Six Degrees-of-Freedom Haptic Interaction with Fluids

Gabriel Cirio, Maud Marchal, Sébastien Hillaire, Anatole Lécuyer

Six Degrees-of-Freedom Haptic Interaction with Fluids

Gabriel Cirio, Maud Marchal, Sébastien Hillaire, and Anatole Lécuyer

Abstract—We often interact with fluids in our daily life, either through tools such as when holding a glass of water or directly with our body when we swim or we wash our hands. Multimodal interactions with virtual fluids would greatly improve the simulations realism, particularly through haptic interaction. However, achieving realistic, stable and real-time force feedback from fluids is particularly challenging. In this work, we propose a novel approach that allows real-time 6 Degrees of Freedom (DoF) haptic interaction with fluids of variable viscosity. Our haptic rendering technique, based on a Smoothed-Particle Hydrodynamics physical model, provides a realistic haptic feedback through physically-based forces. 6DoF haptic interaction with fluids is made possible thanks to a new coupling scheme and a unified particle model, allowing the use of arbitrary-shaped rigid bodies. Particularly, fluid containers can be created to hold fluid and hence transmit to the user force feedback coming from fluid stirring, pouring, shaking and scooping, to name a few. Moreover, we adapted an existing visual rendering algorithm to meet the frame rate requirements of the haptic algorithms. We evaluate and illustrate the main features of our approach through different scenarios, highlighting the 6DoF haptic feedback and the use of containers.

Index Terms—6DoF haptic interaction, computational fluid dynamics, smoothed-particle hydrodynamics, rigid bodies

1 Introduction

Virtual Reality technologies intend to immerse users inside a 3D synthetic world simulated in real-time by a computer. In a typical virtual reality setup, the user is provided with stimulations and sensory feedback on multiple senses such as vision, audition and/or touch (or haptic sense) by means of advanced displays. As an example, in surgical simulations, medical trainees can be provided with visual and haptic feedback that enable to better perceive contacts between the manipulated tools and the surrounding organs.

The computation of haptic feedback in real-time, also known as "haptic rendering", is still an open and active research area. Surprisingly, haptic interaction with fluids has been scarcely studied. Indeed, most current haptic simulations involve only rigid or deformable bodies. However, we often interact with fluids in our daily life, either through tools such as when holding a glass of water or directly with our body when we swim or we wash our hands. Fluids are also found in many applications such as for industrial or medical manipulations - involving for instance blood flow and natural liquids. Enabling haptic feedback in the interaction with fluids, besides allowing more realistic simulations, would enable a wide range of novel simulation scenarios and applications.

The haptic interactive simulation of fluids is particularly challenging, especially to achieve realistic, stable and real-time force-feedback using physically-based models. To simulate interactions between fluids and rigid bodies within haptic rendering, previous studies have proposed pre-computed ad-hoc algorithms [1], approaches featuring only 3 Degrees of Freedom (DoF) and non-viscous fluids [2], or implementations restricted to simple object shapes and small amounts of fluid [3]. Thus, as for today, there is a lack of models and rendering techniques handling complex 6DoF haptic interactions with viscous fluids in real-time.

In this work, we propose a novel approach for real-time 6DoF haptic interaction with viscous fluids through arbitrary shaped rigid bodies and 6DoF haptic devices. It represents a significant leap forward in interaction possibilities compared to previous work on haptic interaction with fluids. It is the first approach to allow a full 6DoF haptic interaction with viscous fluids. Until now, real-time interaction was restricted to simply "swiping" the surface of a fluid volume with simple objects, dramatically reducing the interaction possibilities when compared to real-life conditions. In this work, we show how many rich and complex fluid manipulations are now easily achieved, with stable force and torque outputs. Interaction is no longer limited to 3 degrees of freedom: not only torque feedback plays a major role in providing a compelling feeling of realism, it also opens an entirely new horizon of interaction techniques in Virtual Reality. Among these are the use of concave containers to hold fluid and hence transmit to the user force feedback coming, for example, from fluid stirring, pouring, shaking and scooping, as well as the inertia of the fluid inside the container.

This work also introduces the Smoothed-Particle Hydrodynamics (SPH) model into the fluid haptic simulation, and, to the best of our knowledge, to the haptic

- *G. Cirio, M. Marchal, S. Hillaire and A. Lécuyer are with INRIA Rennes, France.*
  E-mail: {gcirio, mmarchal, shillaire, alecuyer}@inria.fr
- *M. Marchal is also with INSA Rennes, France.*
- *S. Hillaire is also with Orange Labs, Rennes, France.*

realm in general, yielding positive results. Many issues of major concern for the haptic community are avoided in our work through the use of the SPH model, such as the generation of smooth and stable haptic forces, the transparent handling of multiple contact points, the parallelizability of computations and their scalability, the seamless use of arbitrary-shaped objects, and the computation of a fully dynamic world. By showing that haptic forces computed through the SPH model are particularly well suited for the interaction with fluids, haptic interaction with other media can now be imagined for the SPH model.

Hence, our major contributions are:

- **A novel 6DoF approach for the haptic interaction with viscous fluids**,
- **The introduction of the SPH model for haptic fluid computations**

In addition, other technical contributions of our paper concern:

- a generic scheme for the haptic coupling of any number of devices and rigid bodies,
- the GPU acceleration of the underlying fluid and rigid body model to achieve real-time speeds,
- the adaptation of an existing graphic rendering algorithm to meet high frame rate requirements

The paper is organized as follows: section 2 provides a summary of related work. In section 3 we describe our novel haptic rendering technique, initially coupled to a single particle and with 3DoF haptic feedback. The algorithm is extended in section 4 for 6DoF haptic rendering through the use of a unified particle model for rigid body interaction with fluids. Section 5 describes how we adapt a screen space fluid graphic rendering algorithm to meet the frame rate requirements of our rendering technique. We evaluate and illustrate our approach in section 6, highlighting its main features and interaction possibilities through different scenarios. Particularly, fluid containers can be created to hold fluid and hence transmit to the user force feedback coming, for example, from fluid stirring, pouring, shaking and scooping. Strong forces such as fluid resistance and weight can be captured, as well as light forces like the inertia of the fluid inside the container. We discuss our results and the limitations of our technique in section 7, before concluding.

2 Related Work

2.1 Real-Time Fluid Simulation

Two main physical approaches based on Navier-Stokes equations can be distinguished [4]: the Eulerian approach, where space is subdivided in a grid and the different physical quantities are computed for each grid cell, and the Lagrangian approach, where the fluid is treated as a system of independent particles, each with a position and a velocity. Although each approach has been extensively studied, we focus on the Lagrangian approach, since the simulated fluid is not bounded by a grid, is faster to compute and better preserves small-scale details, allowing richer interactions scenarios, which is especially interesting for haptic feedback. On the other hand, compared to Eulerian approaches, they have a lower order spatial accuracy and some interesting behaviors can disappear in low-density regions. For a detailed description of the existing approaches for the simulation of fluids, we refer the reader to [4].

Smoothed-Particle Hydrodynamics [5] (SPH) techniques are very popular among Lagrangian simulations. With SPH, each particle gets its different physical quantities from an interpolation of its neighboring particles quantities. Specific kernels are used to obtain a smooth interpolation of values through the weighted contribution of neighbors according to their relative distance. From these quantities, the Navier-Stokes equations are used to compute the interaction forces between particles, and thus a new state for each particle. SPH simulations of fluids have been shown to be stable and fast enough to achieve interactive rates [6] [7].

Simulating physically-based fluids is a computationally expensive task, a critical factor for real-time applications. In order to accelerate the computation of the simulations, research has focused on the implementation of the SPH algorithm on GPU, being naturally parallel. Early attempts [8] [9] were not fully GPU-based. In [10], the entire simulation runs on GPU, by using a texture representation of a grid space subdivision for neighbor computation purposes. In [11], values are accumulated by computing the contribution of each particle *to* its neighbors, instead of computing the contribution *from* its neighbors. This results in faster computations, at the cost of a huge memory consumption, achieving interactive frame rates with up to 130,000 simulated particles.

2.2 SPH Fluid-Rigid Body Interaction

Realistic rigid-fluid interactions are required to create elaborate scenes, especially when dealing with arbitrary-shaped objects. In SPH simulations, collisions between fluid particles and rigid bodies are usually handled through penalty-based methods, by introducing forces opposite to the particle movement at the contact point [12]. These forces are derived from a penalty boundary potential in [13]. Lennard-Jones attraction and repulsion forces are used in [14] together with fluid viscosity forces to model the interaction between fluid particles and deformable polygonal meshes. Another approach to obtain solid-fluid interactions is to represent rigid bodies as a set of particles, as initially introduced in the Eulerian realm [15]. In an unified approach [16] [7], rigid body particles are simulated as fluid particles, using the same pressure and viscosity as between fluid particles. The mass and density values of each solid object particle are set according to the object properties. With the same particle-based representation of rigid-bodies, a more complex and accurate method is proposed in [17], called direct forcing. Control forces and

constraint equations are used in a predictor-corrector fashion, allowing different slip conditions and enforcing non-penetration.

