumen del desarrollo de esta tesis a lo largo de las distintas secciones de este capítulo: antecedentes, objetivos, metodología, resultados y conclusiones.

A.1 Antecedentes

A.1.1 Simulación de Fluidos en los Gráficos por Computador

La simulación de fluidos ha sido uno de los temas de investigación más activos en el campo de los gráficos por ordenador durante las últimas tres décadas, dando lugar a una gran cantidad de trabajos dedicados a modelar fenómenos fluidos atractivos para una amplia variedad de aplicaciones como largometrajes, anuncios publicitarios, videojuegos o simulación médica.

Sin embargo, a pesar de su amplia aplicación, la simulación de fenómenos fluidos sigue siendo una tarea difícil. Aunque hoy en día la mecánica de fluidos se entiende bastante bien, describir numéricamente su comportamiento requiere la resolución de ecuaciones diferenciales no lineales computacionalmente complejas. En consecuencia, los investigadores han limitado históricamente sus simulaciones a la representación de fenómenos superficiales para entornos en los que se requiere interactividad, relegando la simulación de volúmenes de fluidos a aplicaciones fuera de línea.

A lo largo de los años, los investigadores se han concentrado en desarrollar cuidadosas aproximaciones con el fin de reducir significativamente el coste computacional asociado a la simulación dinámica de tipos específicos de fluidos. Avances como los esquemas de advección estables y baja disipación, los solvers de dinámica restringida para garantizar la incompresibilidad de los fluidos o los enfoques numéricos avanzados, como las técnicas multigrid que hacen un sofisticado tratamiento de las condiciones de contorno, han dado lugar al desarrollo de solucionadores de dinámica de fluidos de alto rendimiento y alta resolución. Estos avances, junto con la popularización de las Unidades de Procesamiento Gráfico (GPU) como her-

ramientas computacionales masivamente paralelas, han permitido el desarrollo de métodos adecuados para su aplicación en entornos interactivos (Crane et al., 2007; Macklin & Müller, 2013).

Sin embargo, estos métodos de simulación se han centrado principalmente en la representación de materiales fluidos newtonianos de baja viscosidad e incompresibles, como el agua. Muchas sustancias cotidianas, como la miel, el ketchup, la nata montada y la arcilla, presentan comportamientos altamente viscosos, viscoelásticos o no newtonianos (es decir, cuya viscosidad depende de la tasa de cizallamiento o del histórico) que no puede reproducirse con dichas formulaciones. Por lo tanto, para cumplir el objetivo de ampliar las posibilidades de interacción de los fluidos en la RV, sería interesante encontrar métodos que permitieran la simulación de tales materiales en tiempo real.

Como es lógico, la simulación de alta viscosidad plantea retos computacionales adicionales. Se requieren formulaciones implícitas para resolver de forma robusta las ecuaciones diferenciales numéricamente rígidas que estos materiales plantean. Además, debido a la dificultad de calcular la deformación del fluido, únicamente podemos aspirar a aproximarla, lo que resulta en deriva numérica y se traduce en una pérdida perceptible de viscoelasticidad. Algunos trabajos han explorado el modelado de materiales de viscosidad moderada a alta para aplicaciones interactivas (Alduán et al., 2017; Macklin & Müller, 2013; T. Takahashi et al., 2016; T. Takahashi et al., 2014). Sin embargo, se limitan a modelar la viscosidad newtoniana, y no logran alcanzar un comportamiento viscoso extremo sin introducir artefactos en forma de deriva excesiva u oscilaciones elásticas no deseadas.

A.1.2 Renderizado Háptico Aéreo

Aunque nuestros sentidos nos permiten percibir y comprender distintas aspectos de la realidad en la que vivimos, el tacto es el único sentido que nos une al mundo. Permite al ser humano relacionarse con su entorno y manipularlo físicamente, revelando características de los objetos que no suelen ser discernibles a través de otros sentidos: forma, rígidez, rugosidad, textura o calor, por ejemplo. Sin embargo, su estudio y comprensión (háptica) está muy por detrás del grado de comprensión de otros sentidos como la vista o el oído.

