Realistic Visual Simulation of Water Effects in Response to Human Motion using a Depth Camera

Jong-Hyun Kim[1], Jung Lee[3], Chang-Hun Kim[2] and Sun-Jeong Kim[*3]
[1] Department of Software Application, Kangnam University, Republic of Korea
[e-mail: gogogoscv@gmail.com]
[2] Computer Science and Engineering, Korea University, Republic of Korea
[e-mail: chkim@korea.ac.kr]
[3] Department of Convergence Software, Hallym University, Republic of Korea
[e-mail: airjung@hallym.ac.kr, sunkim@hallym.ac.kr]
*Corresponding author: Sun-Jeong Kim

Received August 15, 2016; revised November 28, 2016; accepted January 6, 2017; published February 28, 2017

Abstract

In this study, we propose a new method for simulating water responding to human motion. Motion data obtained from motion-capture devices are represented as a jointed skeleton, which interacts with the velocity field in the water simulation. To integrate the motion data into the water simulation space, it is necessary to establish a mapping relationship between two fields with different properties. However, there can be severe numerical instability if the mapping breaks down, with the realism of the human–water interaction being adversely affected. To address this problem, our method extends the joint velocity mapped to each grid point to neighboring nodes. We refine these extended velocities to enable increased robustness in the water solver. Our experimental results demonstrate that water animation can be made to respond to human motions such as walking and jumping.

Keywords: Microsoft Kinect, Coupling of water and human motion, Human–water interaction, Particle-based fluids, Human motion

A preliminary version of this paper was presented at APIC-IST 2016, and was selected as an outstanding paper. This research was supported by Hallym University Research Fund (HRF-201409-012), supervised by the IITP (Institute for Information & Communications Technology Promotion) (IITP-2016-R7518-16-1028), and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2013R1A1A2011602, NRF-2014R1A2A2A01007143). This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) and Korea Creative Content Agency(KOCCA) grant funded by the Korea government(MCST) (R0132-15-1006, Developing the technology of open composable content editors for realistic media).

https://doi.org/10.3837/tiis.2017.02.021 **ISSN : 1976-7277**

1. Introduction

Video scenes increasingly include virtual avatar characters that can be manipulated via motion capture. To facilitate these manipulations, very expensive capture devices are usually required to extract suitable human motion data. Furthermore, labor-intensive postprocessing needs to be performed to refine the captured motion data. For example, Park and Hodgins [6] proposed a data-driven modeling method for capturing high-resolution motions that deform the skin and muscles. Hong et al. [3] proposed a data-driven segmentation method for resolving the twisted artifacts that occur in joints with complex topology, such as the shoulder joint. However, because these methods focused on capturing human motion in an accurate manner, their computational costs were too great to allow their use in interactive applications.

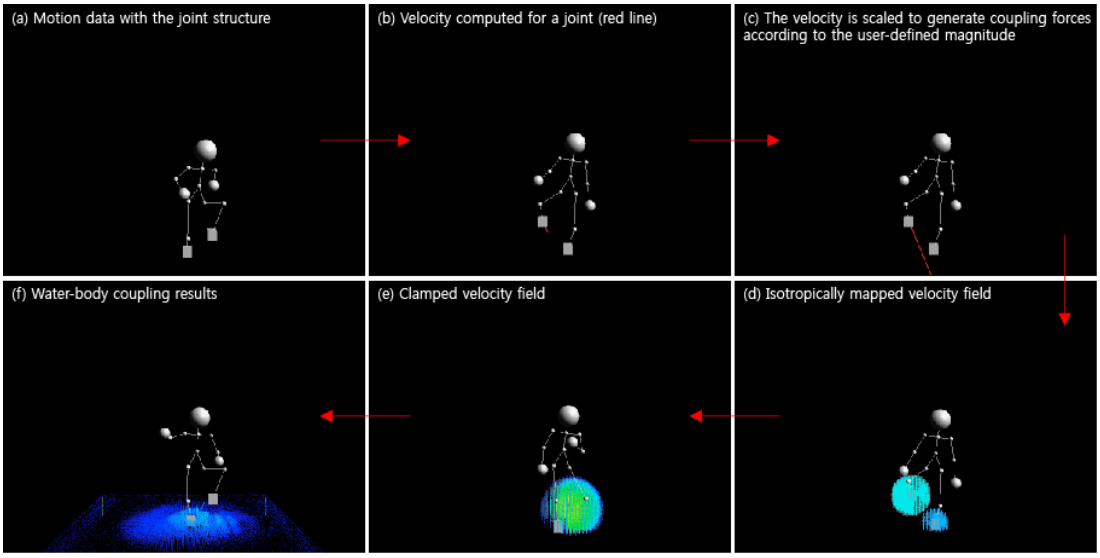


Fig. 1. Overview of our proposed method.

In addition, many real-time motion-capture applications have been proposed. Lange et al. [5] developed a Kinect-based game for rehabilitation and Tseng et al. [8] proposed a Kinect-based rehabilitation system for virtual scenes. These methods aimed to facilitate rehabilitation in the home or other places that might lack appropriate facilities.