With the exception of [14], the aforementioned algorithms have not been used with a real-time implementation.

2.3 Visual Rendering of SPH Fluids

Many techniques have been proposed to render fluids simulated through SPH. To this aim, an optimized marching cube method is proposed in [18]. However, the result looks blobby, and requires a finite grid representation of the fluid (whereas, compared to Eulerian fluids, an SPH simulation does not). To overcome this problem, the fluid mesh is computed in screen space [19]. To this aim, the front fluid surface depth is rendered from the point of view. This depth buffer is then smoothed using a binomial filter to attenuate the irregular look of the fluid. Finally, a mesh representing the fluid is generated in screen space using a marching square algorithm. This method is efficient but still requires an expensive mesh generation.

Recent methods proposed to render fluids in screen space without generating any meshes. In [20], per-pixel normals of the fluid surface in view space are computed using directly the fluid depth buffer, representing the front fluid surface. Thus, per-pixel lighting as well as environment reflection could be computed by rendering a single full screen quad. A similar method has been proposed in [21], where the fluid depth buffer is smoothed out by minimizing its curvature and the overall fluid appearance is improved using high frequency noise. Moreover, fluid thickness is taken into account in order to simulate light extinction through the medium. With a precomputed SPH fluid simulation, this method achieves real time performances when rendering 64,000 particles.

2.4 6DoF Haptic Rendering

There is a long history of research on 6DoF Haptic interaction with complex objects. It is a challenging task, mainly due to cost of collision detection computations. Attempts made by using convex primitives [22] or NURBS representations [23] greatly limit the number, size and complexity of the virtual interactive objects. Voxel-based haptic rendering [24], improved in [25] and [26], was the first technique to allow efficient 6DoF haptic rendering without shape or complexity constraints. Simple penalty forces are computed between the point-sampled dynamic tool and the voxelized static virtual environment. A sensation-preserving simplification algorithm [27] trades accuracy for speed through multiple levels of detail while preserving the interaction forces. A fast contact point tracking algorithm based on spatialized hierarchies is used in [28] for approximating the collision detection. It is combined with a slower algorithm for exact computations, and allows the use of moderately sized moving objects at haptic rates. In [29], the authors extend the god-object 3DoF technique [30], using continuous collision detection and a constraint-based approach, enforcing non-penetration and improving stiffness perception.

These algorithms are efficient when interacting with rigid and often static obstacles of the virtual environment. 6DoF haptic interaction with deformable models has also been addressed in the past [31] [32]. Fluids, however, have the particularity of being highly deformable and dynamic, with their shape varying at each time-step, often with unconnected components (such as drops). Hence, other haptic feedback algorithms and coupling schemes are required and used in the haptic interaction with fluids.

2.5 Haptic Rendering of Fluids

Recent work has explored the possibilities of haptic interaction with fluids. Hence, different techniques were used to lower the computational demands of the simulation in order to achieve acceptable frame rates for haptic interaction. An example is the pre-computation of the non-linear interaction forces between fluid and rigid bodies, as described in [1]. These forces are added to the linear forces computed in real-time, hence obtaining very high frame rates (around 500Hz), since the main computations are performed offline. However, since force models are ad-hoc, it dramatically limits the number of different interaction scenarios that can be simulated. The method is illustrated through a canoe simulation. Another approach is the extension of a 2D technique to 3D space, as shown in [33]. The fluid surface is modeled as a 15x15 mass-spring network, and a stack of 2D fluid layers is below the surface. The layer deformations produced by the haptic probe are transmitted to the other layers according to their depth and density. Since the algorithm is intended for low-end computers with a small computing power, there is a trade-off between the realism of the simulation and its speed, as well as a limited number of degrees of freedom.

Other techniques are based on existing Eulerian physically-based simulations of fluids. In [3], the authors propose a 6DoF interaction technique with an Eulerian viscous fluid. The two-way rigid-fluid interaction is achieved by discretizing the boundary of the haptic probe into the simulation grid. Forces and torques exerted on the probe are computed by summing the contributions of each edge of the probe boundary. The authors illustrate the technique through a painting application, with the fluid defined locally around the brush, hence allowing the painter to feel the paint, and achieving 40 to 70 frames per second (fps). However, interaction possibilities are limited due to the high computational cost of Eulerian approaches, requiring the use of simple probe shapes and small amounts of fluid. Another haptic interaction implementation with an Eulerian fluid has been recently shown in [2], where

the simulation is accelerated through GPU. The user is allowed to interact with smoke via a spherical probe at 30 to 75 fps. However, the viscosity of the fluid is not taken into account, and the implementation is limited to 3DoF with simple object shapes, considerably reducing the interaction possibilities.

Our fluid haptic rendering technique is based on Smoothed-Particle Hydrodynamics [5] for haptic and fluid simulation [6], and it is the first attempt to bring the SPH model to the haptic fluid realm. Other 6DoF techniques provide high rate algorithms, but are often limited to the interaction with static objects and a reduced number of contact points, which precludes their use with a fluid simulation. The different components of our approach are shown in Figure 1. Using a simple and efficient GPU implementation, with performances close to those reported in [11], our physical simulation reaches framerates suitable for haptic interaction with large volumes of fluid. In order to achieve 6DoF feedback, we improve a unified particle-based approach [7] by bringing it to high speeds. We use this force computation approach combined with a Virtual Coupling mechanism [34] to provide 6DoF haptic coupling. Visual rendering is made possible by adapting a fluid graphic rendering algorithm [21] based on meshless screen space rendering to meet the frame rate requirements of our rendering technique.

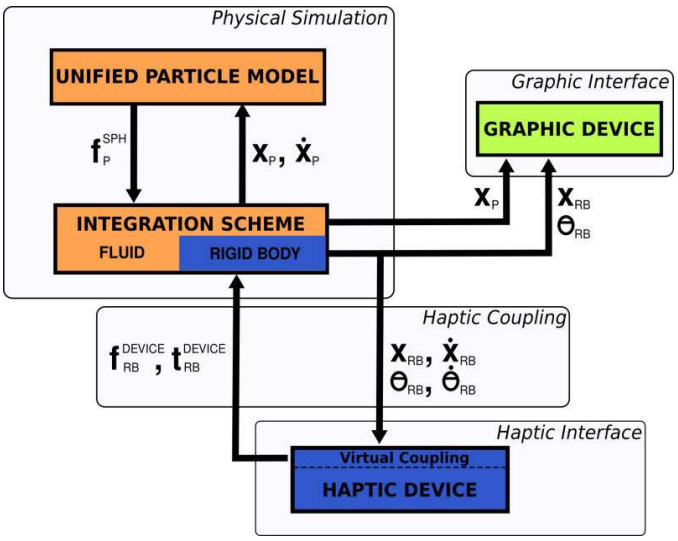


Fig. 1. Simulation overview. The unified particle model computes the different interaction forces between SPH particles P, $\mathbf{f}_P^{SPH}$, including haptic feedback forces. Then, rigid body dynamics are applied to the coupled rigid body RB, taking into account the force $\mathbf{f}_{RB}^{DEVICE}$ and torque $\mathbf{t}_{RB}^{DEVICE}$ generated by the 6DoF haptic device through Virtual Coupling. The new linear and angular velocity ($\dot{X}_{RB}, \dot{\theta}_{RB}$) and position ($X_{RB}, \theta_{RB}$) are sent to the haptic device.

3 Haptic Rendering of Fluids

In this section, we describe our novel haptic rendering technique based on a Smoothed-Particle Hydrodynamics physical model [6]. We first recall the relevant equations involved in the SPH simulation of fluids, for the computation of the physical quantities (density) and the related forces (pressure and viscosity forces). Then, we describe our approach with the computation of SPH haptic forces and a haptic rendering algorithm integrated to the physical simulation, implemented entirely on GPU.

3.1 SPH Simulation of Fluids

An SPH fluid simulation is based on particles carrying different physical properties, such as mass and viscosity, discretizing the fluid volume. These particles have a smoothing radius, a spatial distance defining a neighborhood around them. Physical quantities, such as density and interaction forces, can be computed for each particle through the weighted sum of the relevant properties or quantities of the particles inside its neighborhood. The weight of the contributions of a neighbor particle depends on its distance to the treated particle, and is defined in a function called smoothing kernel.

The smoothed quantity $A(\mathbf{x})$ at any position $\mathbf{x}$ in space is computed through the general formula:

$$A(\mathbf{x}) = \sum_i A_i V_i W(\mathbf{x} - \mathbf{x_i}, h) \quad (1)$$

where A_i is the discrete quantity $A(\mathbf{x_i})$ sampled for neighboring particle i at position $\mathbf{x_i}$, V_i is the volume of i, and W is the smoothing kernel of support h, where particles farther than h are not taken into account.

Hence, the density ρ of particle p can be computed through:

$$\rho_p(\mathbf{x_p}) = \sum_i m_i W(\mathbf{x_p} - \mathbf{x_i}, h) \quad (2)$$

where m_i is the mass of neighboring particle i. We recall $V_i = m_i/\rho_i$, with ρ_i being the density of particle i.

