A Fluid Resistance Map Method for Real-time Haptic Interaction with Fluids

Yoshinori Dobashi
Hokkaido University
Kita-ku, Kita 14, Nishi 9
Sapporo 060-0814, Japan
+81.11.706.6530
doba@nis-ei.eng.hokudai.ac.jp

Shoichi Hasegawa
Tokyo Institute of Technology
4259 Nagatuta
Yokohama 226-8503, Japan
+81.45.924.5050
hasegawa@pi.titech.ac.jp

Mitsuaki Kato
The University of Tokyo
7-3-1, Hongo, Bunkyo-ku
Tokyo 113-8656, Japan
+81.4.7136.3942
kato@fel.t.u-tokyo.ac.jp

Makoto Sato
Tokyo Institute of Technology
4259 Nagatuta
Yokohama 226-8503, Japan
+81.45.924.5050
msato@pi.titech.ac.jp

Tsuyoshi Yamamoto
Hokkaido University
Kita-ku, Kita 14, Nishi 9
Sapporo 060-0814, Japan
+81.11.706.6530
yamamoto@ist.hokudai.ac.jp

Tomoyuki Nishita
The University of Tokyo
5-1-5, Kashiwanoha
Kashiwa 227-8561, Japan
+81.4.7136.3942
nis@is.s.u-tokyo.ac.jp

ABSTRACT

Haptic interfaces enable us to interact with a virtual world using our sense of touch. This paper presents a method for realizing haptic interaction with water. Our method displays forces acting on rigid objects due to water with a high frame rate (500 Hz). To achieve this, we present a fast method for simulating the dynamics of water. We decompose the dynamics into two parts. One is a linear flow expressed by a wave equation used to compute water waves. The other is a more complex and non-linear flow around the object. The fluid forces due to the non-linear flow is precomputed by solving Navier-Stokes equations, and stored in a database, named the *Fluid Resistance Map*. The precomputed non-linear flow and the linear flow are combined to compute the forces due to water.

CR Categories

I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - Animation; Virtual Reality; I.6.5 [Simulation and Modeling]: Model Development – Modeling methodologies; I.6.3 [Simulation and Modeling]: Types of Simulation - Animation.

General Terms

Algorithms.

Keywords

Virtual Reality, Haptics, Fluid Resistance, Simulation, Computational Fluid Dynamics.

1. INTRODUCTION

Intuitive and natural interactions with virtual world are key mechanisms for enhanced perception of the virtual worlds. The haptic rendering is one of such technologies and many methods have been presented [16] [17] [18]. In a typical virtual reality system with a haptic interface, the user controls the movement of a virtual object using the haptic interface, and the system computes forces when the object contacts with other objects. The user can also feel the resulting forces via the haptic interface. This provides the user with a natural interactions with the virtual world. Previous methods have efficiently displayed forces between rigid objects and forces due to soft objects [3] [11] [18] [19]. In this paper, we focus on haptic interactions and rendering with fluids, especially water. In computer graphics, while several publications exist on the visual simulation of fluids such as smoke, water [5] [9] [10], there are few published techniques for the haptic interaction and rendering with fluids. Examples of the forces due to interaction with fluids are aerodynamic resistances on an airplane, hydrodynamic resistances on a ship, water buoyancy, frictional and pressure resistances with the oars of a boat, and a lure used in fishing.

In this paper, we present a method for very rapidly estimating and displaying forces acting on a virtual object due to water. In the field of computational fluid dynamics and ship design engineering, researchers have developed methods to numerically estimate the resistance due to fluids by solving the Navier-Stokes equations [1] [6]. However, their estimation techniques while numerically accurate are prohibitively time consuming. This is especially true for haptic rendering, where a high frame rate is required.

We additionally present a water simulation model for haptic interactions. We decompose dynamics of water into two parts, a linear flow and a non-linear flow. The linear flow is the flow far from the object. The linear flow is well approximated by wave equations derived by linearizing Navier-Stokes equations under the assumption that the flow of water is slow. However, this assumption is not valid for the flow close to the object. The flow around the object is complex and highly non-linear. This non-linear flow has to be computed by directly solving the Navier-Stokes equations, which unfortunately is compute intensive. So we precompute the non-linear flow around the object and create a database, which we call *a fluid resistance map*, or FRM for short.

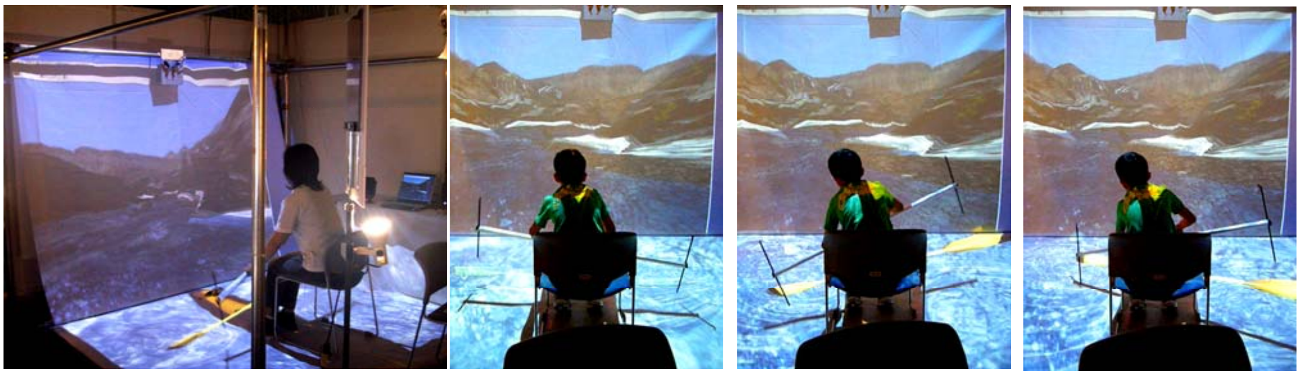


Figure 1: An example application of our method, a virtual canoe simulator. The user can row the virtual canoe using the haptic this system. The forces and the torques on the paddle are calculated in real-time by our method. The user feels the rowing motion via the haptic interface. The motion of water is also calculated by our method and displayed in real-time.

In computing FRM, we separate the effect of the gravity, which creates buoyancy, from the total water force. This eliminates the dependency of FRM on the depth from water surface. Consequently, the presented method computes three types of forces: 1) the buoyancy, 2) the frictional resistance due to the friction between the object surfaces and water, and 3) the pressure resistance due to the movement of the object. In the following, we call the cumulative force of the frictional and the pressure resistances, as fluid resistance. FRM stores the fluid resistance acting on points sampled on the object surface.

