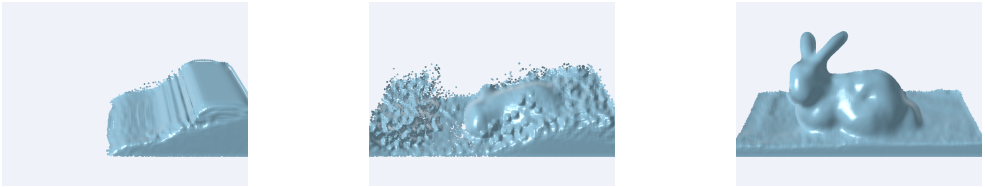
芸術科学会論文誌 Vol. 13, No. 3, pp. 169 – 176

Interactive fluid control by shape matching

Jun Yoshino[1)] **Reiji Tsuruno**[2)]

1) Graduate School of Design, Kyushu University
2) Faculty of Design, Kyushu University
1) yoshino (at) verygood.aid.design.kyushu-u.ac.jp 2) tsuruno (at) design.kyushu-u.ac.jp



(a) (b) (c)

Fig 1: Controlling fluid animation using a technique presented in this paper

Abstract

In this paper, we propose an interactive method to control fluid behavior by using a target shape. To achieve the desired motion, we define additional forces using control particles that employ the Navier–Stokes equation. Control particles are generated from a fluid area chosen by the user, and their behavior is sequentially determined using a shape-matching method referring to the target shape. Our method enables interactive simulation of fluid animation that is gradually deformed to the target shape following physical laws.

本論文は NICOGRAPH International 2013 での発表内容 [1] を芸術科学会論文誌に投稿するものである.

1 Introduction

In recent years, many movies have included water, smoke, and fire scenes generated by computer graphics. As it is difficult for animators to manually generate such fluid animations, they use computer simulations and automatically create these motions according to physical laws. Various physically based fluid-simulation models have been developed in computer graphics [2, 3, 4], and realistic animations can be achieved using these methods.

While fluid simulation makes animation physically precise, animators want to control the shape and motion of fluid to match the image they have in their mind. It is possible to control an animation generated from a computer simulation; however, this involves precise setting of conditions and regulating many parameters, which heavily burdens an animator trying to achieve the desired motion. Thus, controlling fluid shapes in computer simulations remains a problem.

A number of studies have addressed this problem and developed methods that enable the control of fluid simulations. However, existing methods are not suitable for the interactive control of fluids because of the high computational cost involved and dependency on precomputation. In this paper, we propose a novel technique that controls fluid animation using a target shape that can be interactively changed.

The key contributions of our paper are as follows:

- **Interactivity**: Our control framework runs at an interactive frame rate without dependency on precomputed data.
- **Seamless**: When users change the target shape, our technique smoothly interpolates between the current shape and the target shape by shape matching.
- **Easy to embed**: Our control forces are separable from fluid dynamics. Thus, it is easy to introduce our method into an existing fluid simulator.

2 Related Work

Foster and Metaxas were among the first to discuss fluid control[5]. They developed an embedded controller for pressure and velocity to control the flow and fluid surface. This idea was then extended by Foster and Fedkiw[6]. Shi and Yu proposed a method that controls liquids with a free surface[7]. However, because they used a proportional derivative controller, their method had to be carefully tuned. Treuille et al. presented keyframe animation of smoke[8], and they computed virtual wind force by solving a nonlinear optimization problem. The efficiency of this technique is improved by the adjoint method[9], but it is still computationally expensive. Hong and Kim demonstrated the control of smoke using a precomputed potential field[10]. They defined the force field as the gradient of the potential field and effectively generated fluid animations. Thürey et al. introduced control particles to guide the fluid flow[11]. In their method, the behavior of control particles toward the target shape is precomputed. More recently, several studies using lower-resolution simulations have been presented [12, 13]. However, their purpose is to effectively generate a natural controlled flow, which is different from our aim.