En consonancia con el actual renacimiento de la realidad virtual (RV), los dispositivos hápticos (es decir, la tecnología capaz de estimular el sentido del tacto) también han crecido rápidamente en popularidad debido a los importantes beneficios que aportan a la experiencia de la interacción persona-ordenador (HCI). Como resultado, hemos asistido a la aparición de nuevas tecnologías hápticas que emplean diversos principios de actuación para permitir un tacto virtual convincente directamente con nuestras manos.

Las pantallas hápticas aéreas son un ejemplo prometedor de estas tecnologías emergentes. Las pantallas volumétricas, junto con las tecnologías de seguimiento de las manos, permiten liberar a los usuarios de las limitaciones de otras tecnologías de renderizado, como las superficies de contacto o los dispositivos portátiles, al tiempo que amplían considerablemente su espacio de trabajo. Aunque los investigadores han desarrollado pantallas basadas en diversos fenómenos físicos (por ejemplo, flujos de aire (Tsalamlal et al., 2014), láseres (Ochiai et al., 2016) o arcos eléctricos (Spelmezan et al., 2016)), las basadas en equipos ultrasónicos multielemento (Hoshi et al., 2010; Long et al., 2014) son las que han ganado más popularidad en los últimos años gracias a su baja latencia, gran tamaño de los estímulos y amplio espacio de trabajo (Frier, 2020).

Sin embargo, la generación de percepciones táctiles con estos dispositivos sigue siendo un proceso en gran medida desconocido debido a la ausencia de un modelo computacional que mapee los patrones de activación a la percepción. Los investigadores se han centrado principalmente en la representación de objetos holográficos, desarrollando diferentes metáforas de alto nivel para comandar estos dispositivos con tales fines. Sin embargo, estas metáforas de mando son insuficientes por sí solas para representar las interacciones con fluidos.

Cuando interactuamos con fluidos, nuestra piel está sometida a un campo de presión temporal y espacialmente variable cuyas características vienen determinadas por nuestro movimiento y las propiedades inherentes al flujo. Por ello, la interacción táctil con estos medios utilizando háptica ultrasónica podría plantearse como el problema de reproducir dinámicamente un campo de presión en la piel del usuario. Hasta la fecha, ninguna metáfora de mando goza de la capacidad de reproducir directamente un campo de presión arbitrario que varía espacialmente.

A.1.3 Acoplando Fluidos y Hápticos

La representación háptica de la interacción con fluidos ha recibido gran atención por parte de los investigadores. Los distintos métodos existentes abordan el reto de ejecutar simulaciones interactivas de fluidos y proponen diferentes estrategias de acoplamiento entre el dispositivo háptico y el medio simulado.

Tradicionalmente, la mayoría de los trabajos que abordan el renderizado de medios fluidos giran en torno al uso de dispositivos similares a herramientas (Baxter & Lin, 2004; Dobashi et al., 2006; Mora & Lee, 2008). La llegada de las GPU como plataformas de cálculo ha permitido la aplicación de métodos interactivos más ricos. Yang et al. (2009) aprovechó esta tecnología e implementó métodos para acumular las fuerzas de interacción directamente en la GPU. Cirio, Marchal, Otaduy, et al. (2013) realizó una simulación del fluido basada en partículas en la GPU y utilizó un método de acoplamiento virtual para transferir la interacciones entre el fluido y la herramienta al dispositivo háptico.

La representación de la interacción táctil con entornos virtuales (incluidos los fluidos) mediante dispositivos hápticos ultrasónicos puede formularse como un problema de reproducción dinámica de un campo de presión en la piel del usuario. Sin embargo, ninguna de las metáforas de comandado existentes por sí solas sastifacen las necesidades de este problema, ya que no son capaces de reproducir un campo de presión espacialmente variable.

A.1.4 Simulación de Arcilla Virtual

La arcilla es un material viscoplástico cuyo comportamiento se asemeja tanto a un sólido como a un fluido. Presenta enlaces materiales microscópicos que preservan la forma, pero estos enlaces son frágiles y el material fluye incluso bajo pequeñas tensiones, aunque con una viscosidad muy alta. Debido a esta complejidad, la simulación de la arcilla es un problema computacionalmente difícil, y los métodos interactivos existentes apenas aproximan a su verdadero comportamiento. Si bien varios trabajos han tratado de modelar parcialmente la arcilla en sus diferentes modalidades para aplicaciones de RV, incluyendo retroalimentación háptica (Chaudhury & Chaudhuri, 2014; Dewaele & Cani, 2003; Han et al., 2007;

Krause & Lüddemann, 1997; Lee et al., 2008; McDonnell et al., 2001; Pihuit et al., 2008), ningún método de simulación interactiva anterior reproduce el flujo altamente viscoso y la fractura dúctil de los materiales arcillosos.