Martinovikj and Ackovska [1] proposed a gesture-based presentation-control system for analyzing human motions, as an alternative to using a mouse and keyboard. This method analyzed hand gestures, with only two gestures being considered: swipe-left and swipe-right. Tam and Li [7] proposed a gesture recognition system using quick-response codes for mobile devices. In addition, visual simulation techniques that enable interaction with human motion data have been developed. In particular, Kwatra et al. [4] aimed to represent the swimming motions of an articulated body by addressing water–body coupling issues.

In this study, we focus on the response of water to the motion of an actor (see **Fig. 1**). The velocity of each joint is mapped onto the grid space and extended to compensate for the coupling forces. The robustness of the simulation is increased by combining a velocity

refinement method, which is derived from a smoothed particle hydrodynamics kernel, with the coupling process. An overview of our algorithm is shown in **Fig. 1**.

We propose a novel volumetric structure for coupling water with an actor. In particular, the main contributions of this study are as follows.

- Isotropic mapping of the grid velocity: the velocity of each joint is extended to its neighboring grid points to compensate for the coupling forces lost during the point-based velocity mapping.
- Clamped Gaussian refinement: a robust coupling result is obtained by adjusting the magnitude of the velocity for each grid point in inverse proportion to the distance from a joint.
- User controllability enhancement: a weight is assigned to control the intensity of the water's response to human motion. A variety of water motions can be obtained at different scales by controlling this weight.

2. Related Work

The first simulation of the fully three-dimensional Navier–Stokes equation for animating liquids [10] was based on the marker-and-cell method [11] from computational fluid dynamics. Foster and Metaxas [10] used explicit finite differencing for advection and viscosity, successive over-relaxation (SOR) for pressure projection and incompressibility, and massless marker particles for surface representation. Explicit integration methods were subsequently replaced by implicit methods [12] such as semi-Lagrangian advection and implicit viscosity integration, which greatly increased the numerical stability of fluid simulators both for liquid and gas, and making them easier to implement. Later [13], SOR was replaced by more efficient linear solvers, such as the conjugate gradient method, and the particle-based surface representation was reinforced by implicit level-set surfaces, which greatly improved the smoothness of liquid surfaces and their robustness under topological changes. This hybrid surface representation was enhanced by the particle level-set method [14], which has a much improved mass conservation.

While free-surface animation techniques for liquid animation have been extensively developed, for which both the environmental and the enclosed air are ignored, the dynamics of multiphase fluids have received less attention. Takahashi et al. [15] reported on a multiphase fluid simulator that could handle liquid and gas simultaneously but ignored the dynamic characteristics of liquid–gas interactions. On the other hand, Hong and Kim [16] mainly focused on buoyancy and surface tension in their animation of bubbles in liquids. Although their results showed the interesting characteristics of multiphase fluids containing bubbles, it is not clear that their heuristic implementation of surface tension would be generally useful. Furthermore, the effect of viscosity differences was not considered, in spite of the large influence of viscosity on bubble shape. Carlson et al. and Rasmussen et al. [17, 18] used a variational viscosity method to handle thermal changes of viscosity, but they ignored the large changes that can occur across interfaces. Song et al. [19] focused on the small-scale features of multiphase fluids. They demonstrated the characteristics of enclosed air and modeled surface tension using a continuum surface force model [20] that had been used widely in computational fluid dynamics but commented that surface-tension effects were not visually significant in their work. We believe that was because they had replaced small-scale features by undeformable particles instead of simulating them directly. Similarly, Greenwood and

House [21] used escaped particles within a particle level-set method to represent air bubbles. A breakthrough was made by Losasso et al. [22], who animated the crown phenomenon exhibited by milk droplets via accurate simulation of the surface tension of free surfaces. They could use a sufficiently large grid (512^3) that they did not lose small-scale details. However, neither surface tension in bubble surfaces nor viscosity was considered. Their fluid simulator was based on an octree data structure, which is also the basis of the work reported in this paper.

In computational physics, extensive studies have been undertaken to simulate multiphase fluids. There are several good surveys [23, 24, 25], and their references are also recommended. Although it is difficult to evaluate the techniques reported in these papers from the viewpoint of computer graphics, the work of Kang et al. [26] is the most compatible with the liquid-simulation techniques widely used in computer graphics, such as the particle level-set method [14]. The methods used by Kang et al. [26] are motivated by the ghost fluid method [27], and were developed using the variable-coefficient Poisson equation [28]. In computer graphics, this method has been used for the physics-based modeling of fire [29].

3. Underlying Water Simulation

The Navier–Stokes equation for an incompressible viscous fluid is

$$\boldsymbol{u}_t = -(\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla\cdot(\mu\nabla\boldsymbol{u}) - \frac{\nabla p}{\rho} + \boldsymbol{f}, \tag{1}$$

$$\nabla\cdot\boldsymbol{u} = 0, \tag{2}$$

where $\boldsymbol{u} = \{u, v, w\}$ is the velocity, ρ is the density and μ is the (kinematic) viscosity, which is the ratio of the absolute viscosity to the density. The term $\boldsymbol{f}$ can be used to add external forces such as gravity, buoyancy [30], surface-tension force [16, 19], and control forces [31, 32, 33, 34].