Fig. 1 shows an example application of our method, a virtual canoe simulator. The user can control the paddle of the canoe, and feel the forces and torques calculated by our method at 500 Hz. As shown in this example, when we try to move the object in the water, we often feel not only the forces against the movement direction but also the forces perpendicular to the movement direction. This is due to the non-linear flow around the object. That is, vortices generated behind the object cause unexpected fluctuations of the fluid resistance. The fluid resistances are time-varying because of the generation of these vortices. To simulate this, FRM stores the time-varying fluid resistance.

Our method has the following features:

(1) Decomposition of the water dynamics into linear flow and non-linear flow for real-time force display.

(2) Computation and force feedback of the realistic force taking into account the buoyancy, frictional resistance, and pressure resistance.

(3) Simulation of the time-varying fluid resistance.

(4) Rendering of force field due to water by coupling the linear flow (wave equation) and the non-linear flow (FRM).

We demonstrate our method by using a 6 DOF (degrees-of-freedom) haptic interface. The user perceives the torques as well as the forces.

The rest of the paper is organized as follows. First, in Section 2, we discuss the previous work. Next, in Section 3, we present our simulation method for the dynamics of water. In Section 4, we provide an overview of the implementation of the simulation method. In Sections 5 and 6, we provide our method for creating an FRM and the method for the real-time display of the forces. In Section 7, we verify our method by comparing simulation with measurement made with our experimental setup. Furthermore, several examples are demonstrated using haptic interfaces. Next, Section 8 is devoted to discussion of the use of our method. Finally, Section 9 concludes this paper and discusses future work.

2. PREVIOUS WORK

There have been several published methods for haptic interaction and rendering. Mark et al. addressed issues in adding force feedback to graphics systems [16]. Ruspini et al. presented a haptic interface library "HL" [20]. McNeely et al. presented a method for the haptic rendering of complex virtual environments by voxelizing objects [17]. Otaduy and Lin presented a simplification algorithm for faster collision queries with sufficient accuracy for the haptic rendering [18]. Hasegawa and Sato presented a method for real-time rigid body simulations for haptic interactions [11]. These methods realize realistic haptic interactions with virtual worlds. These methods, however, cannot handle forces by objects interacting with fluids. Durbeck et al. displayed forces due to interactions with fluid based on the result of computational fluid analysis [8]. However, they simply displayed the forces according to the flow field and the forces were not physically-based.

To our knowledge, Baxter and Lin tried for the first time to display the forces due to fluids based on physical phenomenon [2]. They succeeded in a real-time haptic display of fluids in two-dimensions by directly solving Navier-Stokes equations. However, computing the fluid force in three-dimension by solving Navier-Stokes equations is time-consuming and therefore the real-time display of the force becomes difficult. In addition, an object controlled by the user is discretized by using grids. The discretized shape changes from frame to frame when the object moves. This causes undesirable noise in the resulting force.

In the field of computational fluid dynamics (CFD), ship design engineering and aeronautics, several researchers have developed methods for computing aerodynamic or hydrodynamic resistances [1] [6]. However, their purposes are to obtain optimal design of an airplane or a ship, that minimizes the drag resistances. These methods require very expensive computational calculations. Therefore, they are not suitable for real-time applications such as interactions with virtual worlds.

3. OUR SIMULATION METHOD OF WATER DYNAMICS

First, we present a method for simulating water dynamics for haptic interactions. Next, we describe a model for the computation of forces between an object and surrounding water.

3.1 Water Dynamics for Haptic Interactions

An object in the water receives forces due to water pressure and frictional drag. We assume that pressure can be decomposed into two components, that is, the pressure due to the movement of the object and the pressure due to gravity. The pressure due to the gravity causes buoyancy. As a result, the presented model treats three types of forces: 1) buoyancy, 2) frictional resistance, and 3) pressure resistance. To display these forces in real-time, we decompose the dynamics of water into linear and non-linear flows as shown in Fig. 2(a). The linear flow is used to simulate the water flow far from the object. The non-linear flow is used for complex flow near the object. The water flow far from the object is generally slow. In this case, the water dynamics is well approximated by linearizing Navier-Stokes equations, resulting in two-dimensional wave equations [13]. On the other hand, the non-linear flow around the object is complex and therefore simulated by solving Navier-Stokes equations. The linear flow can be obtained in real-time since the computational cost for the wave equation is small. However, solving Navier-Stokes equations in real-time to obtain the non-linear flow is difficult. Therefore, non-linear flow is precomputed for various fluid flows and stored as an FRM.

The linear flow (wave equations) and the precomputed non-linear flow (FRM) are combined for real-time haptic interactions. The idea is shown in Fig. 2 (b). First, the shape and velocity of water waves are obtained by solving the wave equations. The shape of the water wave is used to compute the buoyancy. The velocity is used to compute the relative velocity of the flow against the velocity of the object, which is controlled by the user. The fluid resistance is then calculated using the relative velocity by referring to the FRM. The sum of the buoyancy and the fluid resistance is displayed to the user. In addition, the pressure due to the non-linear flow stored in the FRM is applied to the wave equations as external forces. This generates new waves. In this way, the linear flow and the non-linear flow are combined to display the force and simulate the dynamics of water efficiently.

3.2 Force Model

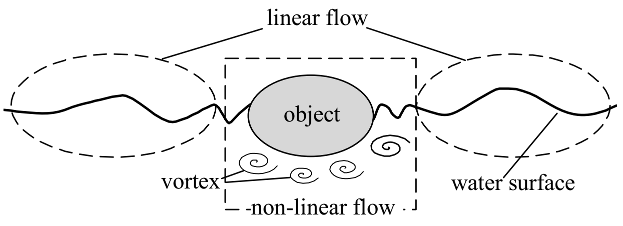
In this subsection, our computation model for the buoyancy and the fluid resistance are described by using Fig. 2(c).

The buoyancy is calculated approximately by using hydrostatic water pressure. The hydrostatic water pressure is the pressure under static water. The force due to hydrostatic pressure, $\mathbf{f}_b(Q)$, on point Q acts in the direction of the normal of point Q (see Fig. 2(c)) and its magnitude is proportional to the depth of point Q below the water surface. That is,

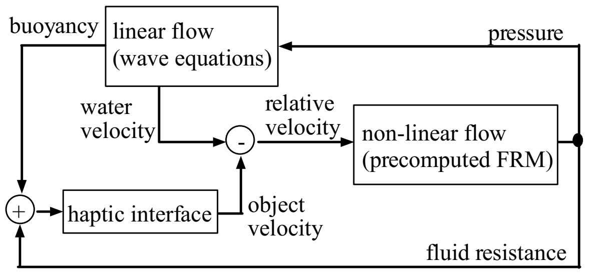
$$\mathbf{f}_b(Q) = -\rho_w w_Q \mathbf{n}_Q \,, \tag{1}$$

where ρ_w is the density of water, w_Q is the depth of point Q, $\mathbf{n}_Q$ is the normal vector of point Q. The buoyancy force of the object is obtained by integrating the above equation over the submerged part of the surface of the object. By definition, the hydrostatic pressure is the water pressure when there are no waves. However, we use depth w_Q taking into account the waves as shown in Fig. 2(c). This allows us to simulate fluctuations of the buoyancy due to the waves.