A.2 Objetivos

El objetivo general de esta tesis es permitir la interacción física con fenómenos virtuales ricos y complejos como los fluidos. Este objetivo general se materializa a través de dos interesantes retos: la creación de modelos novedosos para la simulación de fluidos viscoelásticos y viscoplásticos adecuados para la interacción en tiempo real, y el desarrollo de diversos algoritmos de renderizado háptico que permitan a los usuarios interactuar de forma natural con este tipo de medios.

A.3 Metodología

Para la realización de esta tesis hemos seguido la siguiente metodología:

A.3.1 Revisión Bibliográfica

Con el fin de entender el alcance del problema que aborda la tesis, hemos llevamos a cabo un exhaustivo análisis bibliográfico en materia de simulación de fluidos y renderizado háptico. Dicho análisis se recoge en el Capítulo 2. No obstante, dado el enorme cuerpo bibliográfico dedicado a estas materias, es imposible cubrir con gran nivel de detalle todos los trabajos relacionados en unas pocas páginas. Es por ello que en dicho capítulo nos centramos principalmente en aquellos trabajos que sirven de base para el desarrollo de esta tesis.

Comenzamos con la identificación de los trabajos más relevantes en simulación de fluidos, analizando las numerosas estrategias típicamente empleadas en la literatura de gráficos por computador, al tiempo que nos centramos en aquellos

enfoques que permiten la simulación de dinámica de fluidos de alto rendimiento. A continuación, estudiamos las particularidades y consideraciones necesarias para la simulación de medios más interesantes, enfatizando en aquellos trabajos que abordan la simulación de materiales fluidos viscoelásticos.

Tras identificar y abordar el modelado de dicho tipo de materiales, revisamos la literatura al respecto del renderizado háptico, haciendo especial hincapié en las tecnologías que pudieran servir para alcanzar nuestro objetivo de reproducir la interacción con dichos medios, como los dispositivos de ultrasonidos multielemento. Tras estudiar y comprender los conceptos y principios fundamentales que subyacen a estas tecnologías, estudiamos las técnicas que permiten su comandado, limitaciones e implicaciones perceptivas.

Finalmente, estudiamos las posibles aplicaciones de los métodos desarrollados, identificando así el modelado de materiales tipo arcilla como una prometedora aplicación.

A.3.2 Estudio y Desarrollo de un Modelo de Simulación de Fluidos Viscoelásticos

Analizando el estado del arte en modelado de fluidos de alto rendimiento, observamos que no existía ningún modelo que resolviera el tratamiento de fluidos extremadamente viscosos y viscoelasticos de forma satisfactoria.

En el Capítulo 3 de esta tesis, abordamos este reto y proponemos un método para la simulación de fluidos altamente viscosos y viscoelásticos que es adecuado para aplicaciones interactivas. La clave de su alto rendimiento reside en el uso de solvers de dinámica restringida como Position-based Dynamics (PBD) como alternativa a las formulaciones implícitas para describir estos materiales. Nuestra solución se inspira en un modelo constitutivo de fluidos poliméricos (es decir, fluidos en los que se disuelven polímeros elásticos), que admite una amplia gama de comportamientos de viscoelasticidad bajo una formulación común.

Tras diseñar e implementar nuestro método, llevamos a cabo un análisis de su parametrización, estabilidad y posibles deficiencias. Como fruto de este análisis

se llevó a cabo el desarrollo de un método de resolución de dinámica *doblemente restringida* (al que bautizamos como DC-PBD), que hereda la robustez y eficiencia del método PBD original, pero muestra una estabilidad mejorada bajo restricciones basadas en la velocidad como las de nuestra formulación, especialmente con grandes pasos de tiempo.

Finalmente, estudiamos su aplicación y generamos las diversas secuencias que demuestran su rango de aplicabilidad.

A.3.3 Estudio y Desarrollo de Algoritmos de Renderizado Táctil

Tras la elección de los dispositivos de ultrasonidos multielemento como potencial candidato para reproducir la interacción con fluidos virtuales, pasamos a abordar el problema de comandar su actuación para proporcionar una sensación similar a las interacciones virtuales.

