Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2009)
H. Prautzsch, A. Schmitt, J. Bender, M. Teschner (Editors)

An Application of Photo Realistic Water Surface Interaction Using Mixed Reality

T. Tawara and K. Ono

RIKEN, Japan

Abstract

The Reality of Virtual Reality is affected by many research areas. Therefore, individual researches must be combined to achieve the extreme goal of Virtual Reality. In this paper, we present an application of the novel and clever combination of height field wave simulation, photo realistic rendering, and a 6DOF manipulator exploiting Augmented Reality. In our system, a user can touch a virtual water surface with a real pen attached a tracking cube in natural manners. We also take into account rendering optical effects like shadows and caustics, which give users a great deal of reality. We show how such a combination is important to achieve reality in the videos of our real time demonstrations.

Categories and Subject Descriptors (according to ACM CCS): Information Interfaces and Presentation [K.5.1]: Multimedia Information Systems—Artificial, augmented, and virtual realities; Information Interfaces and Presentation [K.5.2]: User Interfaces—Interaction styles; Computer Graphics [I.3.6]: Methodology and Techniques—Interaction techniques Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Virtual reality

1. Introduction

As the recent growth of computational powers, a real time fluid simulation gains an important role and becomes common in computer graphics. Although interaction techniques become important as real time feedbacks are demanded, user interfaces to achieve them are often limited or specialized. So far, a general device to realize intuitive interactions is not appeared. A common user interface is a mouse which can handle only two degree of freedom. On the other hand, natural operations require six degree of freedom.

In the Virtual Reality (VR) research field, various input devices are developed: wired gloves, magnetic or infrared tracking devices, haptic devices, force feedback devices, and others. However such devices are usually expensive and sometimes need external equipments. Furthermore, such devices may fix on the ground and require a large space to set up equipments [AR06]. Recently, Augmented Reality (AR) attracts a great deal of interest. VR fully builds a virtual world in a computer. By contrast, AR exploits computer vision techniques and projects computer graphics in the real scene captured by a web camera, which is a live video camera connected by a USB interface.

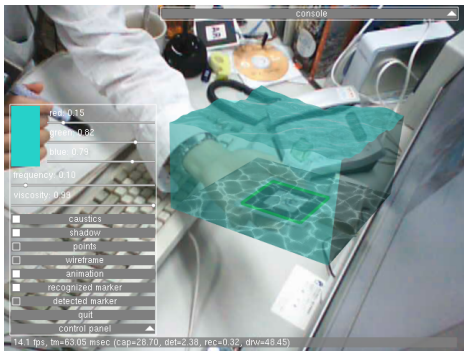

Figure 1: *A side view of a water tank on the user's palm.*

In this paper, we present an application of the novel and clever combination of height field wave simulation, photo realistic rendering, and a 6DOF manipulator exploiting Augmented Reality. In our system, a user can touch a virtual water surface with a real pen attached a tracking cube in natural manners. Only a web camera and printed markers are required by using a marker based motion tracking tech-

nique. This nature is preferred for general PC users because a web camera becomes a reasonable and common device for present PCs. We also take into account rendering optical effects like shadows and caustics, which give users a great deal of reality.

The rest of this paper is organized as follows: Section 2 summarizes previous work. Section 3 explains background techniques. Section 4 introduces our AR environment and user interface. Section 5 details how to interact between a real pen and a virtual water surface. Section 6 describes rendering shadows and caustics. Section 7 describes results of our demonstrations created with the proposed method. Finally, we conclude the paper with a discussion of possible future work in Section 8.

2. Previous Work

There are several established frameworks to build AR applications including ARToolKit [ART, KB99, KBP*00] and ARTag [Fia05]. ARToolKit is probably the best known framework, and is widely used in the AR research field. It detects and recognizes printed markers and returns each transformation matrix from a marker's local coordinate system to a view coordinate system using computer vision based methods. We have built our application based on this framework to handle interaction between a real pen and a virtual water surface.

In computer graphics, many researches for realistic fluid dynamics have been done in recent years. Various approaches are proposed to achieve their own demands including movies, commercial films, and real time games. One of major approach to represent real time water surface simulation is to use a height field [Tes04b, Tes04a, BMF07]. We add a simple model to add an external force from a real pen on this simulation.