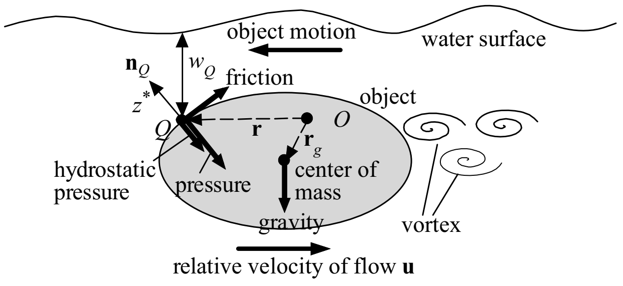
The frictional and the pressure resistances are caused due to the viscosity of water when the object is placed in a flow or when the object moves under water. Basically, these resistances act to prevent the movement of the object. In the following, we assume that the relative velocity of the flow to the velocity of the object is $\mathbf{u}$ (see Fig. 2(c)).



(a) Linear flow and non-linear flow.



(b) Block diagram for our simulation.



(c) Force model.

Figure 2: Our simulation method for water dynamics.

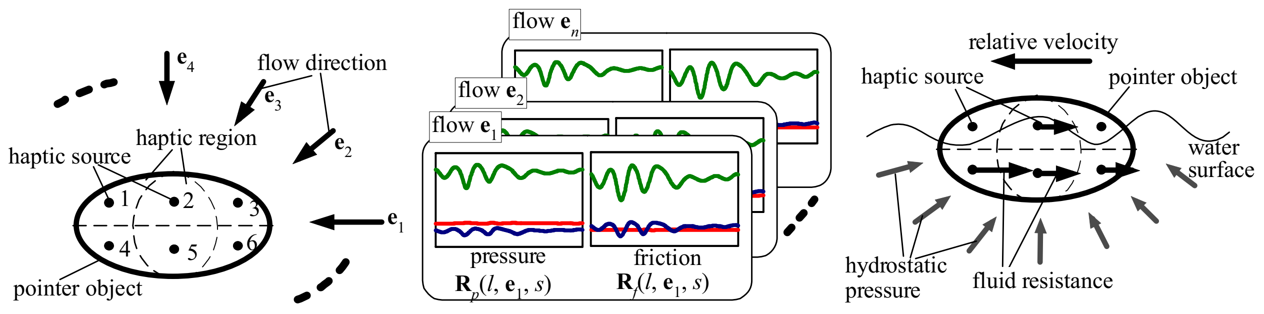
The frictional resistance is due to the friction between the object surface and water. To compute the frictional resistance accurately, we have to compute the flow in the so-called boundary layer, which is turbulent flow layer very close to the object surface [1] [12]. However, the flow in the boundary layer is very complicated and numerical analysis is difficult. Therefore, we employ one of the simplest approximations. That is, the frictional resistance $\mathbf{f}_f(Q)$ on point Q is proportional to the gradient of the relative velocity with respect to the normal direction of point Q [6].

$$\mathbf{f}_f(Q) = \mu \left. \frac{\partial \mathbf{u}^*}{\partial z^*} \right|_{z^*=0} \,, \tag{2}$$

where μ is a dynamic viscosity of water and $\mathbf{u}^*$ is the velocity in a local coordinate system whose z^* axis is the normal vector of point Q (see Fig. 2(c)). The frictional resistance is small for fluids with small viscosity. However, it becomes important for fluids with large viscosity.

The pressure resistance is caused by the difference between pressure in front of the object and pressure behind the object. In general, the pressure behind the object is lower than that in front of the object. The pressure resistance $\mathbf{f}_p(Q)$ on point Q is expressed by the following equation.

$$\mathbf{f}_p(Q) = -p\mathbf{n}_Q \,, \tag{3}$$



(a) Numerical fluid analysis (preprocess). (b) Fluid resistance map for haptic source *l*. (c) Real-time process.

Figure 3: Overview of implementation.

where p is the pressure on point Q.

The total force acting on the object is obtained by integrating the hydrostatic pressure $\mathbf{f}_b$ (Eq. 1) and the fluid resistance $\mathbf{f}_f$ and $\mathbf{f}_p$ (Eqs. 2 and 3) over the underwater part of the object surface, and then adding the effect of gravity of the object. That is,

$$\mathbf{F} = \int_S (\mathbf{f}_b(Q) + \mathbf{f}_p(Q) + \mathbf{f}_f(Q))ds + m\mathbf{g}\,, \qquad (4)$$

where S indicates the underwater part of the surface, m is the mass of the object, and $\mathbf{g}$ is the downward acceleration vector of gravity. A torque is also generated according to the above forces. The torque at point O in the object is calculated by:

$$\mathbf{T} = \int_S \{\mathbf{r} \times (\mathbf{f}_b(Q) + \mathbf{f}_p(Q) + \mathbf{f}_f(Q))\}ds + m(\mathbf{r}_g \times \mathbf{g})\,, \qquad (5)$$

where $\mathbf{r}$ is a vector from point O to point Q, $\mathbf{r}_g$ is a vector from point O to the center of mass (see Fig. 2(c)).

For computing the fluid resistance ($\mathbf{f}_f$ and $\mathbf{f}_p$), the pressure and the velocity distribution around the object are required. Moreover, the fluid resistance is time-varying due to the generation of vortices behind the object, such as the Karman vortex (see Fig. 2(c)). To simulate this efficiently, we make use of the following experimental properties for fluid resistance [4] [7] [15].

1. The magnitude of the pressure resistance is proportional to the square of the relative speed of the flow.
2. The magnitude of the frictional resistance is proportional to the relative speed of the flow.
3. The frequency of the fluctuation of the fluid resistance (i.e. the frequency of the generation of the vortices) is proportional to the relative speed of the flow.

Once the fluid resistance at base speed u_0 is obtained, we can compute the fluid resistance of an arbitrary speed by using the above three properties.

4. OVERVIEW OF IMPLEMENTATION

Fig. 3 shows the overview of the implementation of our simulation method presented in the previous section. First, the FRM is created in a preprocessing step. Then forces and torques are computed in the haptic rendering process. Objects other than water are assumed to be rigid and modeled by polyhedra. The user controls a rigid object by using the 6 DOF haptic interface. In the following, we call the user-controlled object the *pointer object*.