En los Capítulos 4 y 5 abordamos este problema introduciendo dos algoritmos de renderizado que mapean dinámicamente campos de presión arbitrarios a metáforas de control de modulación en amplitud (AM) y modulación espaciotemporal (STM) respectivamente. Nuestros enfoques plantean este mapeo como problemas de optimización que tienen en cuenta las limitaciones perceptivas y técnicas conocidas de ambos métodos, dando como resultado el comandado óptimo que mejor reconstruye las presiones a evaluadas en la simulación.

Cronológicamente, el primer método desarrollado fue el basado en AM debido a su simplicidad. En dicho momento la actuación basada en STM planteaba múltiples incógnitas perceptuales relacionadas con la existencia de un componente temporal en la actuación cuyas variables apenas se habían comenzado a comprender. Tras implementar un método de simulación de medios gaseosos y explorar las diferentes estrategias de optimización, alcanzamos la que sería nuestra estrategia final. Tras ello realizamos un estudio cuantitativo de los diferentes parámetros afectando la reconstrucción y procedimos a la generación de los casos de demostración.

En este lapso de tiempo, el conocimiento sobre los métodos STM avanzó, dilucidando así la relación entre dicha metáfora de mando, la propagación de ondas en la piel y alguna de sus claves perceptuales. Tras reproducir los resultados de los trabajos que abarcaban dichas relaciones, identificamos los criterios que posteriormente darían lugar a nuestra asunción de persistencia, que nos permitiría alcanzar soluciones computacionalmente eficientes al asumir que durante un breve lapso de tiempo las presiones ejercidas son equivalentes a las de un campo cuasi-estático. Tras diseñar, implementar y analizar diferentes estrategias, obtuvimos nuestra estrategia de optimización en dos pasos. Hecho esto, llevamos a cabo un estudio perceptual para determinar su rendimiento con respecto a nuestro método anterior, demostrando así su superioridad en tareas de discriminación. Finalmente, llevamos a cabo la generación de los casos de demostración empleando el mismo método de simulación que en el caso del algoritmo basado en AM.

A.3.4 Estudio e Implementación de Métodos para la Interacción Natural con Arcilla

Finalmente, tras haber desarrollado independientemente los diferentes métodos para la simulación e interacción fluidos, estudiamos las posibles aplicaciones que permitieran aunarlos de forma práctica. Entre las diferentes opciones, elegimos la representación de materiales altamente viscoplásticos como la arcilla por su interés como caso representativo de manipulación diestra en entornos virtuales.

En el Capítulo 6 proponemos un modelo de simulación que permite al usuario conformar, dividir y fusionar la arcilla virtual de forma muy similar a la del mundo real. Nos basamos en una versión simplificada del método presentado en el capítulo Capítulo 3, extendida con un modelo de elastoplasticidad para capturar las principales características de los materiales arcillosos. Acoplamos este método con un modelo existente de simulación de mano natural para lograr un acoplamiento bidireccional, y utilizamos las fuerzas de interacción resultantes para comandar un algoritmo de renderizado táctil basado en el método presentado en el Capítulo 4.

A.4 Resultados

Las principales contribuciones de esta tesis pueden resumirse de la siguiente manera:

- En el Capítulo 3 introducimos un modelo de viscoelasticidad basado en restricciones. Describimos un modelo constitutivo de viscoelasticidad en fluidos poliméricos, que es la contraparte basada en tensión de nuestras restricciones. Tras esto, describimos la derivación de las restricciones implícitas de conformación que actúan sobre las velocidades de los fluidos, así como los parámetros de nuestro modelo.

- En el Capítulo 3 proponemos un método de resolución de dinámica *doblemente restringida* (DC-PBD), que hereda la robustez y eficiencia del método PBD original, pero muestra una estabilidad mejorada bajo restricciones basadas en la velocidad como las de nuestra formulación, especialmente con grandes pasos de tiempo.

- En el Capítulo 3 discutimos la aplicación fenomenológica de nuestro enfoque a la representación de una variedad de materiales que van desde altamente viscosos a viscoelásticos, así como los detalles prácticos de implementación para lograr simulaciones de alto rendimiento, minimizando los artefactos numéricos que resultan en disipación cinética no deseada.