The motion of fluids is driven by the Navier-Stokes equations, with the following formulation (after dropping mass conservation and convective terms, inherent to a Lagrangian simulation) :

$$\rho \frac{D\mathbf{v}}{Dt} = -\nabla P + \rho \mathbf{g} + \mu \nabla^2 \mathbf{v} \quad (3)$$

where P is the pressure, $\mathbf{v}$ the velocity, μ the viscosity coefficient and $\mathbf{g}$ the gravity field. ∇ and ∇^2 are respectively the gradient and Laplacian of the physical quantities. This formulation leads to 3 distinct forces:

$$\rho \mathbf{a} = \mathbf{f}^{pressure} + \mathbf{f}^{gravity} + \mathbf{f}^{viscosity} \quad (4)$$

with $\mathbf{a}$ being the particle acceleration. Pressure forces are computed from pressure quantities, and viscosity forces are computed from velocities:

$$\mathbf{f}_p^{pressure}(\mathbf{x_p}) = -V_p \sum_i V_i \frac{P_p + P_i}{2} \nabla W(\mathbf{x_p} - \mathbf{x_i}, h) \quad (5)$$

$$\mathbf{f}_p^{viscosity}(\mathbf{x_p}) = \mu V_p \sum_i V_i(\mathbf{v}_i - \mathbf{v}_p)\nabla^2 W(\mathbf{x_p} - \mathbf{x_i}, h) \quad (6)$$

The symmetrization of the force computations and the kernel functions follow those proposed in [6]. The pressure computation follows the modified ideal gas state equation proposed in [35]:

$$P = k(\rho - \rho^0) \quad (7)$$

where k is a gas constant that depends on the temperature, and ρ^0 the rest density.

Therefore, the fluid simulation is done in two consecutive steps. First, the density is computed for each particle, and the pressure is derived from it. Second, after computing pressure and viscosity forces, acceleration, velocity and position are deduced through a Leap-Frog integration scheme [36].

Parameters such as the viscosity can be changed to obtain different fluid behaviors, from smoke (no viscosity) to honey (very high viscosity).

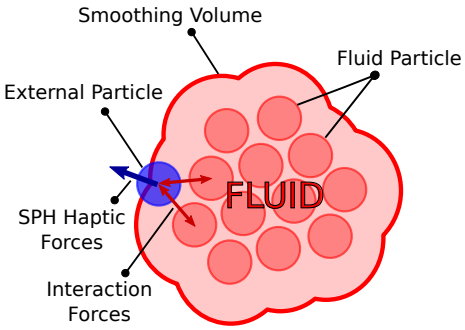


Fig. 2. Smoothing Volume and SPH haptic forces. The Smoothing Volume of a volume of fluid is defined by the sum of the smoothing volumes of each particle belonging to the fluid. When an external particle enters the Smoothing Volume, SPH haptic forces are seamlessly computed between the external particle and the fluid particles whose smoothing volume contains the external particle.

3.2 SPH Haptic Rendering

We introduce a novel haptic rendering algorithm based on Smoothed-Particle Hydrodynamics. Our method allows to compute force-feedback when interacting with fluid through a single fluid particle. Our technique can be classified in the group of distance-field-based haptic rendering techniques, and can therefore be used in simulations where exact surface representation is not required [37]. This is the case for fluids, and particularly SPH fluids, where the exact boundary of the fluid is not explicitly defined: boundary particles are not the boundaries of the fluid, but rather the last points in space that provide information about the fluid boundaries.

3.2.1 Smoothing Volume and SPH Haptic Forces

In the SPH model, forces are computed when a particle enters the smoothing volume of another particle, which is defined as a sphere with the smoothing radius as radius and the particle position as center. The sum of the smoothing volumes of the particles of an entity defines the Smoothing Volume of the entity, behaving like a force field volume around the entity. The Smoothing Volume is illustrated in Figure 2. Any external particle (not belonging to the entity) inside the entity Smoothing Volume will trigger the computation of forces exerted on both the entity and the external particle.

The forces generated when an external particle is inside the Smoothing Volume of an entity are SPH haptic forces, noted $\mathbf{f}^{haptic}$. They repel the external particle from the entity, and the entity from the external particle in a 2-way coupling scenario. Following the computation of quantities in the SPH model, $\mathbf{f}^{haptic}$ has the general form of Equation 1.

The profile of the haptic rendering is closely related to the choice of $\mathbf{f}^{haptic}$. When interacting with fluid through a particle, or in other words when the haptic device is coupled to a particle, an intuitive choice is to treat the coupled particle as a fluid particle. The coupled particle is submitted to the same physical forces as any other fluid particle. Since these forces are then rendered through the haptic device, they provide a direct way to interact with the fluid volume. Hence, our SPH haptic forces, when coupled to a particle, are the sum of pressure and viscosity forces:

$$\mathbf{f}_p^{haptic}(\mathbf{x_p}) = \mathbf{f}_p^{pressure}(\mathbf{x_p}) + \mathbf{f}_p^{viscosity}(\mathbf{x_p}) \quad (8)$$

3.2.2 Haptic Rendering Algorithm

Algorithm 1 shows the different steps of the computation of haptic feedback when interacting with fluid through a single particle coupled to a haptic device. It can be broken down to 4 distinct steps:

- **density computation (lines 1-3)**. A density quantity is computed for each particle, according to the neighboring particles distance and mass.
- **SPH haptic force computation (lines 4-12)**. According to the type of particle (a flag specifying fluid or external), the algorithm computes either fluid interaction forces from section 3.1, or SPH haptic forces from section 3.2.1.
- **fluid particle integration (lines 13-16)**. After adding external forces (such as gravity), a new position and velocity is computed for each fluid particle by integrating forces over the simulation time step.
- **coupled particle integration and haptic coupling (lines 17-22)**. After adding external forces (such as gravity), particles coupled to a haptic device follow a different integration process, taking into account forces from the haptic interface, and sending back their new position and velocity.

3.3 GPU Implementation

The entire physical model is implemented on GPU using the CUDA framework [38]. It is based on Green's imple-

Algorithm 1 Haptic rendering algorithm for a single particle coupling: the different steps of the computation of haptic feedback from the interaction between fluid and external particles are illustrated. Each external particle is coupled to a haptic device.

```
for all particles in fluid do
    compute density (Eq. 2)
end for
for all particles in fluid do
    for all neighboring particles do
        if neighbor is coupled to a haptic device then
            compute SPH haptic forces (Eq. 8)
        else
            compute fluid forces (Eqs. 5 and 6)
        end if
    end for
end for
for all particles in fluid do
    add external forces (gravity)
    integrate (new position and velocity)
end for
for all particles coupled to a haptic device do
    add external forces (gravity)
    add force from haptic interface
    integrate (new position and velocity)
    send new position and velocity to haptic interface
end for
```

mentation of a CUDA particle simulator [39], modified to compute the SPH Navier-Stokes equations.

3.3.1 Optimizations

The increase in performance compared to a CPU implementation comes from the parallelization of the density, force and integration computations. Knowing that an SPH particle only interacts with other particles inside its smoothing radius, the 3D interactive space is subdivided in a regular grid, restricting the search for neighbors to the 26 grid cells around the particle cell and the particle cell itself. Major optimizations regarding memory access and data structures [38] are:

- the sorting of the particles by cell in GPU memory in order to ensure a coalesced read, hence optimizing memory access;
- the binding of the sorted data arrays to textures, and the use of texture lookups to benefit from texture data caching;
- the use of an OpenGL Vertex Array Object for the particle position array, allowing the graphic rendering algorithm to access position data without copying it back to CPU memory.

3.3.2 Update rate

In theory, the graphic rendering frame rate and the update rate of the haptic device could and should be independent. Graphics should be rendered at 24 fps for a comfortable visualization, while the haptic update rate, depending on the computation time of the fluid simulation loop, requires a higher frequency. However, using an all-GPU design to achieve plausible haptic interaction rates requires the use of the GPU for simulation computations *and* for graphic rendering. In current implementations of graphic drivers, graphic rendering is a GPU blocking task, hence the simulation and the graphic rendering on the GPU are not parallelizable. Hence, using a different frequency for graphic update would cause periodic drops in the haptic update rate, introducing artifacts in the haptic rendering. Since every simulation loop has to be rendered, we needed to optimize our graphic rendering algorithm to achieve reasonable haptic update rates, as detailed in section 5. A possible way to dissociate the graphic from the simulation loop would be the use of two GPUs, with one doing simulation computations and the other focusing on graphic rendering. The position data required for the graphic rendering could be copied from the simulation GPU to the other through asynchronous data transfer [38], avoiding a copy lock on the simulation GPU. Moreover, the advent of multi-core GPUs would make dissociation even simpler by removing the need of data transfer between GPUs [40] [41].

4 6DoF Haptic Rendering of Fluids through Rigid Body Interaction

In the previous section, we showed how to generate force feedback when interacting with fluid through a single particle. In this section, we extend our haptic rendering technique to allow the interaction between a rigid body and the fluid, hence producing a 6DoF haptic feedback when coupled to a 6DoF haptic device. We describe our rigid body model, the computation of rigid body dynamics using SPH particles, and the underlying haptic coupling scheme.