3 Our Method

Before describing our method in detail, we present an overview of our fluid control framework. In the proposed method, we first generate a basic fluid animation by means of a physically based fluid simulation. Next, control forces are applied to reflect the user's desired shape. These forces are defined by control particles, which Thürey et al. proposed in

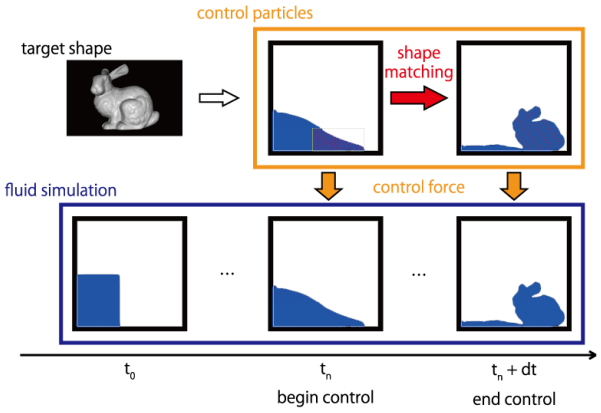


Fig 2: Overview of our framework

[11]. The key difference between previous studies and ours is that we determine the behavior of control particles at each simulation frame. We calculate this behavior by a shape-matching algorithm[14]. Figure 2 shows this flow with a 2D example.

3.1 Fluid Simulation

The fluid we treat is governed by the incompressible Navier–Stokes equation. It is usually written as follows:

$$\rho(\frac{\partial \boldsymbol{v}}{\partial t} + (\boldsymbol{v}\cdot\nabla)\boldsymbol{v}) = -\nabla p + \mu\nabla^2\boldsymbol{v} + \boldsymbol{f} \quad (1)$$

$$\nabla\cdot\boldsymbol{v} = 0 \quad (2)$$

where ρ is the density, $\boldsymbol{v}$ is the velocity, p is the pressure, μ is the viscosity, and $\boldsymbol{f}$ is the external force. In our technique, we solve the Navier–Stokes equation based on smoothed-particle hydrodynamics (SPH). The fluid is approximated with particles in this technique. The fluid property at an arbitrary position $\mathbf{x}$ is computed from the neighborhood particles:

$$A(\boldsymbol{x}) = \sum_j A_j V_j W(\boldsymbol{x}_j) \quad (3)$$

where A_j, V_j, and $\boldsymbol{x}_j$ are the property, volume and position of the jth particle, respectively. The function $W(\boldsymbol{x})$ is a normalized smoothing kernel, and our implementation follows the framework presented by Müller et al.[15].

3.2 Control Particles

Control particles introduce additional forces to neighborhood fluid elements: attraction force ($\boldsymbol{f}_a$) and velocity force ($\boldsymbol{f}_v$). These two forces are appended to the Navier–Stokes equation as external forces.

Attraction force pulls neighbors toward the control particle's position (Figure 3). This force allows us to deform fluid to a desired shape. The attraction force exerted on a fluid element e is

$$\boldsymbol{f}_a(e) = w_a \sum_i s_i \frac{\boldsymbol{p}_i - \boldsymbol{x}_e}{|\boldsymbol{p}_i - \boldsymbol{x}_e|} W(|\boldsymbol{p}_i - \boldsymbol{x}_e|) \quad (4)$$

$$s_i = 1 - min(1, \sum_e V_e W(|\boldsymbol{p}_i - \boldsymbol{x}_e|)) \quad (5)$$

where w_a is the global constant, s_i is the scale factor shown in eq 5, $\boldsymbol{p}_i$ is the ith control particle's position, $\boldsymbol{x}_e$ is the fluid element e's position, V_e is the volume of the fluid element e,

Fig 3: Attraction force

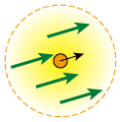


Fig 4: Velocity force

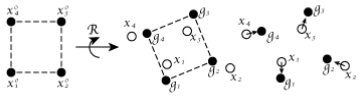


Fig 5: Algorithm for shape matching

and W is the poly6 kernel function—the same as SPH's pressure term.

Velocity force modifies fluid flow following the particle's velocity (Figure 4). The fluid is smoothly controlled owing to this force. Similar to the attraction force, the velocity force is defined as

$$\boldsymbol{f}_v(e) = w_v \sum_i (\boldsymbol{v}_i - \boldsymbol{v}_e) W(\boldsymbol{p}_i - \boldsymbol{x}_e) \qquad (6)$$

where w_v is a global constant, $\boldsymbol{v}_i$ is the ith control particle's velocity, and $\boldsymbol{v}_e$ is the fluid element e's velocity.