Shadow is a very important visual component, thus a number of researches for shadowing have been achieved. Shadows tell us the approximate distance between objects casting and receiving shadows. It is the first step beyond a local illumination model, which lighting effects are defined only on a local surface point.

Another important visual component of rendering a water surface is caustics [NN94, IDN01, IDN03]. Lights refract at a water surface, and converge and diverge, and create complex striped patterns on the bottom of a water surface.

Although quite a number of researches were proposed for AR applications, a very few numbers of researches combined fluid simulation and AR techniques [IAY*06, IYMC06]. In authors' knowledge, only Imura et al. [IAY*06] introduced a similar research handling a real object and virtual water. They used the Smoothed Particle Hydrodynamics (SPH) method for fluid dynamics. They proposed an interaction technique between virtual water and real objects, which its shapes are known beforehand, using magnetic 6DOF sensors that relative to a table. On the other hand, they did not consider rendering shadows and caustics.

3. Background

In this section, we briefly explain background knowledge to build our application.

3.1. Augmented Reality and Marker based Motion Tracking

In Augmented Reality, the most major approach to track motions is to use printed markers [ART, KB99, KBP*00]. A general AR system captures the real scene using a web camera, then finds markers in a video frame. Next, a local coordinate system is defined for each marker and the transformation matrix to a view coordinate system is computed for each one. Finally, computer generated objects are projected on the markers in a video frame. Figure 2 shows an overview of a general marker based AR system. A computer display might be replaced by a head mount display (HMD) attached a web camera. In that situation, video frames follow the user's motions.

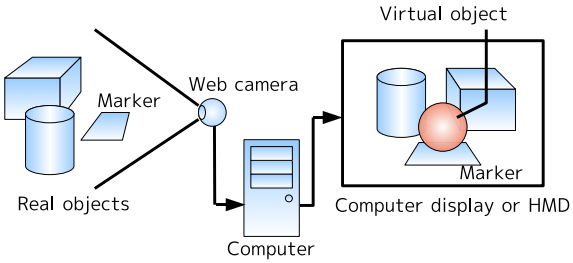

Figure 2: *A general augmented reality system.*

3.2. Wave Equation

Two dimensional wave equation of a height function $h = h(x,y,t)$ is written as:

$$\frac{\partial^2 h}{\partial t^2} = c^2 \nabla^2 h$$

where c is the speed of a wave. This equation is a second order partial differential equation (PDE). To solve the equation, it is divided in two first order PDE: $\frac{\partial h}{\partial t} = v$, $\frac{\partial v}{\partial t} = c^2(\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2})$ The equations are written in discrete representations using Taylor expansion, and then the velocity and height are updated by the fraction of time $\triangle t$:

$$v_{i,j}^{t+1} = v_{i,j}^{t} + \triangle t c^2 (h_{i+1,j} + h_{i-1,j} + h_{i,j+1} + h_{i,j-1} - 4h_{i,j})/d^2$$

$$h_{i,j}^{t+1} = h_{i,j}^{t} + \triangle t v_{i,j}^{t+1}$$

where d is the distance between grid points. Detail description can be consulted great references [Tes04b, Tes04a, BMF07].

4. AR Environment and User Interface

Our system consists of a web camera, a marker representing a water surface, and an AR pen. Figure 3 shows our AR environment. The camera is placed at the enough distant position to enable to see all markers.

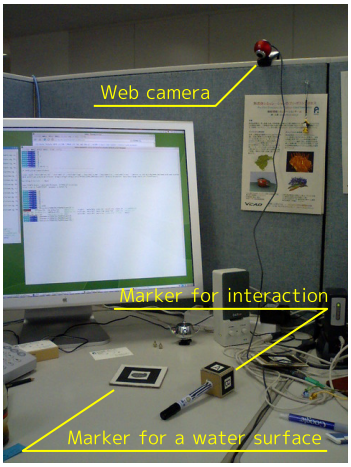

Figure 3: *Our AR environment consists of a web camera, a marker for representing a water surface and an AR pen.*

We introduce our AR pen shown in Figure 4. A user can touch a virtual water surface using it. It consists of a real pen and a motion tracking cube. Each side of the cube has a different AR marker. The tip of a pen is defined as the relative position from each marker. At least one of the markers should be visible from a web camera during operations. If more than one marker are visible, the most well posed marker, which is nearly perpendicular to a viewing direction, is used.