4.1 Unified Particle Model

In order to achieve 6DoF haptic interaction with fluids, we required a rigid body model that could fit in the SPH simulation with a minimum impact regarding computation time, and that would allow a flexible coupling mechanism. We propose a real-time approach based on [7] and improved for haptic interaction: a unified particle model allowing the seamless real-time interaction between fluid and arbitrary-shaped rigid bodies.

Rigid bodies can be simply and efficiently modeled with the same SPH particles used in the fluid simulation. This allows to use the SPH model for the computation of forces between fluid and rigid body particles, removing the need of additional collision detection algorithms. Moreover, since interactions are computed between particles, the overall shape of the rigid body is not important. Hence, the unified particle model allows the seamless use of arbitrary-shaped rigid bodies, including concave objects.

Rigid body polygonal meshes are sampled into a set of particles covering the surface of the mesh. Figure 3 shows a bowl mesh and its corresponding particle sampling. Several mesh sampling techniques exist in the literature. We highlight the work presented in [42], for an offline simulation, where the particle sampling is achieved in a preprocessing step using a distance field, and the random placement of particles inside the distance field. Particles are then subjected to an attraction constraint to the zero-isosurface and a repulsion constraint against other particles, achieving an efficient sampling of the surface when reaching a convergence criteria. The survey and the improvement of other techniques are, however, beyond the scope of this paper.

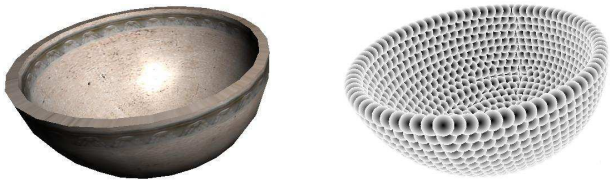


Fig. 3. Rigid body particle sampling. The mesh bowl (left) is converted into a set of particles (right).

The SPH fluid simulation and the haptic rendering algorithm are adapted to allow the simulation of rigid bodies and their interaction with the fluid. These changes are:

- rigid body particles interact with fluid particles through the SPH haptic forces $\mathbf{f}^{haptic}$;
- rigid body particles are given constant densities, since for rigid bodies these quantities are constant throughout the simulation;
- when computing the density of fluid particles, rigid body particles are omitted so that fluid densities are only computed with fluid particles;
- rigid bodies follow rigid body dynamics, as described in section 4.2.

The choice of an efficient $\mathbf{f}^{haptic}$ for a rigid body coupling case is not necessarily the same as in single particle coupling. In the previous section, the coupled particle could be treated as a fluid particle, and hence $\mathbf{f}^{haptic}$ was set to follow fluid forces. When dealing with rigid bodies, previous work has used different forces and mechanisms to simulate a solid-fluid interaction, as described in section 2.2. Among these methods are simple penalty forces [12], Lennard-Jones forces [14] and unified fluid-rigid body forces [16] [7].

In a unified, parallel and time-critical framework, unified fluid-rigid body forces fit well for computation time reasons. Using the same interaction forces as in a fluid-fluid case improves the gain of parallel computation, while providing a reasonable amount of control over the forces through density and viscosity values. These values are set by the user per rigid body, hence allowing different behaviors for each rigid body. However, since Navier-Stokes equations are not physically meant for rigid bodies, density and viscosity values cannot be looked up in the literature, but need to be chosen empirically. This is particularly true for viscosity forces, where higher solid-fluid viscosity values are required to achieve the desired viscous and sticky effects.

We used the pressure forces as for fluid particle interaction, and symmetrized the viscosity forces as in [43] to account for the possibility of having different viscosity values:

$$\mathbf{f}_p^{visc}(\mathbf{x_p}) = V_p \sum_i V_i \frac{\mu_i + \mu_j}{2}(\mathbf{v}_i - \mathbf{v}_p)\nabla^2 W(\mathbf{x_p} - \mathbf{x_i}, h) \quad (9)$$

4.2 Rigid Body Dynamics

With rigid bodies, position and velocity values are no longer computed independently for each particle. SPH haptic forces due to the interaction with the fluid are summed and applied at the center of mass cm of the rigid body, as shown in Figure 4.

$$\mathbf{F}_{cm}^{haptic} = \sum_{i_{body}} \mathbf{f}_{i_{body}}^{haptic} \quad (10)$$

Moreover, torques are also applied to the center of mass due to the same SPH haptic forces exerted on the particles of the rigid body:

$$\tau_{cm}^{haptic} = \sum_{i_{body}} \tau_{i_{body}}^{haptic} \quad (11)$$

with:

$$\tau_{i_{body}}^{haptic} = (\mathbf{x}_{i_{body}} - \mathbf{x}_{cm}) \times \mathbf{f}_{i_{body}}^{haptic} \quad (12)$$

Analogously to the linear movement, the angular acceleration α_{cm} of the center of mass can be computed through the use of rotational dynamics:

$$\alpha_{cm} = I^{-1}\tau_{cm}^{haptic} \quad (13)$$

where I is the moment of inertia of the rigid body.

The angular velocity ω_{cm} and position of the center of mass are then computed using a Leap-Frog integration scheme [36]. The angular velocity is reported to each particle of the body according to their position with respect to the center of mass:

$$\mathbf{v}_{i_{body}} = \mathbf{v}_{cm} + \omega_{cm} \times (\mathbf{x}_{i_{body}} - \mathbf{x}_{cm}) \quad (14)$$

4.3 6DoF Haptic Coupling Scheme

When coupling a rigid body to a 6DoF haptic device, we have to take into account the new rigid body dynamics. Algorithm 2 shows the new rigid body dynamics step, replacing the external particle integration step from Algorithm 1 (lines 17-22), and performing the 6DoF haptic coupling.

Figure 1 summarizes the different steps of the 6DoF haptic rendering technique, from the unified SPH simulation to the 6DoF haptic loop. Following our unified

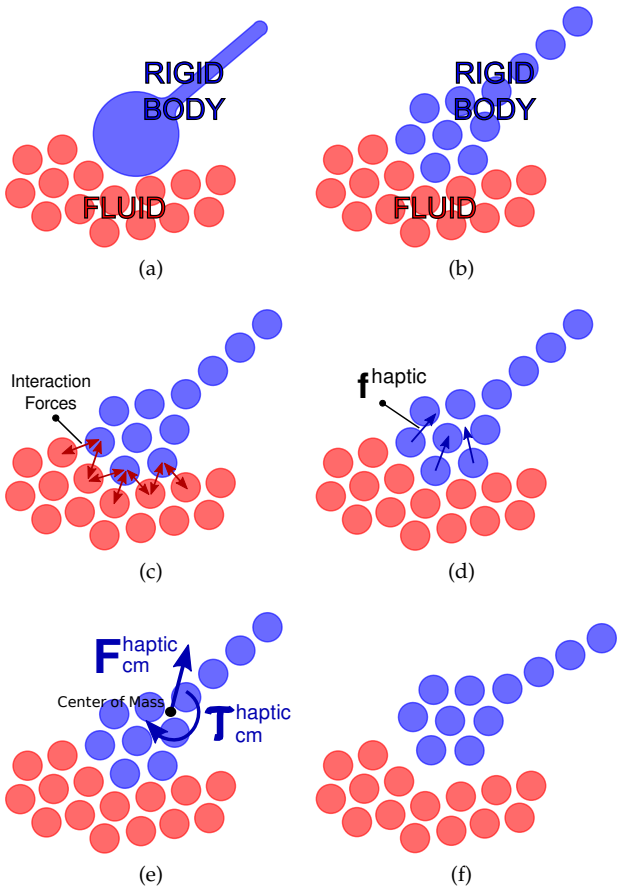


Fig. 4. Illustration of the computation of forces acting on a rigid body. A spoon and volume of fluid (a) are modeled with SPH particles (b). Fluid particles act on the spoon particles (c), generating an SPH haptic force $\mathbf{f}^{haptic}$ per spoon particle (d). These forces are summed resulting in a total force and a total torque applied at the center of mass of the spoon (e). Spoon particles are updated according to the new position and velocity of the spoon(f).

particle model, SPH haptic forces are computed along other forces in the SPH simulation. In the rigid body dynamics step, all forces $\mathbf{f}^{haptic}$ exerted on the coupled rigid body are summed to obtain a total force and a total torque (lines 1-5 of Algorithm 2). The force and torque feedback coming from the haptic device are added to the coupled rigid body (line 7 of Algorithm 2). Then, the new position and velocity are computed by integrating forces over the simulation time step. They are sent to the haptic device, closing the haptic loop in admittance mode [37].

Rigid bodies are also simulated entirely on the GPU. Rigid body particles are stored in the same arrays as fluid particles, since forces are computed for every particle in the simulation. However, they are grouped at the beginning of the arrays in order to ensure a coalesced memory access during rigid body dynamics computations. The sum of SPH haptic forces into a total force following Equation 12 is computed on the GPU by a 2-step tree-based parallel reduction [44], with sequential addressing. In a first CUDA kernel, the force array is partially summed within blocks of maximum thread size, with results stored in global memory. The second CUDA kernel sums the intermediate results into a final force value. Lines 6 to 8 of Algorithm 2 are executed on a single CUDA thread within the previous kernel, since there was no substantial gain in copying the data and executing the instructions on the CPU.