- En el Capítulo 4 introducimos un algoritmo eficiente para la representación de la interacción táctil con los fluidos empleando ultrasonidos, basado en la metáfora de mando de modulación en amplitud (AM). Caracterizamos la actuación del dispositivo mediante un conjunto de puntos focales, optimizando la ubicación e intensidad de dichos puntos para aproximar lo mejor posible un campo de presiones sobre la piel. La clave de la eficacia de nuestra solución es la suposición de que la presión en un punto del espacio depende únicamente del punto focal más cercano, lo que permite desacoplar la ubicación e intensidad en problemas de optimización separados. El método resultante produce una experiencia convincente mientras se interactúa dinámicamente con el fluido virtual.

- En el Capítulo 5 proponemos otro algoritmo de renderizado eficiente, esta vez basado en la metáfora de la modulación espacio-temporal (STM), que caracteriza la actuación del dispositivo a través del control de las trayectorias de los puntos focales. Proponemos una optimización a dos niveles (global y local) para renderizar la distribución de fuerzas resultante de una interacción virtual dinámica. Un aspecto clave de nuestro método es plantear el renderizado STM como un problema cuasi-estático, eliminando así la variable temporal de cada actualización de renderizado dinámico y consiguiendo que el problema sea computacionalmente manejable.

- En el Capítulo 5 también se compara la calidad de la reconstrucción del método con respecto al algoritmo presentado en el Capítulo 4 observando que nuestro algoritmo basado en STM consigue proporcionar una cobertura mayor y más suave que el método basado en AM, a la vez que muestra un rendimiento superior en las tareas de discriminación.

- Por último, en el Capítulo 6, proponemos una solución computacional para la simulación interactiva de materiales similares a la arcilla con un realismo sin precedentes, junto con una representación táctil que proporciona una experiencia tangible natural. Nuestra solución amplía los métodos propuestos en Capítulos 3 y 4 al incluir un novedoso modelo de elastoplasticidad y una formulación del problema de optimización que considera una serie de pesos perceptuales a la hora de determinar la solución. Nuestro algoritmo toma como entrada las fuerzas de interacción entre un modelo de mano virtual y el material similar a la arcilla. Demostramos la eficacia de nuestro método mediante supuestos creativos.

A.5 Conclusiones

En el Capítulo 7 presentamos las conclusiones generales y específicadas de cada uno de los métodos propuestos en esta tesis, así como sus limitaciones y posibles líneas de investigación futuras.

El objetivo general de esta tesis era permitir la interacción física con fenómenos virtuales ricos y complejos como los fluidos. Lo hemos conseguido de dos maneras. En primer lugar, desarrollando un novedoso modelo de simulación de fluidos que es extremadamente eficiente, permitiendo la representación de una amplia gama de materiales en entornos virtuales. En segundo lugar, desarrollando varios métodos capaces de representar la interacción táctil con dichos medios utilizando tecnología háptica moderna, como los dispositivos ultrasónicos multielemento.

En el Capítulo 3 nos centramos en el diseño del citado modelo para la simulación eficiente de fluidos viscoelásticos. Nuestra formulación se basa en un modelo constitutivo para fluidos poliméricos, que describe las propiedades elásticas y viscosas del fluido en función de la evolución temporal de un tensor de conformación. Como resultado, nuestro método permite la representación de una amplia gama de materiales (desde invisibles hasta altamente viscosos o viscoelásticos) bajo una única formulación. Además, nuestro enfoque basado en restricciones permite el uso de solucionadores de dinámica restringida de alto rendimiento, lo que permite su implementación en escenas interactivas. El desarrollo de este trabajo culminó con la publicación del artículo *Conformation Constraints for Efficient Viscoelastic Fluid Simulation* (Barreiro et al., 2017) en la revista **ACM Transactions on Graphics** (JCR Q1).