3.3 Shape Matching

Shape matching, which was proposed by Müller et al.[14], is an algorithm for simulating deformable objects. The basic idea behind this algorithm is that deformed points are pulled toward goal positions (Figure 5).

$$\Delta \boldsymbol{v}_i = \alpha \frac{\boldsymbol{g}_i - \boldsymbol{x}_i}{\Delta t} \qquad (7)$$

where $\boldsymbol{v}_i, \boldsymbol{g}_i, and \boldsymbol{x}_i$ is the ith point of velocity, goal position, and current position, respectively, and $\alpha \in [0, 1]$ is the stiffness of the object. The goal positions are computed by optimal rigid transformations of an original shape.

$$\boldsymbol{g}_i = R(\boldsymbol{x}_i^0 - \boldsymbol{x}_{cm}^0) + \boldsymbol{x}_{cm} \qquad (8)$$

where $\boldsymbol{x}^0$ is the position of the original shape, and $\boldsymbol{x}_{cm}^0$ and $\boldsymbol{x}_{cm}$ are the center of mass of the original and actual shapes, respectively. R is the rotation matrix, which is obtained from the polar decomposition of the optimal linear transformation A. Using relative locations $\boldsymbol{p}_i = \boldsymbol{x} - \boldsymbol{x}_{cm}$ and $\boldsymbol{q}_i = \boldsymbol{x}^0 - \boldsymbol{x}_{cm}^0$, A is described as

$$A = (\sum_i m_i \boldsymbol{p}_i \boldsymbol{q}_i^T)(\sum_i m_i \boldsymbol{q}_i \boldsymbol{q}_i^T)^{-1} \qquad (9)$$

where m_i is mass of the ith point.

In this study, we use a cluster-based deformation to generate a more flexible motion. This increases the computational load; hence, it is desirable to use an effective solver such as [16] or [17].

3.4 Fluid Control

In response to the user's operations, control particles are interactively generated. We define a reference particle that is created from the voxelized target shape (Figure 6). Control particles are placed according to these and deformed linearly to fill the user-specified fluid area (Figure 7). Ideally it is desirable to place these at random positions in the fluid area, but this simple approach is sufficient for our purpose.

In addition, control particles receive feedback force from the fluid flow in neighbors. This enables us to generate a more natural fluid animation. Because the velocity of the fluid particles

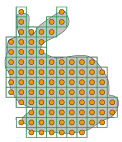


Fig 6: Reference particles: Each particle is placed at the center of a voxel

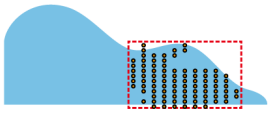


Fig 7: Initial placement of control particles

is gradually decaying, the power of the feedback becomes zero in the end. As a result, control particles converge toward the target shape by shape matching.

4 Results and Discussion

We have demonstrated our system by applying it to a number of target shapes. Our results were rendered in real time using the techniques presented in [18].

Figure 10 shows the results of applying our control method, and Figure 11 is the same scene with control particles visualized. In Figure 10(a), control particles were generated by user interaction. From Figure 10(b) to 10(d), the fluid was approximately deformed toward the target shape, gradually revealing its details. Following the current fluid flow, control particles were advected toward the left side (shown from Figure 11(b) to Figure 11(d)). Therefore, our technique bears more resemblance to natural fluid behavior.

Figure 9 shows another example generated by our system. In each of these, a target shape has been adequately reflected with our technique. In Figure 12, we demonstrated interactive changes in the target shape. When the target shape was updated, the fluid shape was collapsed at once following physical laws. Then, a new target shape was reflected. By shape matching, our method generated interpolated animation between two target shapes without precomputing.

We show the performance of our proposed method in Figure 8. Our simulation was implemented in C/C++ and has achieved significant acceleration by employing OpenMP directives. All our experiments were run on a Windows machine with Intel Core i7-2700K 3.50GHz CPU. We used $12k$ fluid particles and 0.5–$3.2k$ control particles. (The number of control particles was changed per target shape. See Table 1 for details.) Each time step took approximately 15–40 ms, which is enough for interactive applications.