The numerical simulation of Equations 1 and 2 proceeds by updating the value of $\boldsymbol{u}$ at the nth time step, $\boldsymbol{u}^n$, to $\boldsymbol{u}^{n+1}$ during a finite interval Δt. Following Chorin's projection method [35], we discretize Equation 1 by splitting it into two equations with an intermediate status $\boldsymbol{u}^*$, giving

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -(\boldsymbol{u}^n\cdot\nabla)\boldsymbol{u}^n + \nabla\cdot(\mu\nabla\boldsymbol{u}^n) + \boldsymbol{f}, \tag{3}$$

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\frac{\nabla p}{\rho}. \tag{4}$$

To obtain $\boldsymbol{u}^*$ from $\boldsymbol{u}^n$, we compute the advection term $-(\boldsymbol{u}^n\cdot\nabla)\boldsymbol{u}^n$ using a semi-Lagrangian method [12], and the viscosity term $\nabla\cdot(\mu\nabla\boldsymbol{u}^n)$ using explicit finite differencing or an implicit variable-viscosity formulation [17].

The final step is determining $\boldsymbol{u}^{n+1}$ from $\boldsymbol{u}^*$. We can write the divergence of Equation 4 as a form of Poisson's equation, giving

$$\nabla^2 p = \frac{\rho}{\Delta t}\nabla\cdot\boldsymbol{u}^*, \tag{5}$$

because Equation 2 indicates that $\nabla\cdot\boldsymbol{u}^{n+1}$ should be zero. After the pressure profile is determined by solving Equation 5, we obtain the final velocity profile as

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^{*} - \frac{\Delta t}{\rho}\nabla p. \quad (6)$$

Buoyancy can be implemented in a simple way by exerting buoyant forces using a spatial constant ρ [30] or, more accurately, by allowing for the fact that $\rho = \rho(x)$ in the solution of Equation 4 [19]. Instead of exerting continuous surface-tension forces [19] using $\boldsymbol{f}$, we use the discontinuity of p in solving Equation 5 to model surface-tension effects, thereby obtaining subgrid accuracy. Viscosity changes across the interfaces are also considered in solving Equation 3.

Another issue to be considered is the interface tracking method. We use a signed distance function $\emptyset$ to represent (implicitly) the interfaces of two immiscible fluids, at least one of which is a liquid. The advection of $\emptyset$, driven by $\boldsymbol{u}$, can be described by the level-set equation

$$\emptyset_t + \boldsymbol{u} \cdot \nabla\emptyset = 0. \quad (7)$$

To solve Equation 7 numerically, we use the semi-Lagrangian particle level-set method [36].

4. Coupling Simulation

To combine the motion of a human being with the external forces in water, the velocity of each joint in the skeleton must be mapped onto the grid in the water simulation space. Occasionally, the response of the water may be expressed indistinctly, even if the velocity is mapped accurately. In addition to the magnitude of the human motion, the velocity of the joint is scaled freely according to a user-specified threshold to obtain a specific water-response level (see **Fig. 1c**). The threshold is reflected by the weight used in the equation that controls the velocity of each joint, namely

$$v_{t+\Delta t} = \left(\frac{p_{t+\Delta t} - p_t}{\Delta t}\right) w_0, \quad (8)$$

where p denotes the position of a joint, v denotes its velocity, and w_0 denotes the constant weight specified by the user. During this mapping process, each joint is represented as a single point, rather than a regional entity (see **Fig. 1a**), and the coupling force is limited locally around the mapped grid point (see **Fig. 1b**). To extend the influence of the velocity to its neighborhood, we map its velocity isotropically around the grid point (see **Fig. 1c**).

First, the velocity $u(x)$ of each grid point x is calculated using smoothed particle hydrodynamics, giving

$$u(x) = \frac{\sum_i m_i v_i W(p_i - x, h)}{\sum_i m_i W(p_i - x, h)}, \quad (9)$$

where h denotes the kernel radius, m_i denotes the mass of each joint i and $W(r, h)$ is used to compute the average quantities for the neighboring joints. That is,

$$W(r,h) = \begin{cases} 1 - \frac{\|r\|^2}{h^2} & 0 \le \|r\| \le h \\ 0 & \text{otherwise} \end{cases}. \quad (10)$$

After calculating the velocities, the velocity of each grid point is extended to the neighboring grid points inside a circle of a specific radius. The radius of the extension is determined as a proportion of the magnitude of the velocity (see **Fig. 2c**).