Algorithm 2 6DoF haptic rendering algorithm: rigid body dynamics step and haptic coupling with a 6DoF haptic device.

```
1:  for all rigid bodies do
2:    for all particles in the rigid body do
3:      add force exerted on the particle to total force
4:      add torque from the particle to total torque
5:    end for
6:    add external forces and torques (gravity)
7:    add force and torque from haptic device
8:    integrate (new rigid body position and velocity)
9:    send new position and velocity to haptic device
10:   for all particles in the rigid body do
11:     update particle position and velocity
12:   end for
13: end for
```

This coupling scheme allows the seamless haptic coupling with any rigid body, provided dimensions and masses are compatible with the device span and its maximum efforts. It also allows the computation of N different haptic couplings on the same scene, as well as between N devices and the same rigid body (as holding a bucket with two hands).

4.4 Virtual Coupling

A Virtual Coupling mechanism [34] is introduced between the haptic device and the coupled rigid body, creating a viscoelastic link between them. This mechanism allows the separation of the impedance of the haptic device from the impedance of the virtual environment. It reconciles a high update rate haptic device with a lower rate simulation, leading to an increase of stability [37].

5 Visual Fluid Rendering

As explained in section 3.3.2, the haptic loop needs high frequency updates, which required us to design a visual rendering method with performance over quality in mind. Our visual rendering is close to the one proposed in [21], with reduced computation time. We propose a three step approach: (1) compute per pixel fluid data (front and back fluid volume depth from view), (2) smooth the front surface depth in screen space using a fast bilateral filter and (3) compose the final frame.

5.1 Computing Per-Pixel Fluid Data

In order to render fluids, our method needs the front and back depth of the fluid volume for each pixel of

the view point. To this aim, we render each particle as a sphere. For the sake of performance, we do not render spheres using polygons but as point splats [21]. The back depth of the fluid volume is obtained using a reversed depth test. In order to obtain a connected fluid surface, point splats radius is 1.2 times larger than simulated particles radius. To accelerate this step, this buffer can be computed at a lower resolution than the screen. In our case, we used half the screen resolution.

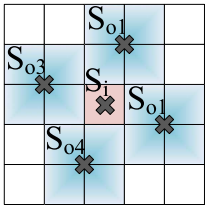


Fig. 5. The bilateral sampling kernel used to smooth the front surface of the fluid.

Using the raw depth values for the front fluid surface makes it appear blobby. To smooth out depth values, we could have used a bilateral Gaussian filter but this is a non separable blur filter. The algorithm proposed in [21] is very efficient but requires too many iterations to be effective. Thus, we propose to use a fast bilateral blur filter taking advantage of the hardware bilinear filtering (Figure 5). First, the depth is sampled for the inner pixel S_i. Then, four outer depths S_o are sampled. Thanks to hardware bilinear filtering, each of these outer samples represents the average of four depth values. The bilateral filtering is achieved by weighting outer depth samples according to the depth difference to the depth sampled at S_i. An outer sample is used only if the four averaged depths are from the fluid surface. We give a weight of 2.0 for S_i and 4.0 for S_o samples. This method allows us to average 17 depth values using only 5 texture samples. This bilateral filter is of lower quality than the filter proposed in [21] and can result in plateaus of equal depth. However, we can obtain a smooth fluid surface in few iterations. After preliminary testings we set the number of iterations to four.

5.2 Fluid Compositing

The fluid volume is finally composed with the virtual scene during a single full-screen pass using the virtual scene color S_c and linear depth S_d buffers, as well as the fluid data buffer. As in [21], the distortion of the scene color Sc perceived through the fluid is approximated as a texture look-up in screen-space with a displacement vector corresponding to the normal of the front fluid interface scaled by the fluid depth and refraction intensity. The final pixel color $col(p)$ is given by:

$$col(p) = \texttt{lerp}(\mathbf{F_c}, \mathbf{S_c}(\mathbf{p} + \beta \times T(p) \times \mathbf{n}.xy), \exp^{-T(p)*\mathbf{F_e}}) + \mathbf{k_s}(\mathbf{n} \bullet \mathbf{h})^{\alpha} \quad (15)$$

Fluid thickness $T(p)$ is first computed for each pixel p using $T(p) = \min(F_bD(p), S_d(p)) - \min(F_fD(p), S_d(p))$ where F_fD and F_bD are respectively the front and back depth of the fluid volume. In the case $T(p) = 0$, we simply copy the scene color. Otherwise, if $T(p) > 0$, we compute the final pixel color using Equation 15 where $\mathbf{F_c}$ is the fluid color, $\mathbf{F_e}$ is the wavelength-dependent color extinction coefficients of the fluid, β represents the amount of background color distortion simulating a refractive index, $\mathbf{h}$ the Blinn half-angle vector, $\mathbf{k_s}$ the specular color and α the specular exponent. The normal n is computed using finite difference on the front fluid surface depth [21].

The proposed optimizations of the technique described in [21] makes the approach suitable for real-time applications. These optimizations can thus be considered as a trade-off between performance and quality.

6 Evaluation

In this section, we evaluate our approach in terms of computation time and haptic rendering (forces and torques). The quality of our real-time visual rendering is also discussed.

6.1 Hardware Setup

The evaluation scenarios were carried out using two Virtuose 6DoF force-feedback devices from Haption (Soulge-sur-Ouette, France). Figure 12 shows both devices in use. The simulations were run on a laptop computer with a Core 2 Extreme X7900 processor at 2.8GHz, 4GB of RAM memory, and a Nvidia Quadro FX 3600M GPU with 512MB of graphic memory.

6.2 Computation Time

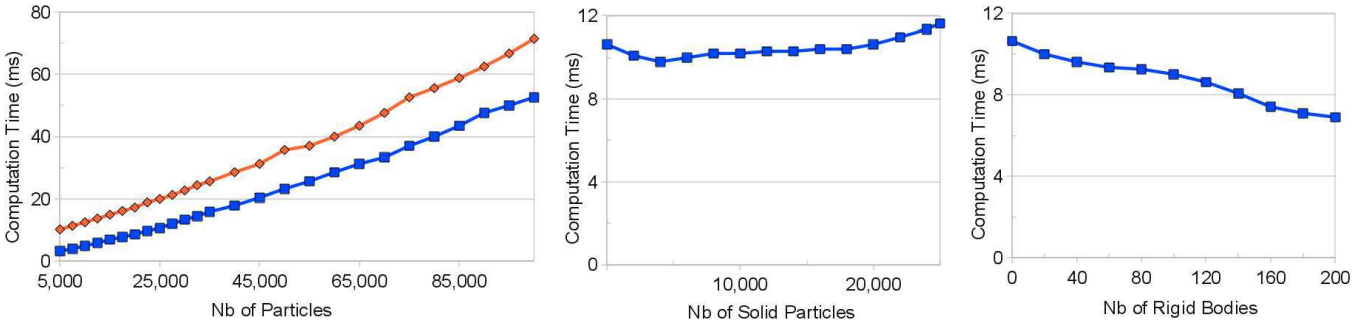
6.2.1 Fluid Simulation Performance

Ideally, for rigid body simulations, update rates should be close to 1000Hz [37]. However, forces due to fluids do not change rapidly. Hence, in order to achieve a smooth and stable haptic rendering, the update rate can be considerably lowered, and rates close to 70Hz have been reported to be satisfactory [3] [2].

Figure 6a shows the performance of our fluid simulation with respect to the number of fluid particles. We measured the computation time of the simulation of a 1x1m fluid pool with and without the graphic rendering.

Under 50ms (20 Hz), we consider that the simulation is not adequate for real time applications. With around 2,300 particles, the simulation and the graphic rendering take the same amount of time, while with 100,000 particles the simulation is around 4 times slower. Hence, the bottleneck of our simulation, when increasing the number of particles, is the physical simulation, while the graphic rendering remains efficient with a much lower drop in frame rate.

At 70 Hz, our implementation can thus simulate 32,000 fluid particles with the aforementioned hardware configuration, thus allowing the interaction with a considerable volume of fluid.



(a) Performance of the fluid simulation (in blue, squares) and the fluid simulation including graphic rendering (in red, diamonds) with respect to the number of fluid particles

(b) Performance of the simulation of a pool of fluid and a rigid body sphere with respect to the size of the rigid body. The number of total particles is constant.

(c) Performance of the simulation of rigid bodies with constant total number of particles with respect to the number of rigid bodies.

Fig. 6. Performance evaluation of our simulation algorithms

6.2.2 *Unified Particle Model Performance*

Figure 6b shows the computation time required to simulate a pool of fluid and a rigid body sphere with respect to the size of the rigid body. As the size of the rigid body increases, the size of the pool decreases to maintain a constant number of particles. The total number of particles is set to 25,000 since it allows a good frame rate for haptic rendering while being enough to simulate detailed objects and large amounts of fluid.