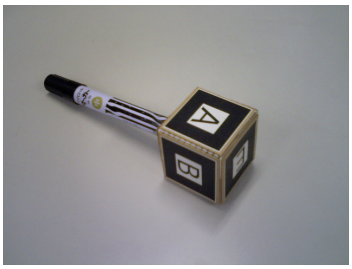

Figure 4: *The AR pen: A cube composed of five AR markers is attached at the end of a pen.*

5. Interaction with a Wave Surface

In this section, we detail our interaction techniques between an AR pen and a virtual water surface.

5.1. Transformation Matrix

As mentioned before, each marker returns a 4×4 transformation matrix respecting to a viewing coordinate system using ARToolKit [ART]. On the other hand, each marker does not know each other. To interact with a water surface using an AR pen, it is necessary to use the same coordinate system. It is convenient to use the coordinate system on the water surface as the base for interaction because many interactions occur on the water surface including adding external forces, computing shadows and caustics, handling obstacles. Figure 5 illustrates transformations around local coordinate systems. We build combinations of transformation matrices for interaction and rendering. The position of the tip of an AR pen can transform on the water surface by way of the viewing coordinate system.

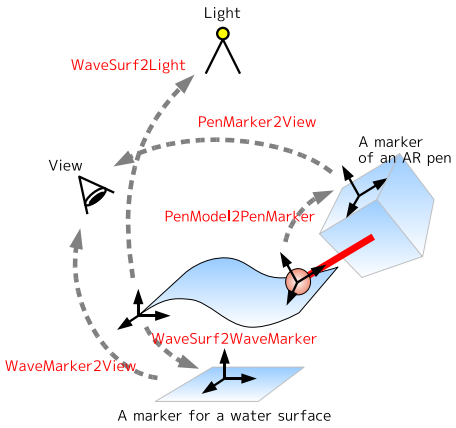

Figure 5: *Transformation Matrices.*

5.2. External Forces

When the tip of an AR pen touches on a water surface, vertically downward forces are added to a water surface. A simple and easy to control model is desired because our purpose is real time interaction. We use a Gaussian function to represent the distribution of acceleration around the tip of an AR pen. Acceleration of an AR pen is modeled as follows:

$$a(d) = s\exp(-\frac{1}{2c^2}\frac{d^2}{r^2})$$

where s is the scale of force, r is a user defined radius, and d is the distance from the center of the tip of an AR pen. The constant c is set to .2 as $a(d)$ becomes close to zero at $d = r$. To satisfy volume conservation, an acceleration in the

opposite direction a_{up} is added for each computational grid point:

$$a_{total} = \sum_{i,j \in radius} a_{i,j}$$

$$a_{up} = -\frac{a_{total}}{n_x n_y}$$

where n_x and n_y are the numbers of computational grid points for x and y axes respectively. We ignore out of radius contributing a negligible amount of force. From observation of trial and error, we decide that force is constantly added in the downward direction if the tip of an AR pen is under the water surface. Figure 6 illustrates directions and scales of acceleration for all computational grid points. Computed accelerations are added to velocities of the next time step after multiplying $\triangle tc^2$ in the equation presented in Section 3.2.

Figure 7 shows the visualization of accelerations for every computational grid point. Red points show upward accelerations and blue points show downward them.

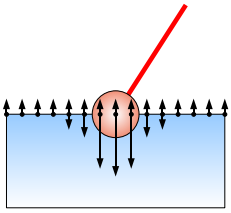

Figure 6: *External forces on a water surface.*

Figure 7: *Visualization of acceleration. Red points show upward acceleration and blue points show downward one.*

5.3. Frequency Control

In our software, a user can control the frequency of waves by changing the user defined radius presented in the previous section. When the radius sets to smaller, then a higher frequency, smaller wave appears. On the other hand the radius sets to larger, then a lower frequency, larger wave is