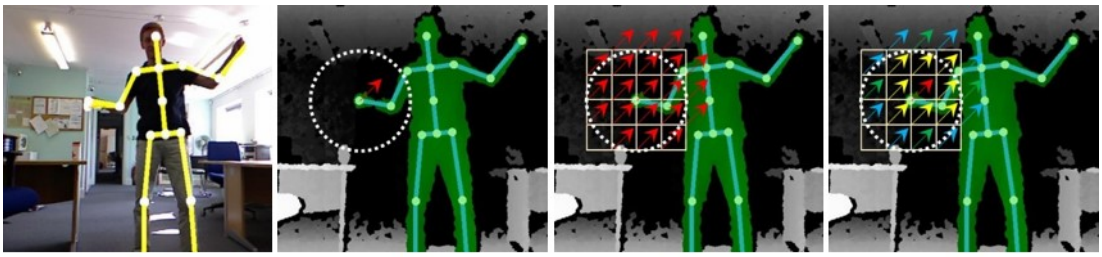


Fig. 2. Overview of our isotropic velocity-mapping method. (a): Motion skeleton and the captured velocity of a joint. (b): Velocity mapped onto a simulation grid point. (c): Isotropically mapped velocities around the grid point. (d): Clamped velocities, where the velocity of each grid is color-coded from red (high) to blue (low) according to the magnitude of the velocity.

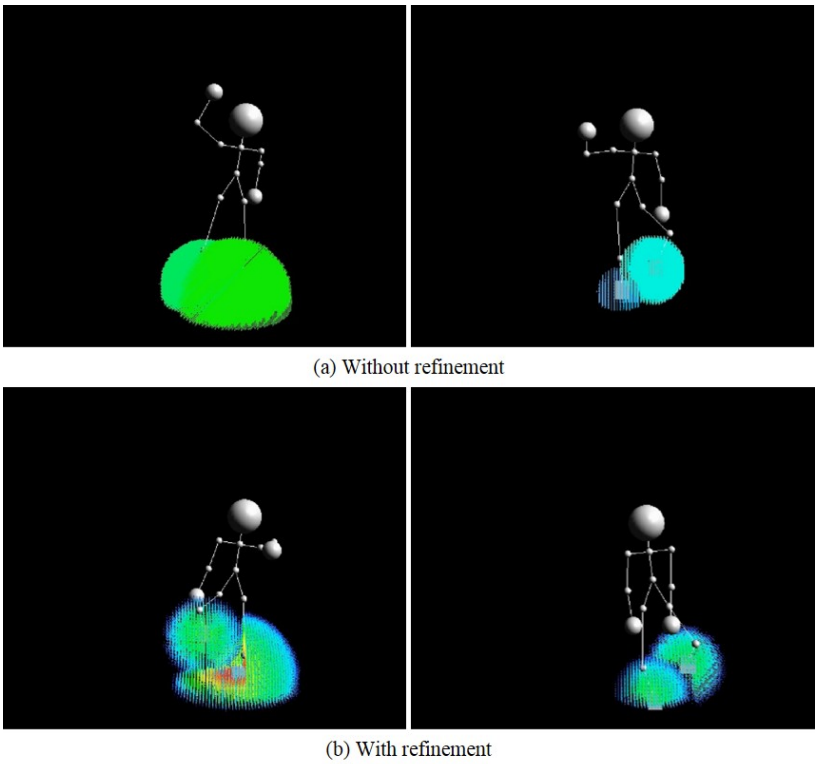
The isotropic velocity-mapping method compensates very well for the locally limited coupling effect, but the robustness of the simulation is reduced because the dramatically magnified velocities can generate numerically unstable motions compared with those in the neighborhood. To reduce this instability, we refine the velocities using a clamped Gaussian equation based on the weighted isotropic kernel (see **Fig. 2d**).

The results obtained after clamped Gaussian refinement are shown in **Fig. 3**, where each velocity field is represented as a spherical shape with a radius that varies according to the velocity of each joint and the magnitude of the velocity is represented in inverse proportion to the distance from the joint. The refined velocity field in **Fig. 3b** is much smoother than that shown in **Fig. 3a** and it generates numerically stable simulation results, even for noisy motion data. The refinement procedure is given by

$$\boldsymbol{u}^* = \underbrace{w_1\big((x-p)/h\big)\Big(1-\big((x-p)/h\big)^2\Big)^2}_{clamped\ Gaussian}\boldsymbol{u}, \quad (11)$$

where u^* denotes the refined velocity and w_1 denotes a Gaussian constant. The refined velocity field given by Equation 11 is inserted into the external forces of the fluid simulation framework and the Navier–Stokes equation can be rewritten as

$$\frac{\boldsymbol{u}^{n+1}-\boldsymbol{u}^n}{dt} = -(\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n + \nabla \cdot (\mu \nabla \boldsymbol{u}^n) - \frac{\nabla p}{\rho} + \frac{\boldsymbol{f}}{\rho} + \boldsymbol{u}^*, \quad (12)$$



(a) Without refinement

(b) With refinement

Fig. 3. Results obtained after clamped Gaussian refinement.

where $\boldsymbol{u}^n$ and $\boldsymbol{u}^{n+1}$ denote the velocity at the current time step and the next time step, respectively. In this study, a splitting scheme is used to solve each term in Equation 12 [2], with these terms being defined as

$$\hat{\boldsymbol{u}} = \boldsymbol{u}^n - dt(\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n, \tag{13}$$

$$\acute{\boldsymbol{u}} = \hat{\boldsymbol{u}} + \frac{dt}{\rho}\boldsymbol{f} + \boldsymbol{u}^*, \tag{14}$$

$$\tilde{\boldsymbol{u}} = \acute{\boldsymbol{u}} + dt\nabla \cdot (\mu \nabla \acute{\boldsymbol{u}}), \tag{15}$$

$$\boldsymbol{u}^{n+1} = \tilde{\boldsymbol{u}} - \frac{dt}{\rho}\nabla p. \tag{16}$$

Equation 12 can generate fluid animations that respond dynamically to human motion because of its divergence-free condition. **Fig. 4** shows the results obtained after coupling the velocity field of the water simulation and a human motion field.