Up to 5,000 rigid body particles, the computation time decreases, due to the skipping of the density computation for rigid body particles. Beyond 5,000 particles, the computation time starts increasing again, since updating the rigid body particle positions from the newly computed values for the center of mass becomes more time consuming. In the GPU, each rigid body is treated in a single CUDA block in order to share the common rigid body values. When dealing with several rigid bodies of smaller size, which is more common than a scene with a single rigid body, the simulation becomes much more efficient, as shown in Figure 6c. From the single 25,000 particles rigid body sphere, we made several non-contacting rigid bodies spheres of equal size. When increasing the number of rigid bodies, with a constant number of total particles, we can see how computation time decreases due to the use of more CUDA blocks with a lower number of threads, making the particle update more parallel and efficient. Overall, the overhead due to rigid body simulation remains small, since in the worst-case scenario of a single rigid body the simulation takes less than 10% more time to compute than a pool of fluid with the same number of particles.

6.3 Graphic Rendering

Figure 7 shows the result of our technique (left) and of the technique proposed in [21] (right) for a fluid volume made of 32768 particles, rendered at a resolution of 1024x768. The method described in [21] requires 60 passes (GPU computation time of 45ms for a final framerate of 20fps) to result in a smooth fluid surface. However, our method only requires 4 passes

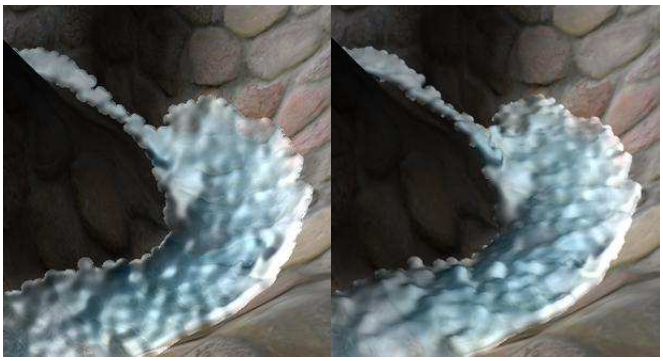


Fig. 7. Comparison of our graphic rendering method (left) with the original version [21] (right) for a volume of fluid.

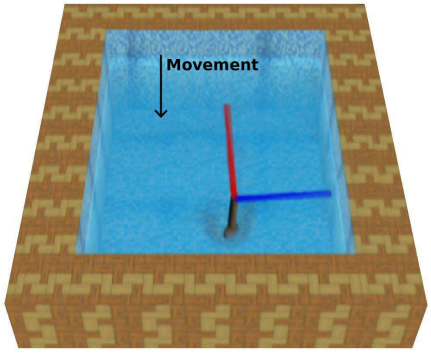
(GPU computation time of 8ms for a final framerate of 90fps). Our method being designed primarily for speed over accuracy, some artifacts are visible at the edges of the volume whereas, using the original approach, the edges are preserved. Moreover, our fluid rendering exhibits plateaus of equal depth producing a surface that seems more flat. However, our method is faster than the original method [21] and accurate enough for our use-cases.

6.4 Example Scenarios

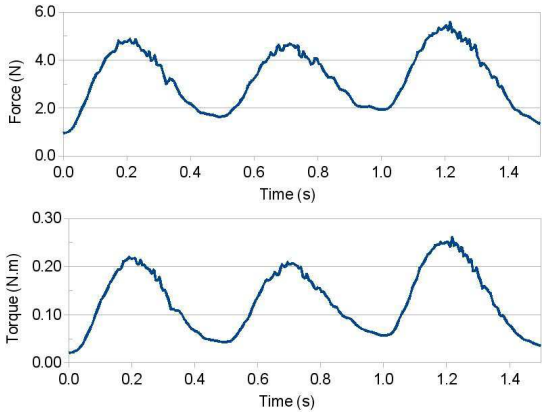
Throughout the following examples, we highlight some interesting interaction scenarios to illustrate the various possibilities of our approach. We recorded the efforts exerted on the haptic device, and provide plots with the magnitude of the efforts with respect to time, in order to have a visual feedback on the haptic rendering.

6.4.1 *6DoF Interaction*

This scenario was designed to highlight the 6DoF capabilities of our technique. The possibility to perceive torques is a main feature of our work, since these torques were very limited or not possible at all in prior existing work. The user exerts a forward and backward movement with a rigid body spoon on a pool of low



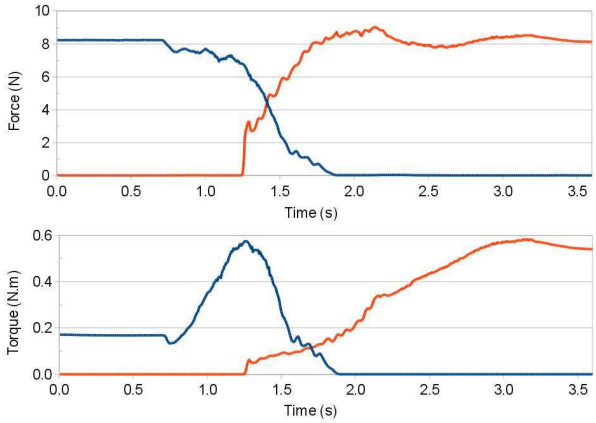
(a) Force (in red) and torque (in blue) exerted on the spoon. The force is opposite to the spoon movement. The torque shows a clock-wise effort in accordance to the spoon movement.



(b) Plot of the force (top) and the torque (bottom) generated by the forward, backward and forward movement of the spoon on the fluid.

Fig. 8. 6DoF haptic interaction: scenario illustrating 6DoF interaction possibilities, with a user holding a virtual spoon through a haptic device and using it to stir a pool of fluid. The user feels the force exerted on the spoon, as well as the corresponding torque, which plays a major role in the haptic perception of the stirring movement.

(a) Fluid is being poured from the bowl into the pan, making the fluid mass shift between both rigid body containers. The containers are hand held by the user through two 6DoF haptic devices.



(b) Plot of the force (top) and the torque (bottom) of the bowl (in blue/dark) and the pan (in red/light) during the pouring motion of some fluid. The fluid mass shift is clearly visible in the force plot, while the torque plot illustrates how the bowl is first tilted (torque peak) and then emptied (torque decrease).

Fig. 9. Container interaction: scenario illustrating the interaction with fluid through rigid body containers. A bowl and a pan, each coupled to a 6DoF haptic device, are used as containers. The fluid held by the bowl is poured into the pan.

viscosity fluid. Figure 8a shows the forces and torques involved in the stirring of the fluid with the spoon. We can notice the force (in red) going opposite to the back-to-front movement of the spoon. The torque (in blue) is generated through a rotation around an axis orthogonal to the linear movement. These visual cues match the plotting of the force and the torque over the duration of the movement (3 forward and backward movements) in Figure 8b.

6.4.2 *Container Interaction*

In this scenario, we illustrate the use of rigid bodies as containers to interact with the fluid. We used two 6DoF Virtuose haptic devices. The first device (left hand) is coupled to a bowl, while the second device (right hand) is coupled to a pan. The fluid is poured from the bowl into the pan, as shown in Figure 9a. Figure 9b shows the torques involved in the pouring movement of the bowl, and how weight shifts from the bowl to the pan.

6.4.3 *Variable Viscosity*

An important feature of our approach is the possibility to interact with fluids of different viscosity, and hence "feel" these different viscosities through the haptic devices. We designed a scenario to capture the forces exerted on a rigid body container being spun around its vertical axis. The container is full of fluid, and initially at rest. The glass is spun with constant speed, kept rotating, and then stopped suddenly. The fluid has a viscosity of 2 Pa.s, while four different viscosities are used for

the rigid body container: 2, 10, 20, 40 Pa.s. The increase in viscosity leads to higher forces exerted on the haptic device. Figure 10 shows the torques generated in each case, clearly showing each of the 4 phases (rest, start, spin, stop) of the movement and each of the 4 viscosities (from low to high: blue, red, yellow, green).

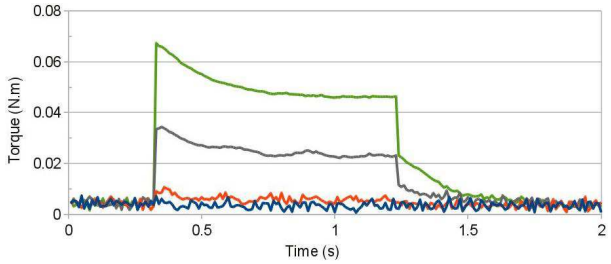


Fig. 10. Plot of the torques exerted on a rigid body container full of fluid (2 Pa.s viscosity) during a spinning movement. The container has different viscosities: 2 Pa.s (blue), 10 Pa.s (red), 20 Pa.s (yellow), 40 Pa.s (green).