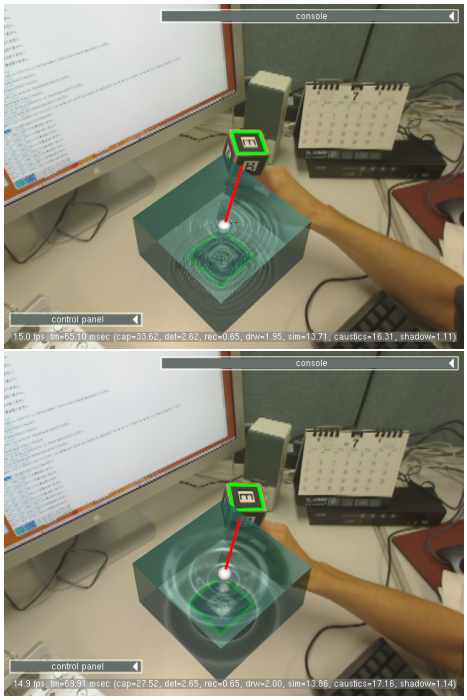

Figure 8: *Difference of frequencies of waves. upper) high frequency wave, lower) low frequency wave.*

generated. Figure 8 shows the difference of frequencies of waves.

5.4. Viscosity Control

For every time step, velocities of the next time step are attenuated by multiplying an attenuation coefficient. A user can control viscosity by changing the amount of an attenuation coefficient. When an attenuation coefficient sets to zero, waves oscillate forever. While an attenuation coefficient becomes bigger, then fluid restricts the movement. Therefore, viscosity becomes high. Figure 9 shows a fluid like a honey. It shows that a longer trajectory remains.

5.5. Obstacles

Obstacles are prepared as volume data, which represents inside or outside of obstacles by 0 or 1 respectively. During simulation each computational grid point on a wave surface is transformed into the obstacle's coordinate system then the point is checked whether it is inside or outside of obstacles. If the point is inside of obstacles, the velocity and height of the corresponding grid point on a wave surface set to zero. This simple boundary condition around obstacles makes reflection of a wave. In fact, this is not physically correct because the endpoint of a wave is fixed on the surface of an obstacle. Nevertheless, this simple approach creates nice look-

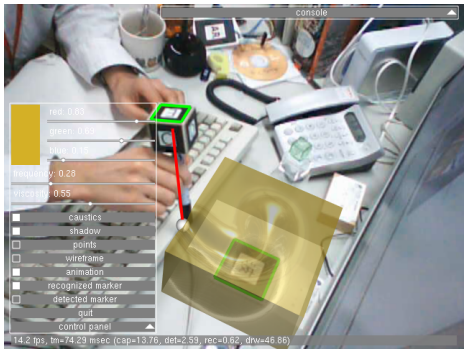

Figure 9: *Fluid like a honey.*

Figure 10: *Demo of obstacles. Two blocks make a slit and waves are passing through it from the right side of blocks. Caustics nicely represent passing waves.*

ing results. Figure 10 shows obstacles in a virtual water tank. The image shows that waves are passing through a slit made by two blocks.

6. Rendering

We adopt three pass rendering method for each frame in our software: 1st pass) computing a shadow map, 2nd pass) computing a caustics map, and 3rd pass) rendering a water surface using a shadow map and a caustics map. A water surface is reconstructed from a height field as a regular triangle mesh for each time step. Then, above three rendering passes are computed for each frame. We also use a reflection map for reflection on a water surface. Sides of a water tank are also displayed as users can feel the volume of water.

6.1. Shadows

Shadowing has a very important role to tell users a sense of distance. Without shadowing, objects look floating in the air. We use a shadow mapping technique cooperating with a framebuffer object (FBO). FBO is a part of OpenGL 2.0 standard, and it can be used for offline rendering. It is a faster method than reading pixels from a frame buffer because FBO directly renders to a texture. We implemented a straightforward approach that every pixel of a shadow map stores the distance from the light source to the closest object. It must be noted that every distance of a shadow map is normalized in the first rendering pass. Therefore, in the third rendering pass, the distance from the 3D point of a rendering pixel to the light source in a view coordinate system must be also normalized to compare with the normalized distance stored in a shadow map.

6.2. Caustics

To render caustics, we adopt the algorithm proposed by Nishita et al. [NN94]. It considers illumination volume and accumulates light intensity of cross sections intersected by scan lines. We only consider caustics on the bottom of a water surface and ignore caustics casting on obstacles. As mentioned in Iwasaki et al. [IDN01], it is very simplified because we do not take into account shafts of light either. The algorithm works as follows for every triangle composing a water surface. First, the three refracted directions are computed at three vertices for a triangle. Second, the three intersection points between the three refracted rays and the bottom of the water are computed. Third, the intensity for each intersection points is computed taking into account the length of the refracted ray and the form factor of the projected triangle on the bottom of the water. Finally, the projected triangle is rendered with the intensity computed above. Here FBO is used again to store a caustics map for each time step.