In the preprocess, the pressure and the velocity distributions around the pointer object placed in various directions $\mathbf{e}_i$ of uniform flows are calculated by CFD as shown in Fig. 3(a). The speed of the uniform flows is set to the base speed u_0. Then, the fluid resistances are calculated by using Eqs. 2 and 3, respectively. The results are stored in an FRM. As we mentioned before, the fluid resistance is time-varying. However, storing the time-varying fluid resistance for all points on the object surface requires a large amount of memories. Fortunately, the fluid resistances on neighboring points are very similar. We make use of this coherency. That is, as shown in Fig. 3(a), the surface of the pointer object is subdivided into several regions. We call these regions *haptic regions*. At the center of each haptic region, we generate *a point haptic source*. FRM stores the fluid resistance acting on the haptic region. Fig. 3(b) shows contents of FRM for point haptic source l. FRM consists of the pressure resistance $\mathbf{R}_p(l, \mathbf{e}_i, s)$ and the frictional resistance $\mathbf{R}_f(l, \mathbf{e}_i, s)$. l is the haptic source number and s indicates time. Time series of xyz components of these resistances are stored for each direction of the flow $\mathbf{e}_i$.

In the haptic rendering process, the buoyancy and the fluid resistance are calculated in real-time. The water surface is also displayed in real-time by solving the two-dimensional wave equations. The water surface is expressed as a height field represented by a two-dimensional mesh. At each grid point of the mesh, the height and the velocity of the waves are calculated. Then, the underwater portion of the object surface is extracted. The hydrostatic pressure at each point on the underwater part is calculated to obtain the buoyancy. The fluid resistance is calculated by referring to FRM using the relative velocity of the pointer object against the wave velocity (see. Fig. 3(c)). The obtained fluid resistance is used not only for the force display but also for generating waves due to the movement of the object.

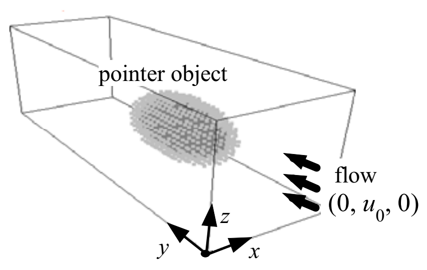
Details of the preprocessing step and the haptic rendering process are described in the subsequent sections.

5. FLUID RESISTANCE MAP

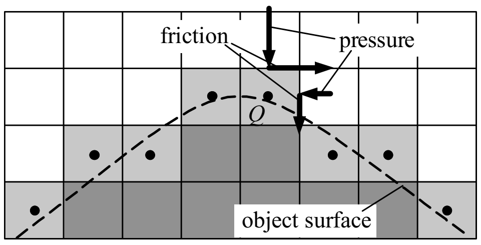
In this section, we describe the method for computing FRM. FRM is created by the numerical analysis of the flow around the pointer object placed in uniform flows with various directions $\mathbf{e}_i$. The speed of the flows is set to the base speed u_0.

5.1 Numerical Fluid Analysis

As shown in Fig. 4(a), an analysis region is specified around the pointer object and is divided into voxels. The flow inside this analysis region is simulated by a finite difference method [21]. The number of voxels is determined experimentally so that the shape of the object is well approximated by the voxels. The time step for the fluid simulation is set as 1/500 sec. Velocity $\mathbf{u} = (u_x, u_y, u_z)$ and pressure p are assigned to each voxel. As a boundary condition, the velocity at $y = 0$ is $(0, u_0, 0)$. A natural boundary



(a) Fluid analysis.



(b) Computation of pressure and friction.

Figure 4: Creating fluid resistance map.

condition is applied to other boundaries. That is, the differentials of velocity and pressure normal to the boundaries are 0. The initial velocities and pressures of all voxels are zero.

Navier-Stokes equations are solved numerically under the above conditions. Then the frictional and the pressure resistances are calculated by using Eqs. 2 and 3. In this fluid analysis, the buoyancy is not taken into account since it is calculated in the real-time process using Eq. 1. This is equivalent to solving Navier-Stokes equations without taking into account the gravity. The fluid simulation is computed for a specified number of time steps. During the early time steps, however, the simulation is in a transition state and is unstable. So, the computation of the fluid resistance begins after the simulation becomes stable.

5.2 Generating Fluid Resistance Map

The frictional and the pressure resistances are calculated by using velocities and pressures at voxels corresponding to the object surface. As shown in Fig. 4(b), the object is represented by voxels (Fig. 4(b) shows a two-dimensional case for simplicity). Therefore, the resistances on the voxel are obtained by computing the resistances acting on the side faces of the voxel. The pressure resistance is perpendicular to the faces and the frictional resistance is tangential to the faces as shown in Fig. 4(b). Then, the sum of the resistances of the voxels corresponding to the surface of each haptic region is stored.

The above computation is repeated at each time step of the simulation. After the simulation, time series of the fluid resistances are stored in FRM (see Fig. 3(b)). Unlike the method by Baxter and Lin [2], our method does not produce the undesirable noises. The fluid resistances stored in the FRM are smooth and continuous since the pointer object is discretized only in this preprocessing step and the descretized shape does not change throughout the simulation. In the real-time haptic rendering process, we use FRM periodically since the size of FRM is finite. However, FRM obtained by the above method is not periodic and hence the user would feel the discontinuous forces at the ends of FRM. To address this, we transform the time series into the frequency domain by FFT and removes high frequency components, whose amplitudes are smaller than a specified threshold. After that, we transform them back into the time domain. This process makes FRM periodic.

6. REAL-TIME HAPTIC RENDERING

In the real-time process, the images of water and the objects are displayed at 30 Hz. On the other hand, the forces are computed at 500 Hz. Experimentally, we found that 500 Hz is sufficient for stable calculations since the forces due to water do not change rapidly. The following procedure is executed at every 1/500 sec.

1. Update of the position and the velocity of the pointer object using the pointer position of the haptic interface.
2. Computation of the buoyancy
3. Computation of the fluid resistance
4. Computation of the water surface
5. Display of the resulting forces and the torques using the haptic interface

Before starting the simulation, sample points to detect the underwater part of the pointer object are generated on the surface of the pointer object as follows. Each polygon of the pointer object is subdivided into a specified number of triangles. Then the sample points are generated at the centers of the triangles. An area and a normal of the triangle are stored for each sample point.

In the following subsections, Details of steps 3, 4, and 5 in the above procedure are described.

6.1 Computing Buoyancy

The buoyancy is obtained by computing the hydrostatic pressures (Eq. 1) at the sample points on the surface of the pointer object. First, the sample points under the water surface are extracted. Since the water surface is represented by the height field, it is very easy to check if the sample point is under the water surface or not. Then the hydrostatic pressures are calculated at those sample points. The buoyancy is obtained by summing the resulting hydrostatic pressures after multiplying the areas stored for the sample points.