Los Capítulos 4 y 5 están dedicados al desarrollo de novedosos algoritmos de interacción táctil para tecnologías hápticas emergentes como los dispositivos ultrasónicos multielemento. En la interacción con fluidos, los patrones de activación de estos dispositivos deben ser controlados para producir una sensación táctil que se asemeje a las interacciones con el fluido virtual. Conseguimos este objetivo planteando la actuación como un problema de optimización numérica, centrándonos en dos metáforas de control de alto nivel: la modulación en amplitud (AM) y la modulación espaciotemporal (STM). Optimizando la intensidad y las ubicaciones/trayectorias de los focos ultrasónicos a lo largo del tiempo, encontramos el conjunto de variables de control que producen la distribución de presiones en la piel que mejor se ajusta a las extraídas de la simulación. Además, y gracias a nuestro enfoque basado en optimización, también somos capaces de incorporar el conocimiento de las limitaciones técnicas y perceptuales de ambas metáforas de control para así maximizar la eficacia de nuestras soluciones. Hasta donde sabemos, ningún trabajo anterior había abordado el problema siguiendo un esquema similar.

Esta investigación culminó con una presentación en una conferencia y una publicación en una revista. El algoritmo de renderizado basado en AM se presentó en la **IEEE World Haptics Conference 2019** bajo el título *Ultrasound Rendering of Tactile Interaction with Fluids* (Barreiro et al., 2019), mientras que el algoritmo basado en STM fue publicado en la revista **IEEE Transactions on Haptics** (JCR Q2) bajo el título *Path Routing Optimization for STM Ultrasound Rendering* (Barreiro et al., 2020). La novedad de este último trabajo fue reconocida con el premio a la mejor ponencia en el congreso **IEEE Haptics Symposium 2020**.

Finalmente, en el Capítulo 6 hemos demostrado la aplicabilidad de los métodos presentados en esta tesis abordando la simulación virtual de la interacción con la arcilla. Para ello, hemos extendido el modelo presentado en el Capítulo 3 para incorporar el comportamiento elastoplástico característico que presentan los materiales arcillosos. Este modelo, junto a una simulación de mano en tiempo real y un algoritmo de renderizado háptico basado en el presentado en el Capítulo 4, permite la manipulación natural de materiales virtuales similares a la arcilla con un grado de realismo sin precedentes. Este trabajo ha culminado con una presentación en el congreso **IEEE World Haptics 2021** bajo el título *Natural Tactile Interaction with Virtual Clay* (Barreiro et al., 2021).

En términos generales, hemos presentado varios modelos y algoritmos altamente innovadores que contribuyen sustancialmente al avance del estado del arte en la simulación de fluidos y la interacción háptica. Su idoneidad se ha demostrado a través de múltiples casos prácticos que permiten la interacción de los usuarios con una variedad de materiales fluidos interesantes.

Dada la rápida evolución de las tecnologías de RV, su popularización y la demanda cada vez mayor de modelos de interacción realistas y sofisticados, confiamos en que las soluciones propuestas serán bien recibidas tanto por la industria como por los profesionales y, por tanto, servirán de punto de partida para seguir explorando en este campo.

Sin embargo, nuestras aportaciones no se limitan al campo de la RV, sino que llegan también a otras industrias. Tal es el caso de *Next Limit Technologies*, pionera en el desarrollo de soluciones *software* para la simulación de fluidos para efectos especiales (VFX). Como resultado de nuestra colaboración con ellos, uno de nuestros trabajos, concretamente el método descrito en el Capítulo 3, ha sido

integrado en el solucionador de partículas *Dyverso* (Alduán et al., 2017) incluido en el paquete *RealFlow*, recibiendo comentarios extraordinariamente positivos de artistas y directores técnicos.

Por último, estamos expectantes ante el potencial de inmersión en mundos virtuales que ofrecerá la tecnología del futuro. La estandarización de modelos mecánicos capaces de capturar con precisión el comportamiento de los tejidos blandos biológicos como la mano (Verschoor et al., 2018) permitirá una mejora significativa de la calidad de las interacciones en los mundos virtuales y su acoplamiento con otros fenómenos físicos. Cuando esto ocurra, los investigadores estarán en una posición privilegiada para desarrollar modelos perceptivos que proporcionen una información mayor y más fiable para dirigir las elecciones de los futuros métodos de renderizado táctil. Esto, junto con los continuos avances en las capacidades de procesamiento del ecosistema de la RV y la introducción de nuevos y más potentes algoritmos de aprendizaje automático capaces de manejar el flujo cada vez mayor de información sensorial, configurará el futuro de la RV y permitirá grados de detalle y realismo antes inconcebibles.