5. Experimental Results and Discussion

We performed the simulations using an Intel i7-2600k 3.40 GHz CPU with 16 GB of RAM and an NVIDIA GeForce GTX 580 graphics card. To show the robustness of our algorithm, we analyzed our method under various scenarios, using the configurations summarized in

Table 1. As seen from this table, the user can create these various results by changing the values for w_0 and w_1.

(a) (b) (c)

Fig. 4. Results obtained for the interactions between human motion and falling water.

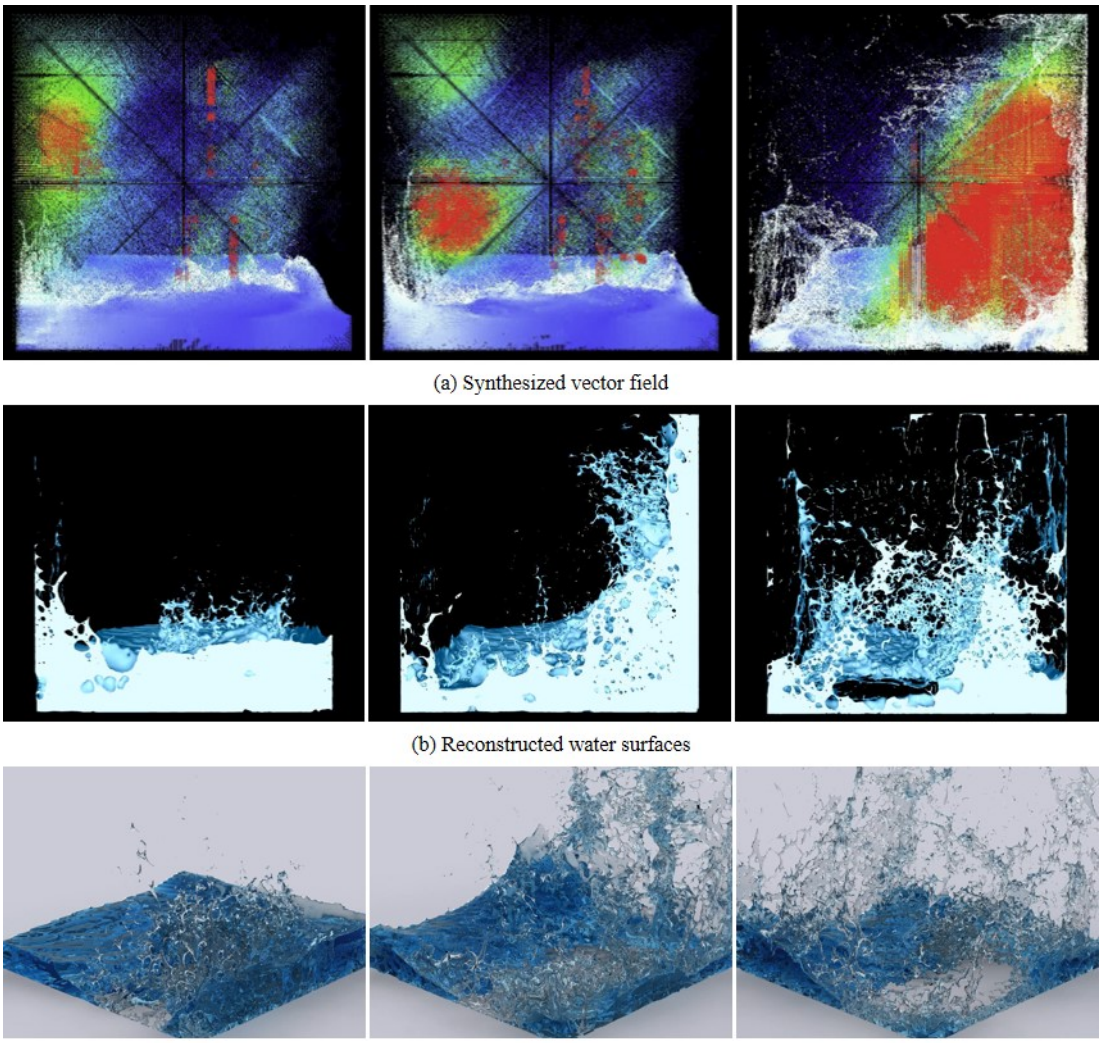
The falling-water scene in **Fig. 4** shows very well the interactive features of our method. In this scene, the time step was set to 0.001 s and 250,000 water particles were used. In these results, our coupling method shows a natural interaction with the opened arms of the human character.