6.2 Computing Fluid Resistance

The fluid resistance is calculated by using FRM. FRM stores the frictional resistance $\mathbf{R}_f(l, \mathbf{e}_i, s)$ and the pressure resistance $\mathbf{R}_p(l, \mathbf{e}_i, s)$ acting on haptic source l of the pointer object placed in a flow (speed u_0 and direction $\mathbf{e}_i$) at time s. To avoid confusion, we denote the time in FRM as s. The actual time is t. The fluid resistance acting on the pointer object moving at an arbitrary velocity is calculated by making use of the three properties described in Section 3.2. The basic idea is to interpolate the resistances stored in FRM by using the velocity of each haptic source. Details are described in the following.

Let us assume that time $t = 0$ when the haptic rendering process starts. The fluid resistance is calculated at a short time interval Δt (= 1/500 in our case). The total fluid resistance is obtained by summing the resistances at each haptic source. In the following, $t_k = k\Delta t$, where k is a non-negative integer indicating time steps in the simulation. We explain the computation method for haptic source l.

First, a ratio of area of the underwater part of the haptic region of haptic source l is calculated by using the sample points generated on the surface of the pointer object. Let us denote the ratio as α_l. When $\alpha_l = 0$, the fluid resistance is set to $\mathbf{0}$. Otherwise, first, the velocity of haptic source l is calculated according to the

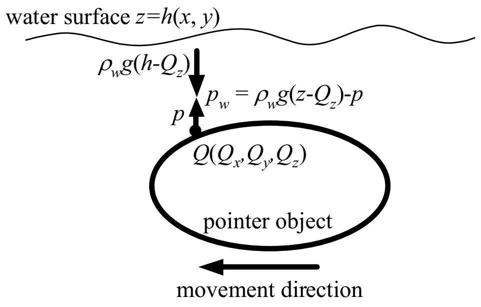


Figure 5: Computing water surface.

movement of the pointer object. The velocity of the waves of the water surface is also calculated by the method described in the next subsection. The relative velocity of the haptic source against the velocity of the water waves is used to calculate the fluid resistance. This can simulate the reaction due to water waves. Let us denote the magnitude of the relative velocity as $u_l(t_k)$ and the direction as $\mathbf{e}_l(t_k)$. Then the fluid resistance is calculated by referring to FRM using $u_l(t_k)$ and $\mathbf{e}_l(t_k)$. That is, the pressure resistance $\mathbf{f}_{l,p}$ and the frictional resistance $\mathbf{f}_{l,f}$ are calculated by the following equations so that the three properties in Section 3.2 are satisfied.

$$\begin{cases} \mathbf{f}_{l,p}(t_k) = \alpha_l (u_l(t_k)/u_0)^2 \mathbf{R}_p(l,\mathbf{e}_l,s_k) \\ \mathbf{f}_{l,f}(t_k) = \alpha_l (u_l(t_k)/u_0) \mathbf{R}_f(l,\mathbf{e}_l,s_k) \quad , \\ s_k = (u_l(t_k)/u_0)\Delta t + s_{k-1} \end{cases} \tag{6}$$

where $s_0 = 0$. As shown in the above equations, when the relative speed is $u_l(t_k)$, the value at intervals of $(u_l(t_k)/u_0)\Delta t$ in FRM is used to compute $\mathbf{f}_{l,p}$ and $\mathbf{f}_{l,f}$. This is equivalent to making the frequency of the fluctuation of the resistances $(u_l(t_k)/u_0)$ times higher (or lower). The magnitudes of the pressure and the frictional resistances are proportional to $(u_l(t_k)/u_0)^2$ and $(u_l(t_k)/u_0)$, respectively. So, the properties described in Section 3.2 are satisfied.

The above processes are repeated for all haptic sources and the total fluid resistance is obtained by summing them.

6.3 Computing Water Surface

When the pointer object moves near the water surface, waves are generated. We simulate this by combining FRM with the method proposed in [13].

Similarly to [13], we employ the following assumptions: the effect of the viscosity is very small, the speed of the water waves is slow, and amplitudes of the waves are very small to the depth from the bottom of the water. Then, the water surface is represented by a height field $h(x, y)$. Navier-Stokes equations are linearized as follows.

$$\frac{\partial \mathbf{u}}{\partial t} = -\frac{1}{\rho_w}\nabla p_w , \tag{7}$$

$$\frac{\partial h}{\partial t} = -d\nabla \cdot \mathbf{u} , \tag{8}$$

where $\mathbf{u} = (u_x, u_y, u_z)$ is a velocity vector, ρ_w is a fluid density coefficient, d is a parameter of the wave equation, meaning the average water height from the bottom. The waves caused by the motion of the pointer object are simulated as follows. Inside the pointer object, the velocity of water is forced to the velocity of the object. In addition, the object pushes the water with pressure p due to the movement of the object. This effect is simulated as follows. Let us consider a pressure at point $Q(Q_x, Q_y, Q_z)$ on the pointer object as shown in Fig. 5. The total pressure p_w at point Q is the sum of pressure p due to the movement of the pointer object and the weight of water above point Q. That is,

$$p_w = \rho_w g(h - Q_z) - p , \tag{9}$$

where g is the acceleration of gravity. By putting Eq. 9 into Eq. 7, the following equation is obtained.

$$\frac{\partial \mathbf{u}}{\partial t} = -g\nabla h + \frac{1}{\rho_w}\nabla p . \tag{10}$$

We solve Eqs. 8 and 10 to obtain the shape and the velocity of the water waves. These equations are solved numerically by a finite difference method. To compute the water surface efficiently, we assume that the vertical speed of the waves is very small compared to the horizontal speed, that is $u_z = 0$. The pressure p is obtained by a weighted average of pressures at neighboring haptic sources, calculated by the method described in Section 6.2. We use a Gaussian function for computing the weights. We consider only the upper part of the object surface as shown in Fig. 5. The pressure on the upper part of the object surface directly reaches the water surface under the assumption of small viscosity. Since the water surface is represented by two-dimensional grids, we evaluate the pressure gradient in Eq. (10) at each grid point above the surface of the pointer object.

7. RESULTS

In this section, we first verify the validity of our method by comparing measured data with simulation results. Next, several examples are demonstrated, where the 6 DOF haptic interface named SPIDAR-G [14] is used to interact with the virtual world while display the forces and the torques. We connected SPIDAR-G with a desktop PC (Pentium IV 2.8GHz).