6.4.4 Bimanual coupling on the same rigid body

Our haptic coupling and rendering mechanism allows the 6DoF haptic coupling between one or multiple haptic device and any rigid body, as well as between N devices and the same rigid body. This last feature of our approach is illustrated through a bucket full of low viscosity fluid held with two hands. The virtual bucket is coupled to the two 6DoF Virtuose haptic devices, as holding the bucket by each of its handles. The bucket is tilted to the left, to the right, and then to the left again, each time spilling some fluid. Figure 11 shows the force and torque plot for each haptic device during the tilting movements of the bucket.

6.5 A complete use-case

To showcase the main features of our approach, we developed a complete use-case: a virtual cooking simulator. It is a training and entertaining 2-handed interactive application that simulates the preparation process of a crepe. The user holds two virtual objects, a bowl and a pan, through two 6DoF Virtuose haptic devices from Haption, as shown in Figure 12. The simulation guides the user through all the steps required to prepare a crepe (Figure 13): from the stirring and pouring of the batter to the spreading of different toppings on top of the crepe. By preparing virtual crepes, users can experience 6DoF haptic interaction with fluids of varying viscosity. It is one of the many promising applications that can be designed around 6DoF haptic interaction with fluids.

7 Discussion

Table 1 compares our technique in terms of features with previous techniques allowing real-time haptic interaction with fluids, highlighting the important aspects of an interaction scenario. The following discussion is structured around these different features.

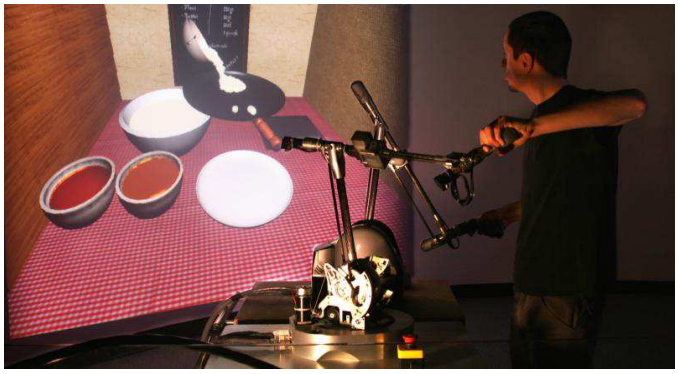


Fig. 12. The virtual crepe preparation simulator: the user pours some batter (a high viscosity fluid) from the bowl and into the pan. He feels the forces and torques from the pouring movement, as well as the weight shifting between his hands as the batter goes from the bowl to the pan.

A main feature of our work is the use of a Lagrangian SPH-based technique for haptic interaction with fluids, with many advantages in a real-time application: it is fast to compute (inherent mass conservation and no advection), it provides freedom regarding the scene (scalability, not bounded to a grid), and provides an intuitive way for haptic coupling. In the SPH model, the inherent smoothing of the SPH haptic forces, and the lack of discrete contact points, ensure that there are no discontinuities in the magnitudes of the resulting forces and torques, providing a stable and smooth haptic feedback.

Regarding the performance of our algorithms, we could run our scenarios at higher frame rates than the other techniques surveyed in section 2.5, even with more complex scenes. The only exception is the work presented in [1], which is not entirely computed in real-time, limiting the interaction possibilities. The higher performance is achieved through the use of the SPH model and our efficient GPU implementation, and performing better than [9] and [10] (12 and 1.7 times faster, respectively), and close to [11] (1.2 times slower), which focus exclusively on the optimization of a particle-based simulation of fluids. The different force and torque plots of section 6.4 provide a visual representation of the force feedback. The haptic forces and torques exerted on the device match the scenarios and the different interactions that are visually perceived, showing that the haptic rendering is faithful and realistic.

The disadvantages of our technique are common to distance-field-based haptic rendering techniques. The use of a smoothing volume makes the interaction approximate, hence focusing the use of the technique on applications where exact surface representation is not required. As mentioned earlier, this approach is very well suited for the haptic interaction with fluids due to their approximate boundary.

Other approaches for solid-fluid interaction, such as direct forcing with constraint equations, provide

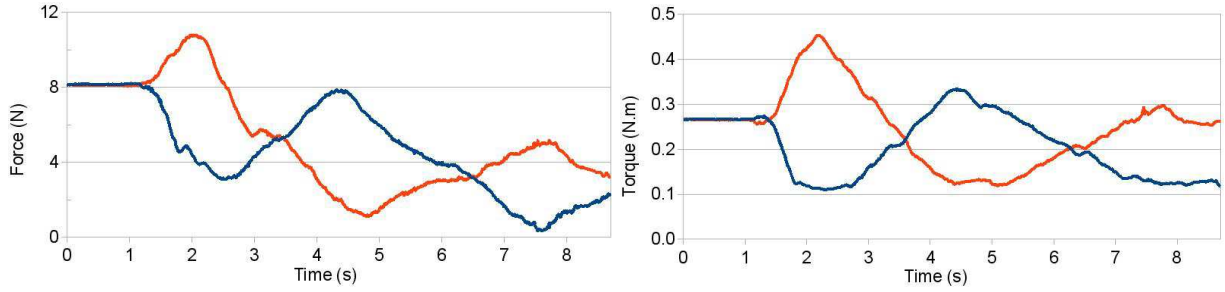


Fig. 11. Bimanual coupling on the same rigid body. Plot of the force (left) and the torque (right) generated by fluid inside a bucket coupled to two 6DoF haptic devices (left hand in red/light, right hand in blue/dark). The peaks and valleys correspond to the tilting movements, increasing the force and the torque on the side to which the bucket is tilted. Forces and torques progressively decrease as fluid is spilled.

Method	Physical Model	DOF	Real-time Computations	Performance[1]	GPU Acceleration	Arbit.-Shaped Rigid Bodies	Variable Viscosity
[33]	Eulerian	3	✓	30 Hz			
[1]	-	6		500 Hz			
[3]	Eulerian	6	✓	40 - 70 Hz			
[2]	Eulerian	3	✓	30 - 75 Hz	✓		
Our method	Lagrangian	6	✓	60 - 120 Hz	✓	✓	✓

TABLE 1
Comparison of our approach with previous work on real-time haptic interaction with fluids.

more control on boundary conditions and enforce non-penetration for colliding particles. However, there is a computational cost with up to 3 collision detection steps for 2-way coupling scenarios, and some restrictions on the number of interacting bodies. Regarding the haptic coupling, the use of a Virtual Coupling mechanism improves the stability of the haptic feedback, but inevitably affects the transparency of the haptic coupling due to the introduction of a viscoelastic link between the haptic device and the rigid body. Further investigation will assess the impact of using a Virtual Coupling mechanism on the perception of fluid haptic feedback.

Although we did not run into stability issues, using explicit integration schemes means that unconditional stability is not guaranteed. Moreover, if a particle moves more than a kernel-size in one time step it could pass through other particles without collision, no matter the solid-fluid interaction algorithm. However, this can only happen with very fast particles (above 10 $m.s^{-1}$ with our current simulation parameters), one order of magnitude faster than the highest speed of a fluid particle in our simulation scenarios. Further investigation is required to assess these issues, but they can only be solved with expensive implicit integration schemes, which would make computations overly expensive given the high number of particles and contact points.

Regarding its benefits, our haptic interaction technique provides real-time 6DoF force-feedback, which was only possible in [1] with pre-computed ad-hoc forces. Moreover, our unified particle model allows the seamless use of complex rigid bodies of arbitrary shape for the 6DoF haptic interaction with the fluid, including concave rigid body containers that can hold the fluid or parts of it. It allows an entire range of scenarios and applications, such as using complex utensils like a mixer to stir a soup, or odd shaped probes to explore the cavities of human body vessels. Other advantages, not illustrated in this work but inherent to the SPH model, allow the design of rich and compelling scenes: an entirely dynamic virtual world and topology-changing rigid bodies without impacting the computation time.

We believe many applications could be designed based on our approach. Such applications span from the medical field (organic fluids like blood) to industrial scenarios (painting, manipulating dangerous fluids) and gaming or entertainment simulations (water and mud in natural scenes, water sports).

8 Conclusion

In this work, we proposed a novel approach for 6DoF haptic interaction with viscous fluids based on a Smoothed-Particle Hydrodynamics physical model. It allows real-time 6DoF haptic interaction with fluids of variable viscosity through arbitrary shaped rigid bodies. Through a novel haptic rendering technique, we compute SPH haptic forces to produce a smooth haptic interaction. Thanks to a unified particle model, rigid bodies can interact with fluids and provide 6DoF haptic feedback. We designed different example scenarios to illustrate and evaluate some of the interaction possibilities

1. Regarding performance, fair comparison is difficult, since existing haptic fluid interaction techniques are based on different physical models and were computed on different hardware. We provide these performance values for information purposes only.

Fig. 13. New interaction possibilities: a bowl coupled to the 6DoF haptic device is used to stir the batter (left), and the same bowl pours maple syrup on the crepe (right).

offered by our technique. The evaluation of the technique showed the efficiency of the rendering algorithm in scenes rendered at 70Hz with up to 32,000 particles.

This novel approach proves that fluids are now readily available for complex 6DoF haptic interactions, and opens an entirely new horizon of applications and interaction techniques in Virtual Reality.