7. Result

Figure 11 shows captured images from a real time demonstration. The number of computational grid is 83×83, which outmost grid points are used for boundary conditions. The number of triangles is $80 \times 80 \times 2$. Our software achieves rendering 15 frames per second on a desktop PC (Core 2 Duo 2.4GHz, Geforce8800 GTX) and a note PC (Core 2 Duo 2.5GHz, GeForce8400M GS). Detailed timings per frame are shown in Table 1.

In our application, the most time consuming parts of simulation and caustics are computed in CPU. They will be implemented in GPU. We must mention that the computation of caustics requires calculating the form factor for each triangle, which is the ratio of the area of a triangle on a wave surface to the projected triangle on the bottom of the water. It may be difficult to implement in a fragment program which cannot allow access to the information of a projected triangle.

8. Conclusion and Future Work

We have proposed an application of photo realistic water surface interaction exploiting Augmented Reality. Although

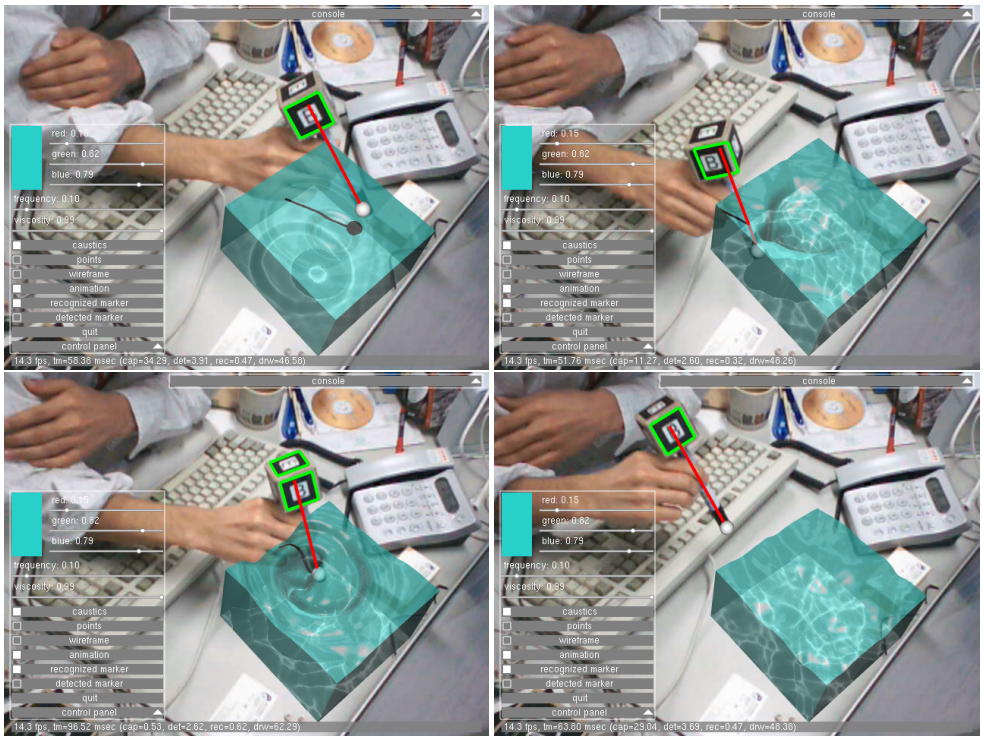

Figure 11: *Captured images from a real time demonstration. Green squares represent detected markers.*

cap	det	rec	drw	sim	caustics	shadow	total
12.00	2.84	0.34	3.59	29.09	17.28	1.05	66.31

Table 1: *Timings per frame. The unit is mili seconds. Columns show times of video capture, marker detection, maker recognition, OpenGL drawing, wave simulation, caustics computation, shadow computation and total time from left to right.*

many algorithms to solve each problem for fluid simulation, photo realistic rendering, and interaction techniques have been proposed so far, few papers have explored the reality when those algorithms are combined. The Reality of Virtual Reality is affected by many research areas. Therefore, individual researches must be combined to achieve the extreme goal of Virtual Reality which a synthesized world becomes indistinguishable to the real. In this paper, we have developed an application of the novel and clever combination of algorithms focusing on height field wave simulation, shadows and caustics rendering, and a 6DOF manipulator. In our system, a user can touch a virtual water surface using a real pen as usual in the real world. Even though our fluid simulation and the proposed force model are very simple, quite interesting photo realistic interaction has been confirmed using a clever combination of simplified existing algorithms. Our contribution is to attract an attention how a combination of carefully selected simple algorithms makes a great deal of reality shown in the videos of our real time demonstrations. In conclusion, it is the most important to reach the extreme goal of Virtual Reality.