7.1 Comparison with Measured Data

We measured the forces acting on a parallelopiped. The dimension of the parallelopiped is 5 cm x 1 cm x 50 cm. We chose the parallelopiped in order to verify the capability of our method to handle the anisotropic shape. The experimental setup is shown in Fig. 6. We used a robot arm to measure the force. A force sensor is attached to the tip of the arm. The parallelopiped is then attached to the force sensor. We programmed the arm to track a quarter of a circle with the 4/5 part of the parallelopiped underwater. We chose two speeds of the arm movement, 7.5 cm/s, and 10 cm/s. Furthermore, for each speed, we measured the forces twice by attaching the parallelopiped at different angles, that is, 0 and 60 degrees. Zero degree corresponds to moving the parallelopiped in the direction perpendicular to its wider face. We measured the forces in the inverse and perpendicular directions to the movement of the arm. We compared the measured data with the simulated forces under the same situation. For the simulation, we created FRM by two-dimensional fluid analysis using the cross-sectional shapes of the parallelopiped. The number of the directions of uniform flows for creating FRM is 36. The base speed is $u_0 = 50$ cm/s.

Fig. 7 shows our comparison. Figs. (a) and (b) correspond to the forces at speed 7.5 cm/s, respectively and Figs. (c) and (d) to the forces at speed 10 cm/s. We measured the forces by making the robot arm go back and forth several times. The sharp peaks in Fig. 7 correspond to the turning point of the movement. The black lines correspond to the measured forces and the gray lines to the simulated forces. The thick lines correspond to the forces in the

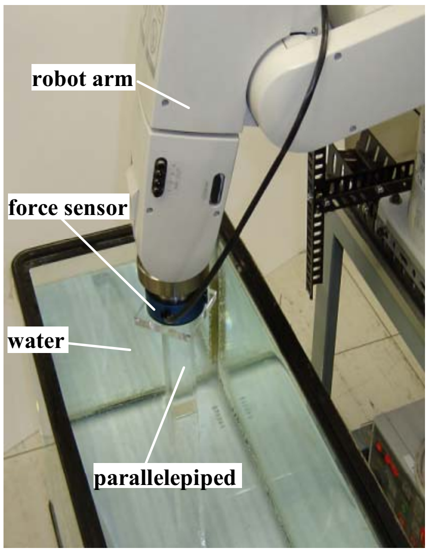
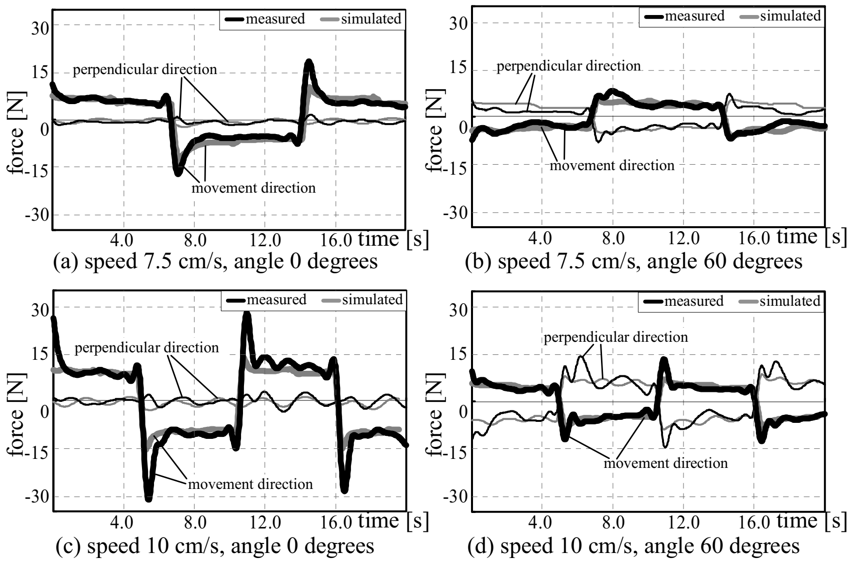


Figure 6: Experimental setup.



(a) speed 7.5 cm/s, angle 0 degrees (b) speed 7.5 cm/s, angle 60 degrees (c) speed 10 cm/s, angle 0 degrees (d) speed 10 cm/s, angle 60 degrees

Figure 7: Comparison with measured forces.

direction of the movement and thin lines to the forces perpendicular to the movement. At the turning point, there are sudden changes of the forces since the relative velocity of the parallelopiped to the surrounding water increases at that time. Similar effects are also observed in the simulated data. As shown in Fig. 7, our method reproduced the forces very similar to the measured forces. However, some differences are observed around the peaks. This is because the inertia forces were included in the measured data. When the direction of the movement changes, the force sensor measured the forces from inertia of the parallelopiped as well as the forces due to water. The inertia forces were not calculated in the simulation.

7.2 Examples using Haptic Interface

Figs. 8 and 9 show simple applications with the haptic interface. In Fig. 8, the user controls a sphere with radius 5 cm and feel the forces in real-time when the sphere interacts with water. The water surface is represented by 128 x 128 grids. The numbers of haptic sources are 26 and 20 for Figs. 8 and 9, respectively. The user moves the sphere in a clockwise circular orbit. Figs. 8(a) through 8(c) correspond to the images at time 0.7, 1.3, and 1.9 seconds. The arrows in these images indicate the direction of the forces. Fig. 8(d) shows the relation between the speed of the sphere and the force. Three components of the force are shown: forces in the movement direction, the horizontal direction, and vertical direction to the movement. The force against the movement is dominant and the user hardly feels the forces horizontal and vertical to the movement. This is because the flow around the sphere is smooth and symmetrical. In Fig. 9, the user controls a parallelopiped instead of the sphere. The dimension of the parallelopiped is 8 cm x 1 cm x 30 cm. When the user tries to move the parallelopiped forward as shown in this figure, the strong fluid resistance acts in the horizontal direction (see Fig. 9(d)) they cause the parallelepiped to move left and right alternatively. We often observe this phenomenon in the real world. In Figs. 8 and 9, the images of the water surface are rendered by mapping a sky image using an environmental texture mapping technique.

Figs. 1 and 10 show more practical applications. Fig. 1 is a canoe simulator. We constructed a human-scale haptic interface by extending the SPIDAR-G haptic interface. The user controls the paddle using this interface. The buoyancy and the fluid resistance are computed not only for the paddle but also for the canoe. The canoe moves according to the forces due to water as well as the forces due to the paddle. Fig. 10 shows a lure-fishing simulator. To improve the reality, we attached to SPIDAR-G a stick that simulates a rod as shown in Fig. 10(a). The force on the underwater fish lure is calculated and this drags the tip of the rod. In these examples, the numbers of the haptic sources are 40 and 30 for Figs. 1 and 10, respectively. The water surface is represented by 200 x 200 and 160 x 200 grids for Figs. 1 and 10, respectively.