Fig. 5 simulates water responding to a jumping motion by a human character. In the real world, the water flow is changed even by a small jump. However, previous methods usually fail to show the interaction for these small motions. As mentioned earlier, this is a significant issue for interactive applications. **Fig. 5** shows that the proposed method can generate the water waves resulting from even small jumping motions.

(a) (b) (c)

Fig. 5. Results obtained for the interactions between jumping motions and water.

Fig. 6a shows the results obtained after coupling water particles generated using a fluid-implicit-particle method [9] with motion data. As described earlier, the coupling force was varied in proportion to the magnitude of the human motion to generate realistic water interactions. **Fig. 6b** shows the realistic water surfaces reconstructed by the marching-cubes algorithm [37], as well as splash effects. **Fig. 6c** shows the high-quality rendering results achieved by ray tracing.

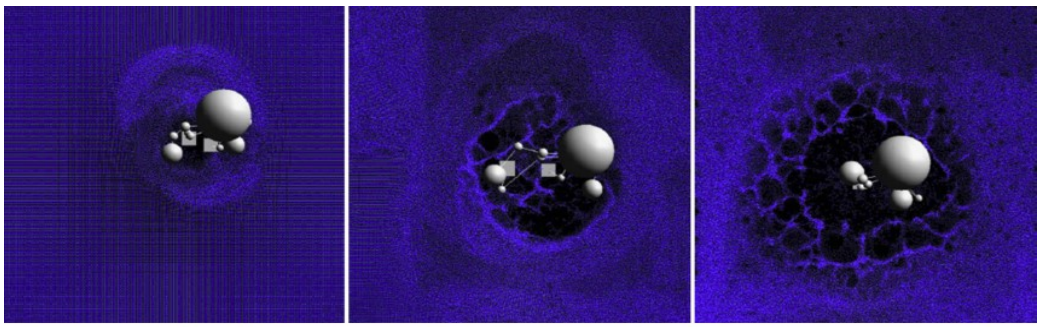


(a) Synthesized vector field

(b) Reconstructed water surfaces

(c) High quality rendering (only water)

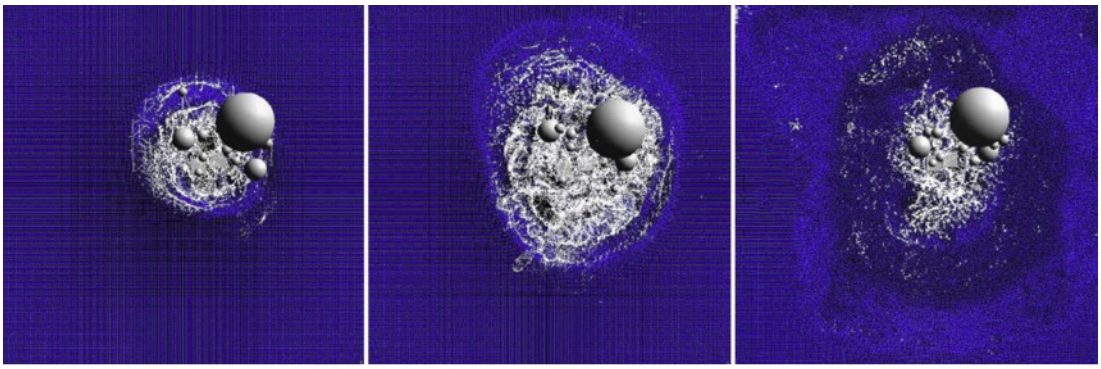
Fig. 6. Simulation results obtained for water interacting with human motion. The color denotes the magnitude of the velocity (red: high, blue: low).



(a) 40 frame (b) 216 frame (c) 318 frame

Fig. 7. Results obtained (top view) after coupling water particles with human motion data. This scene shows the movement of water interacting with the feet of an avatar. As the jumping becomes stronger, the disturbance of the water becomes more intense.

Fig. 7 shows a top view of results obtained after coupling water particles with human motion data. **Fig. 8** shows some of the secondary effects generated, i.e., water splashes. Splash particles are generated when the coupling force exceeds the user-specified threshold and more detailed water components are included. Using the proposed method, the water interacting with human motion expresses splash effects naturally.



(a) 37 frame (b) 62 frame (c) 119 frame

Fig. 8. Water splashes produced during interaction with human motion (blue: water particles, white: splash particles). Splash effects are represented by the water interacting with the motion of the avatar.

6. Conclusion

In this study, we have proposed a water–motion coupling method for generating a detailed animation of water interacting with human motion. Because general motion data are represented as points, it is difficult to couple them with a conventional water solver. Our method involves calculating the velocity field for the motion data and then applying clamped Gaussian refinement. Our experiments show that this generates effective and realistic water animation, even with noisy motion data and at low grid resolutions. In future research, we will track human motion more accurately by modeling anisotropic joint motion. We also aim to design a more accurate coupling solver that considers continuous collision detection among adjacent joints.

Table 1. Parameter values for our example scenes.