Our application based on a marker based Augmented Reality technique. So there are a few problems inheriting from it. One of the such problems is that a marker of an AR pen must be located in a camera view. Sometimes, even though the tip of an AR pen is inside of a camera view, all markers of an AR pen is outside of a camera view. Consequently, the user loses the control of an AR pen. Similarly, if an AR pen and a user's hand hide a marker for a water surface, a water tank is ill posed. Another problem is detection of markers, whose accuracy depends on a lighting condition in a room. Although markers are sprayed by matte coating, strong light sometimes makes markers be white by reflection. However, such problems exist, it is very worthful to get a six degree of freedom user interface by just preparing a web camera.

There are many possible directions of future work. Our AR pen can be used as modeling and sculpting tools in graphics applications. Marker less tracking AR techniques [AMR*07, WP09] might bring a great impression that the

user can touch a virtual water surface by their own hands. We believe that strong collaboration among simulation, lighting, and AR techniques gives applications a great deal of reality.

9. Acknowledgements

We would like to thank reviewers for their thoughtful comments. This work was supported by KAKENHI (a Grant-in-Aid for Young Scientists (B)) (No. 21700126).

References

[AMR*07] ALLARD J., MENIER C., RAFFIN B., BOYER E., FAURE F.: Grimage: Markerless 3d interactions. In *Sigggraph 2007 (Emerging Technologies)* (2007).

[AR06] ALLARD J., RAFFIN B.: Distributed physical based simulations for large vr applications. In *IEEE VR, March 2006* (2006).

[ART] Artoolkit. In *http://www.hitl.washington.edu/artoolkit/*.

[BMF07] BRIDSON R., MÜLLER-FISCHER M.: Fluid simulation. In *SIGGRAPH 2007 Course Notes* (2007).

[Fia05] FIALA M.: Artag, a fiducial marker system using digital techniques. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (2005), pp. 590–596.

[IAY*06] IMURA M., AMADA T., YASUMURO Y., MANABE Y., CHIHARA K.: Synthetic representation of virtual fluid for mixed reality. In *Proceedings of 8th International Conference on Virtual Reality* (2006), pp. 135–142.

[IDN01] IWASAKI K., DOBASHI Y., NISHITA T.: Efficient rendering of optical effects within water using graphics hardware. In *Proc. of Pacific Graphics 2001* (2001), pp. 374–383.

[IDN03] IWASAKI K., DOBASHI Y., NISHITA T.: A fast rendering method for refractive and reflective caustics due to water surfaces. *Computer Graphics Forum 23*, 3 (2003), 601–609.

[IYMC06] IMURA M., YASUMURO Y., MANABE Y., CHIHARA K.: Fountain designer: Control virtual water as you like. In *ISMAR 2006 Demo* (2006).

[KB99] KATO H., BILLINGHURST M.: Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)* (1999), pp. 85–94.

[KBP*00] KATO H., BILLINGHURST M., POUPYREV I., IMAMOTO K., TACHIBANA K.: Virtual object manipulation on a table-top ar environment. In *Proceedings of the International Symposium on Augmented Reality (ISAR 2000)* (2000), pp. 111–119.

[NN94] NISHITA T., NAKAMAE E.: Method of displaying optical effects within water using accumulation-buffer. In *Proc. of SIGGRAPH'94* (1994), pp. 373–380.

[Tes04a] TESSENDORF J.: Interactive water surfaces. In *Game Programming Gems 4* (2004), Kirmse A., (Ed.), Charles River Media.

[Tes04b] TESSENDORF J.: Simulating ocean surface. In *SIGGRAPH 2004 Course Notes* (2004).

[WP09] WANG R. Y., POPOVIĆ J.: Real-time hand-tracking with a color glove. *ACM Transactions on Graphics 28*, 3 (2009).