In the examples shown in this subsection, the precomputation times for FRM ranged from 1 to 24 hours. The sizes of FRM ranged from 1 MB to 10 MB. In the haptic rendering process, the forces and torques are calculated and displayed at 500 Hz. Please refer to the movies corresponding to each of the examples shown in this subsection, provided in the supplementary submitted files.

8. DISCUSSION

When using our method, the following points should be kept in mind.

First, our method cannot take into account randomness due to turbulent flow. However, we believe that our method reproduces the distinctive features of the flow forces. Hence, our presented method is very viable for haptic interfaces in a virtual reality systems. To reproduce the randomness caused by turbulent flow, Navier-Stokes equations must be fully solved taking into account the water velocity around the submerged object. However, this is very time-consuming and therefore is not applicable to haptic interactions requiring the high frame rate.

Secondly, when the user rotates the pointer object, the forces computed by our method are not totally correct. This is because the FRM is created only for uniform flows. The forces due to rotation are not stored in the FRM. In our method, the forces due to rotation are calculated by assuming that each haptic source is placed in a uniform flow at the speed of the rotation during the short time interval of a real-time simulation. To simulate the forces due to the rotation accurately, the rotational motion of the pointer object must be simulated in a preprocessing step and the resulting forces need to be stored. Again, they will lead to a significant increase in the computation time and the memory requirement.

Thirdly, the proposed method cannot reproduce the forces due to the other objects near the pointer object accurately. This effect is simulated approximately in the following way. The objects near the pointer object can generate water waves calculated by wave equations. These water waves affect the forces acting on the pointer object since the force is determined by the relative velocity of the pointer object against the velocity of the water wave.

Finally, we would like to mention that our method can be applied to the simulation of forces due to other fluids such as air. In this case, the buoyancy forces can be ignored since they are rather small compared to the fluid resistance forces.

9. CONCLUSION

We have presented a method for the real-time display of the forces acting on rigid objects due to water. Our method can display the buoyancy, the frictional resistance, and the pressure resistance. To compute these forces in real-time, we have presented a method for simulating the dynamics of water. The water dynamics was decomposed into linear flow and non-linear flow. For linear flow, we used the wave equations to create water waves. For non-linear flow, we have presented an FRM, the fluid resistance map. The FRM is created in a preprocessing step by solving Navier-Stokes equations, and the resulting fluid resistance is stored. Then, the wave equations and the FRM are combined to compute the forces due to water in real-time. The accuracy of our method has been demonstrated by comparison with actual measured data. The usefulness of the method has also been demonstrated by several examples using the SPIDAR-G haptic interface.

Our method provides a natural way to interact with fluids in a virtual world. We believe that this greatly enhances the reality of the virtual environment.

There are a few things to be done in the future. Currently, we determined the number of the haptic sources experimentally. However, the computational cost for the real-time haptic rendering process is proportional to the number. Therefore, an adaptive method for determining the optimal number and positions of the haptic sources need to be developed to speed up the computation without losing the accuracy. Next, as we discussed in Section 8, the fluid resistances corresponding to various velocities and rotational motions of the pointer objects should be stored in FRM in order to display the forces accurately under more complex situations. In this case, we have to develop a compression method suitable for FRM since the memory requirement for storing such fluid resistances increases significantly.

10. REFERENCES

[1] Anderson, J. D., JR., Computational Fluid Dynamics, McGraw-Hill (USA), 1995.

[2] Baxter, W. and Lin, M. C., Haptic Interaction with Fluid Media, In *Proceedings of Graphics Interface2004*, Annual Conference Series, 2004, 81-88.

[3] Baxter, B., Scheib, V., Lin, M. C., and Manocha, D., DAB, Interactive Haptic Painting with 3D Virtual Brushes, In *Proceedings of ACM SIGGRAPH 2001*, Annual Conference Series, 2001, 461-468.

[4] BloomerJ., Practical Fluid Mechanics for Engineering Applications," *Marcel Dekker*, 2000.

[5] Carlson, M., Mucha, P. J., and Turk, G., Rigid Fluid: Animating the Interplay Between Rigid Bodies and Fluid, ACM Transaction on Graphics (*Proceedings of ACM SIGGRAPH 2004)*, 23, 3, 2004, 377-384.

[6] Chung, T. J., Computational Fluid Dynamics, Cambridge University Press, 2002.

[7] Dobashi, Y., Yamamoto, T., and Nishita, T., Real-time Rendering of Aerodynamic Sound Using Sound Textures based on Computational Fluid Dynamics, *ACM Transaction on Graphics (Proceedings of SIGGRAPH 2003)*, *23*, 3, 2003, 732-740.

[8] Durbeck, L., Macias, N., Weinsten, D., Johnson, C., and Hollerback, J., SCIRun Haptic Display for Scientific Visualization, *Phantom Users Group Meetings*, 1998.

[9] Enright, D., Marshner, S., and Fedkiw R., Animation and Rendering of Complex Water Surfaces, *ACM. Transaction on Graphics (Proceedings of ACM SIGGRAPH2002)*, *21*, 3, 2002, 737-744.

[10] Foster, N. and Fedkiw, R., Practical Animation of Liquids, In *Proceedings of SIGGRAPH2001*, Annual Conference Series, 2001, 23-30.

[11] Hasegawa, S. and Sato, M., Real-time Rigid Body Simulation for Haptic Interactions based on Contact Volume of Polygonal Objects, *Computer Graphics Forum* (*Proceedings of EUROGRAPHICS 2004*), *23*, 3, 2004, 529-538.

[12] Hoerner, F. S., Fluid Dynamic Drag: Practical Information on Aerodynamic Drag and Hydrodynamic Resistance, Hoerner Fluid Dynamics, 1965.

[13] Kass, M. and Miller, G., Rapid, Stable Fluid Dynamics for Computer Graphics, *Computer Graphics (Proceedings of ACM SIGGRAPH 1990)*, *24*, 4, 1990, 49-57.

[14] Kim, S., Hasegawa, S., Koike, Y., and Sato, M., Tension based 7-dof force feedback device: Spidar-g, *In Proceedings of IEEE Virtual Reality 2002*, Annual Conference Series, 2002, 283-284.

[15] Landau, L. D. and Lifschitz, E. M., Fluid Mechanics, *Pergamon Press*, 1975.

[16] Mark, W. R., Randolph, S. C., Van Verth, J. M., and Taylor II, R. M., Adding Force Feedback to Graphics Systems: Issues and Solutions, In *Proceedings of ACM SIGGRAPH 96*, Annual Conference Series, 1996, 447 - 452.