Figure	# of water particles	# of splash particles	Δt	w_0	w_1
4	250,000	-	0.001	3.5	0.9
5	250,000	-	0.001	3.5	0.9
6	400,000	-	0.001	3.5	0.9
7	152,000	-	0.001	3.5	0.9
8	152,000	43,000	0.001	3.5	0.9

References

[1] Ackovska, D.M.N, "Gesture recognition solution for presentation control," in *Proc. of 10th Conference for Informatics and Information Technology*, 2013. Article (CrossRef Link).

[2] Bridson, R., Fedkiw, R. and Müller-Fischer, M., "Fluid simulation: SIGGRAPH 2006 Course Notes," *ACM SIGGRAPH Courses*, 2007. Article (CrossRef Link).

[3] Hong, Q. Y., Park, S. I. and Hodgins, J. K., "A data-driven segmentation for the shoulder complex," *Computer Graphics Forum*, pp. 537–544, 2010. Article (CrossRef Link).
[4] Kwatra, N., Wojtan, C., Carlson, M., Essa, I. A., Mucha, P. J. and Turk, G., "Fluid simulation with articulated bodies," *IEEE Transactions on Visualization and Computer Graphics*, pp. 70–80, 2010. Article (CrossRef Link).
[5] Lange, B., and Koenig, S., McConnell, E., Chang, C.Y., Juang, R., Suma, E., Bolas, M. and Rizzo, A., "Interactive game-based rehabilitation using the Microsoft Kinect," in *Proc. of IEEE Virtual Reality Workshops*, pp. 171–172, 2012. Article (CrossRef Link).
[6] Park, S. I. and Hodgins, J. K., "Data-driven modeling of skin and muscle deformation," *ACM SIGGRAPH*, pp. 96:1–96:6, 2008. Article (CrossRef Link).
[7] Tam, V. and Li, L.-S., "Integrating the Kinect camera, gesture recognition and mobile devices for interactive discussion," in *Proc. of IEEE International Conference on Teaching, Assessment and Learning for Engineering*, pp. H4C–11, 2012. Article (CrossRef Link).
[8] Tseng, C.M., Lai, C.L., Erdenetsogt, D. and Chen, Y. F., "A Microsoft Kinect-based virtual rehabilitation system," in *Proc. of International Symposium on Computer, Consumer and Control*, pp. 934–937, 2014. Article (CrossRef Link).
[9] Zhu, Y. and Bridson, R., "Animating sand as a fluid," *ACM SIGGRAPH*, pp. 965–972, 2005. Article (CrossRef Link).
[10] Foster, N. and Metaxas, D., "Realistic animation of liquids," *Graph Models Image Process* 58(5):471–483, 1996. Article (CrossRef Link).
[11] Harlow, F. H. and Welch, J.E., "Numerical calculation of time-dependent viscous incompressible flow of fluid with free surfaces," *Phys. Fluids* 8:2182–2189, 1965. Article (CrossRef Link).
[12] Stam, J., "Stable fluids," *ACM SIGGRAPH*, pp. 121–128, 1999. Article (CrossRef Link).
[13] Foster, N. and Fedkiw, R., "Practical animation of liquids," *ACM SIGGRAPH*, pp. 23–30, 2001. Article (CrossRef Link).
[14] Enright, D., Marschner, S. and Fedkiw, R., "Animation and rendering of complex water surfaces," *ACM SIGGRAPH* 2002, ACM Transactions on Graphics (TOG), vol. 21(3), pp. 736–744, July 2002. Article (CrossRef Link).
[15] Takahashi, T., Fujii, H., Kunimatsu, A., Hiwada, K., Saito, T., Tanaka, K. and Ueki, H., "Realistic animation of fluid with splash and foam," in *Proc. of Computer Graphics Forum (Eurographics 2003)*, vol. 22(3), pp. 391–400, Sept 2003. Article (CrossRef Link).
[16] Hong, J.-M. and Kim, C.-H., "Animation of bubbles in liquid," *Computer Graphics Forum (Eurographics 2003)*, vol. 22(3), pp. 253–262, Sept 2003. Article (CrossRef Link).
[17] Carlson, M., Mucha, P.J., van Horn II, R.B., and Turk, G., "Melting and flowing," *ACM SIGGRAPH* /Eurographics *Symposium on Computer Animation*, pp. 167–174, 2002. Article (CrossRef Link).
[18] Rasmussen, N., Enright, D., Nguyen, D., Marino, S., Sumner, N., Geiger, W., Hoon, S., Fedkiw, R., "Directable photorealistic liquids," in *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 193–202, 2004. Article (CrossRef Link).
[19] Song, O.-Y., Shin, H.-C. and Ko, H.-S., "Stable but non-dissipative water," *ACM Transactions on Graphics*, 24(1):81–97, 2005. Article (CrossRef Link).
[20] Brackbill, J. U., Kothe, D. B. and Zemach, C., "A continuum method for modeling surface tension," *Journal of Computational Physics*, 100(2):335–354, 1992. Article (CrossRef Link).
[21] Greenwood, S., House, D., "Better with bubbles: enhancing the visual realism of simulated fluid," in *Proc. of* In: *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 287–296, 2004. Article (CrossRef Link).
[22] Losasso, F., Gibou, F. and Fedkiw, R., "Simulating water and smoke with an octree data structure," *ACM Transactions on Graphics*, vol. 23(3), pp. 457–462, Aug 2004. Article (CrossRef Link).
[23] de Sousa, F., Mangiavacchi, N., Nonato, L., Castelo, A., Tome, M., Ferreira, V., Cuminato, J. and McKee, S., "A front-tracking/front-capturing method for the simulation of 3D multi-fluid flows with free surfaces," *Journal of Computational Physics*, 198(2):469–499, 2004. Article (CrossRef Link).