Acknowledgments

This work was supported by the European Community under FP7 FET-Open grant agreement n°222107 NIW - Natural Interactive Walking.

References

[1] Y. Dobashi, M. Sato, S. Hasegawa, T. Yamamoto, M. Kato, and T. Nishita, "A fluid resistance map method for real-time haptic interaction with fluids," *Proc. ACM Symp. Virtual Reality Software and Technology*, 2006.

[2] M. Yang, J. Lu, Z. Zhou, A. Safonova, and K. Kuchenbecker, "A GPU-Based approach for Real-Time haptic rendering of 3D fluids," *Proc. ACM SIGGRAPH Asia Sketches*, Dec. 2009.

[3] W. Baxter and M. C. Lin, "Haptic interaction with fluid media," *Proc. Graphics Interface*, pp. 81–88, 2004.

[4] R. Bridson and M. Muller-Fischer, "Fluid simulation" *ACM SIGGRAPH courses*, 2007.

[5] J. J. Monaghan, "Smoothed particle hydrodynamics," *Annual Review of Astronomy and Astrophysics*, vol. 30, no. 1, pp. 543–574, Sep. 1992.

[6] M. Muller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation*, 2003.

[7] B. Solenthaler, J. Schlfli, and R. Pajarola, "A unified particle model for fluid-solid interactions," *Comput. Animat. Virtual Worlds*, vol. 18, no. 1, pp. 69–82, 2007.

[8] T. Amada, M. Imura, Y. Yasumuro, Y. Manabe, and K. Chihara, "Particle-Based fluid simulation on GPU," *Proc. GP2 Workshop*, 2004.

[9] K. Hegeman, N. Carr, and G. Miller, "Particle-Based fluid simulation on the GPU," *LNCS*, pp. 228–235, 2006.

[10] T. Harada, S. Koshizuka, and Y. Kawaguchi, "Smoothed particle hydrodynamics on GPUs," in *Proc. Computer Graphics Int'l*, 2007.

[11] Y. Zhang, B. Solenthaler, and R. Pajarola, "Adaptive sampling and rendering of fluids on the GPU," *Proc. Symp. Point-Based Graphics*, Aug. 2008.

[12] J. J. Monaghan, "Smoothed particle hydrodynamics," *Reports on Progress in Physics*, vol. 68, no. 8, pp. 1703–1759, 2005.

[13] J. Bonet and S. Kulasegaram, "A simplified approach to enhance the performance of smooth particle hydrodynamics methods," *Applied Mathematics and Computation*, vol. 126, no. 2-3, pp. 133–155, Mar. 2002.

[14] M. Muller, S. Schirm, M. Teschner, B. Heidelberger, and M. Gross, "Interaction of fluids with deformable solids," *Comput. Animat. Virtual Worlds*, vol. 15, no. 3-4, pp. 159–171, 2004.

[15] M. Carlson, P. J. Mucha, and G. Turk, "Rigid fluid: animating the interplay between rigid bodies and fluid," *ACM SIGGRAPH Papers*, 2004.

[16] R. Keiser, B. Adams, P. Dutre, L. Guibas, and M. Pauly, "Multiresolution Particle-Based fluids," ETH Zurich, Tech. Rep., 2006.

[17] M. Becker, H. Tessendorf, and M. Teschner, "Direct forcing for lagrangian Rigid-Fluid coupling," *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 3, pp. 493–503, 2009.

[18] I. D. Rosenberg and K. Birdwell, "Real-time particle isosurface extraction", *Proc. ACM Symp. on Interactive 3D Graphics and Games*, 2008.

[19] M. Muller, S. Schirm, and S. Duthaler, "Screen space meshes," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation*, 2007.

[20] H. Cords and O. Staadt, "Instant liquids," *Poster Proc. ACM Siggraph/Eurographics Symp. Computer Animation*, 2008.

[21] W. J. van der Laan, S. Green, and M. Sainz, "Screen space fluid rendering with curvature flow," *Proc. ACM Symp. Interactive 3D Graphics and Games*, 2009.

[22] A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, and D. Manocha, "Six degree-of-freedom haptic display of polygonal models," *Proc. IEEE Conf. Visualization*, 2000.

[23] D. D. Nelson and E. Cohen, "Optimization-Based virtual surface contact manipulation at force control rates," *Proc. IEEE Virtual Reality*, 2000.

[24] W. A. McNeely, K. D. Puterbaugh, and J. J. Troy, "Six degree-of-freedom haptic rendering using voxel sampling," *ACM SIGGRAPH Papers*, 1999.

[25] M. Wan and W. A. McNeely, "Quasi-Static approximation for 6 Degrees-of-Freedom haptic rendering," *Proc. IEEE Visualization*, 2003.

[26] W. A. McNeely, K. D. Puterbaugh, and J. J. Troy, "Advances in voxel-based 6-DOF haptic rendering," *ACM SIGGRAPH Courses*, 2005.

[27] M. A. Otaduy and M. C. Lin, "Sensation preserving simplification for haptic rendering," *ACM SIGGRAPH Papers*, 2003.

[28] D. E. Johnson, P. Willemsen, and E. Cohen, "Six Degree-of-Freedom haptic rendering using spatialized normal cone search," *IEEE Trans. Visualization and Computer Graphics*, vol. 11, no. 6, pp. 661–670, 2005.

[29] M. Ortega, S. Redon, and S. Coquillart, "A six Degree-of-Freedom God-Object method for haptic display of rigid bodies with surface properties," *IEEE Trans. Visualization and Computer Graphics*, vol. 13, no. 3, pp. 458–469, 2007.

[30] C. Zilles and J. Salisbury, "A constraint-based god-object method for haptic display," *Proc. IEEE/RSJ Intelligent Robots and Systems*, 1995.

[31] S. Cotin, H. Delingette, and N. Ayache, "Real-Time elastic deformations of soft tissues for surgery simulation," *IEEE Trans. Visualization and Computer Graphics*, vol. 5, no. 1, pp. 62–73, 1999.

[32] J. Barbic and D. James, "Time-critical distributed contact for 6-DoF haptic rendering of adaptively sampled reduced deformable models," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation*, 2007.

[33] J. Mora and W. Lee, "Real-Time 3D fluid interaction with a haptic user interface," *Proc. IEEE 3D User Interfaces*, 2008.

[34] J. Colgate, M. Stanley, and J. Brown, "Issues in the haptic display of tool use," *Proc. IEEE/RSJ Intelligent Robots and Systems*, 1995.

[35] M. Desbrun and M. Cani, "Smoothed particles: a new paradigm for animating highly deformable bodies," *Proc. Eurographics Workshop on Computer Animation and Simulation*, 1996.

[36] C. Pozrikidis, *Numerical Computation in Science and Engineering*, illustrated edition ed, Oxford University Press, USA, Apr. 1998.

[37] M. C. Lin and M. Otaduy, Eds., *Haptic Rendering: Foundations, Algorithms and Applications*, A. K. Peters, Jul. 2008.

[38] NVIDIA, "NVIDIA CUDA programming guide 2.0," 2008.

[39] S. Green, "CUDA particles," Jun. 2008.

[40] C. R. Johns and D. A. Brokenshire, "Introduction to the cell broadband engine architecture,", *IBM J. Res. Dev.*, vol. 51, no. 5, pp. 503–519, 2007.

[41] L. Seiler, D. Carmean, E. Sprangle, T. Forsyth, M. Abrash, P. Dubey, S. Junkins, A. Lake, J. Sugerman, R. Cavin, R. Espasa, E. Grochowski, T. Juan, and P. Hanrahan, "Larrabee: a many-core

x86 architecture for visual computing," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–15, 2008.
[42] N. Bell, Y. Yu, and P. J. Mucha, "Particle-based simulation of granular materials," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation*, 2005.
[43] M. Muller, B. Solenthaler, R. Keiser, and M. Gross, "Particle-based fluid-fluid interaction," *Proc. ACM SIGGRAPH/Eurographics Symp. on Computer Animation*, 2005.
[44] M. Harris, "Optimizing parallel reduction in CUDA," Jun. 2008.

Gabriel Cirio is a PhD student at the French National Research Institute for Computer Science and Control (INRIA) in Rennes, France. His main research interests include virtual reality, physically-based modeling and haptic rendering.

Maud Marchal is an Associate Professor in the Computer Science Department at INSA (Engineer school) in Rennes, France. She received her PhD in Computer Science from the University Joseph Fourier in Grenoble, France in 2006. Her main research interests include physical simulation, biomechanics, haptic interfaces, 3D interaction and virtual reality.

Sébastien Hillaire is a PhD student at the French National Research Institute for Computer Science and Control (INRIA) and France Telecom Orange Labs in Rennes, France. His main research interests include virtual reality, gaze-tracking, visual attention models and real-time high-quality rendering on GPUs.

Anatole Lécuyer is a Research Scientist at the French National Research Institute for Computer Science and Control (INRIA) in Rennes, France. His main research interests include virtual reality, 3D interaction, haptic interfaces, and brain-computer interfaces. He promotes a perception-based approach for the design of virtual environments.