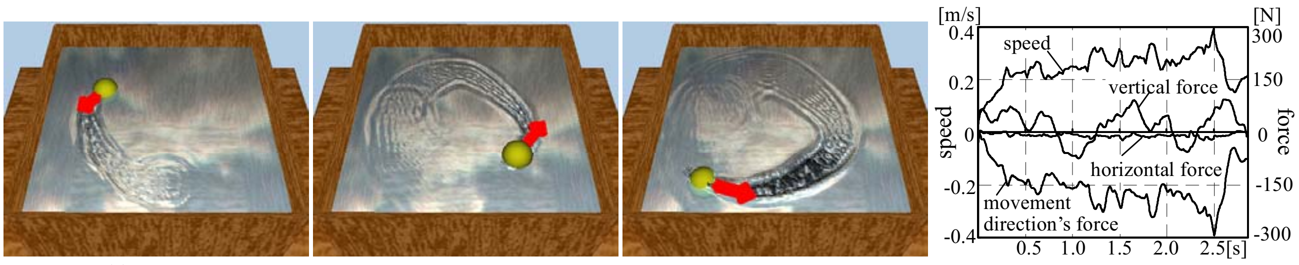
[17] Mcneely, W. A., Puterbaugh, K. D., and Troy, J. J., Six Degree-of-Freedom Haptic Rendering Using Voxel Sampling, In *Proceedings of ACM SIGGRAPH 99*, Annual Conference Series, 1999, 401 - 408.

[18] Otaduy, M. A. and Lin, M. C., Sensation Preserving Simplification for Haptic Rendering, *ACM Transaction on Graphics (Proceedings of ACM SIGGRAPH 2003)*, *23*, 3, 2003, 543-553

[19] Pai, D. K, Kees, van den D., James, D. L., Lang, J., Lloyd, J. E., Richmond, J. L., Yau, S. H., Scanning Physical Interaction Behavior of 3D Objects, In Proceedings of SIGGRAPH 2001, Annual Conference Series, 2001, 87-96.

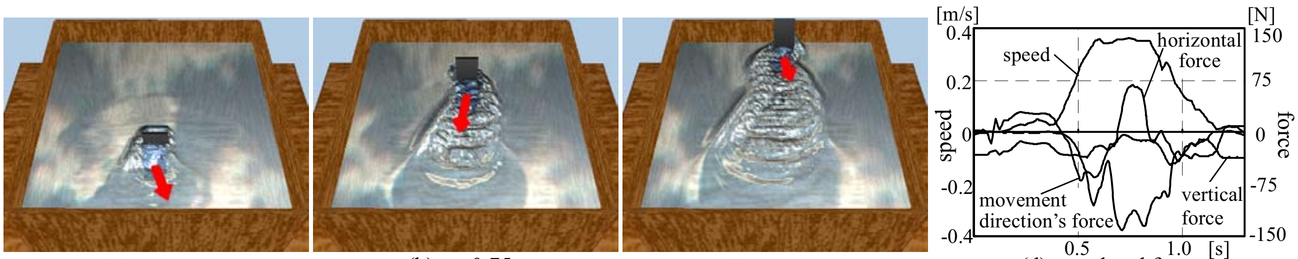
[20] Ruspini, D. C., Kolarov, K., and Khatib, O., The Haptic Display of Complex Graphical Environments, In *Proceedings of ACM SIGGRAPH 97*, Annual Conference Series, 1997, 345-352.

[21] Stam, J., Stable Fluids, In *Proceedings of ACM SIGGRAPH 99*, Annual Conference Series, 1999, 121-128.



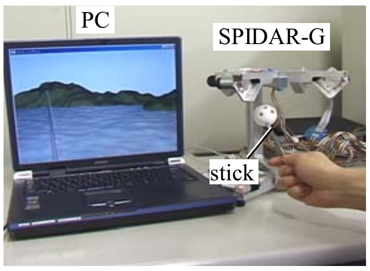
(a) $t = 0.7$ (d) $t = 1.3$ (c) $t = 1.9$ (d) speed and force

Figure 8: Interaction of a sphere with water. Figs. (a)-(c) show different snapshot images during the interaction. The arrow indicates the direction of the force due to water. (d) shows the relation between the speed of the sphere and the force. Three components of the force are shown, that is, forces in the movement direction, the horizontal direction and vertical direction to the movement

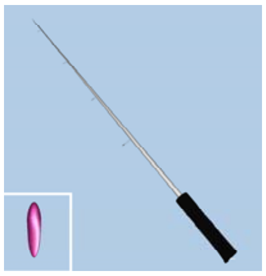


(a) $t = 0.6$ (b) $t = 0.75$ (c) $t = 0.9$ (d) speed and force

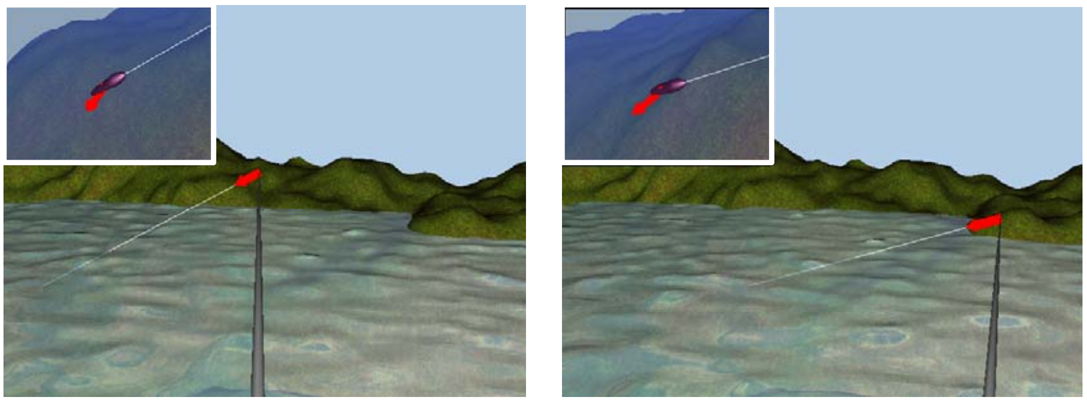
Figure 9: Interaction of a parallelopiped with water. The user moves the parallelopiped forward, however, the parallelopiped moves left and right slightly due to the creation of wake vortices (see horizontal force in Fig. (d)).



(a) Setup for fish simulator.



(b) Virtual lure and rod.



(c) Example images during simulation.

Figure 10: A simulator for lure-fishing. (a) shows the haptic interface with a stick attached to simulate the rod. (b) shows the virtual lure and the rod. (c) shows images during the simulation. In the upper left corner of the images in (c), the motion of the underwater lure is displayed. The force on the lure and the force dragging the tip of the rod are indicated by the arrows.