[24] Sussman, M., "A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles," *Journal of Computational Physics*, 187(1):110–136, 2003. Article (CrossRef Link).
[25] Tryggvason, G., Bunner, B., Esmaeeli, A., Juric, D., Al-Rawashi, N., Tauber, W., Han, J., Nas, S. and Jan, Y.-J., "A front-tracking method for the computations of multiphase flow," *Journal of Computational Physics*, 169(2):708–759, 2001. Article (CrossRef Link).
[26] Kang, M., Fedkiw, R. P. and Liu, X.-D., "A boundary condition capturing method for multiphase incompressible flow," *Journal of Scientific Computing*, 15(3):323–360, 2000. Article (CrossRef Link).
[27] Fedkiw, R., Aslam, T., Merriman, B. and Osher, S., "A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method)," *Journal of Computational Physics*, 152(2):457–492, 1999. Article (CrossRef Link).
[28] Liu, X.-D., Fedkiw, R. P. and Kang, M.-J., "A boundary condition capturing method for Poisson's equation on irregular domain," *Journal of Computational Physics*, 172(1):71–98, 2000. Article (CrossRef Link).
[29] Nguyen, D., Fedkiw, R. and Jensen, H., "Physically based modeling and animation of fire," *ACM SIGGRAPH*, vol. 21(3), pp. 721–728, July 2002. Article (CrossRef Link).
[30] Fedkiw, R., Stam, J. and Jensen, H. W., "Visual simulation of smoke," *ACM SIGGRAPH*, 15–22, 2001. Article (CrossRef Link).
[31] Hong, J.-M. and Kim, C.-H., "Discontinuous fluids," *ACM Transactions on Graphics*, 24(3):915–920, 2005. Article (CrossRef Link).
[32] Hong, J.-M. and Kim, C.-H., "Controlling fluid animation with geometric potential," *Computer Animation and Virtual Worlds* 15(3–4):147–157, 2004. Article (CrossRef Link).
[33] McNamara, A., Treuille, A., Popović, Z. and Stam, J., "Fluid control using the adjoint method," in *Proc. of ACM SIGGRAPH 2004, ACM Transactions on Graphics*, vol. 23(3), pp. 449–456, Aug 2004. Article (CrossRef Link).
[34] Treuille, A., McNamara, A., Popović, Z. and Stam, J., "Keyframe control of smoke simulations," in *Proc. of ACM SIGGRAPH 2003, ACM Transactions on Graphics*, vol. 22(3), pp. 716–723, July 2003. Article (CrossRef Link).
[35] Chorin, A. J., "A numerical method for solving incompressible viscous flow problems," *Journal of Computational Physics*, 135(2):118–125, 1997. Article (CrossRef Link).
[36] Enright, D., Losasso, F. and Fedkiw, R., "A fast and accurate semi-Lagrangian particle level set method," *Computer &Structures*, 83(6–7):479–490, 2005. Article (CrossRef Link).
[37] Lorensen, W. E. and Cline, H. E., Marching Cubes: "A high-resolution 3D surface construction algorithm," *ACM SIGGRAPH*, pp. 163-169, 1987. Article (CrossRef Link).

Jong-Hyun Kim received the B.A. degree in the department of digital contents at Sejong University in 2008. He received M.S. and Ph.D. degrees in the department of computer science and engineering at Korea University, in 2010 and 2016. He is an assistant professor in the department of software application in Kangnam University. His current research interests include fluid animation and virtual reality.

Jung Lee is an assistant professor in the department of convergence software in Hallym University. His current research interests include augmented/virtual reality, fluid animation and computer graphics.

Chang-Hun Kim received the BA degree in economics from Korea University in 1979. He received the PhD degree from the department of electronics and information science, Tsukuba University, Japan, in 1993. He is a professor in the department of computer science and engineering at Korea University. After graduation, he joined the Korea Advanced Institute of Science and Technology (KAIST) as a research scientist, where he was involved in many national research projects in the area of Computer Aided Design and Geometric Modeling. During 1993-1995, he headed the Human Interface and Graphics Laboratory for the System Engineering Research Institute (SERI). His current research interests include fluid animation and mesh processing. He is a member of the IEEE Computer Society and the ACM.

Sun-Jeong Kim is a professor in the department of convergence software in Hallym University. Her research interests include geometric modeling, scientific visualization, virtual reality and augmented reality.