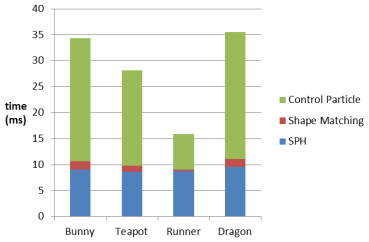


Fig 8: Performance of our experiments

Bunny	Teapot	Runner	Dragon
3197	2154	549	2714

Table 1: The number of control particles for each target shape

As shown in Figure 8, our method increases the burden according to the number of control particles involved. On more detailed investigation, we found that the computation of applying control force by control particles was the bottleneck. There is a possibility to decrease that burden by adjusting the influence radius of the control particles.

Although satisfactory results were obtained by the proposed method, there is one drawback: our solver is sensitive to parameters. In some cases, a slight change in a parameter makes the simulation unstable. Therefore, we have to choose parameters very carefully.

There are several avenues for future work. In our method, we generate control particles based on a user-specified area. However, the result

strongly depends on this parameter. Therefore, we feel that the control particles should be semi-automatically generated.

Our technique could be refined to a finer scale. One possible solution is optimization for GPU implementation. As the algorithm is separable at each phase, we could easily make use of GPU acceleration. Another possible solution is adaptive control of a fluid. Initially, control particles are generated with a uniform radius. After the fluid is approximately deformed to the target shape, each particle is replaced with suitable particles according to the distance from the boundary.

5 Conclusion

In this paper, we presented an interactive control method for physically based fluid animation. To control fluid shape, we dynamically generate control particles that affect neighbor fluid elements. The behavior of the control particles is sequentially determined by a shape-matching method. Unlike previous studies, our technique can interactively change target shapes. This enables more flexible fluid representations in interactive applications such as video games. We believe our algorithm is applicable to other animations, such as particle systems and flocks.

Acknowledgments

This work was supported by JSPS KAKENHI.

References

[1] Interactive shape control for fluid animation, pp. 108–114, 2013.

[2] year. Stable fluids, pp. 121–128, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[3] year, year. Visual simulation of smoke, pp. 15–22, New York, NY, USA, 2001. ACM.

[4] year. Fluid Simulation. A. K. Peters, Ltd., Natick, MA, USA, 2008.

[5] year. Controlling fluid animation, pp. 178–188, Washington, DC, USA, 1997. IEEE Computer Society.

[6] year. Practical animation of liquids, pp. 23–30, New York, NY, USA, 2001. ACM.

[7] year. Taming liquids for rapidly changing targets, pp. 229–236, New York, NY, USA, 2005. ACM.

[8] year, year, year. Keyframe control of smoke simulations. ACM Trans. Graph., pp. 716–723, July 2003.

[9] year, year, year. Fluid control using the adjoint method. ACM Trans. Graph., pp. 449–456, August 2004.

[10] year. Controlling fluid animation with geometric potential: Research articles. Comput. Animat. Virtual Worlds, pp. 147–157, July 2004.

[11] year, year, year. Detail-preserving fluid control, pp. 7–12, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[12] year, year, year, year. Guiding of smoke animations through variational coupling of simulations at different resolutions, pp. 217–226, New York, NY, USA, 2009. ACM.

[13] year. Guide shapes for high resolution naturalistic liquid simulation. ACM Trans. Graph., pp. 83:1–83:8, July 2011.

[14] year, year, year. Meshless deformations based on shape matching. ACM Trans. Graph., pp. 471–478, July 2005.

[15] year, year. Particle-based fluid simulation for interactive applications, pp. 154–159, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[16] year. Fastlsm: fast lattice shape matching for robust real-time deformation. ACM Trans. Graph., July 2007.

[17] year, year. Fast adaptive shape matching deformations, pp. 87–94, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.

[18] year, year. Screen space fluid rendering with curvature flow, pp. 91–98, New York, NY, USA, 2009. ACM.

Appendix

The list of parameters, which were used in the Stanford bunny demo, is given below:

fluid simulation parameters	
mass of particle	0.0002
radius of particle	0.0059
influence radius	0.012
rest density	800.0
gas constant	0.5
viscosity coefficient	0.02
time step	0.004
control particle parameters	
influence radius	0.03
attraction force constant	0.00008
velocity force constant	0.00005
fluid feedback coefficient	0.02
shape matching parameters	
stiffness of object	0.3
time step	0.005

Jun Yoshino

Jun Yoshino had been graduate school student of Kyushu University, and is currently working in Japanese game studio as a programmer. His research has focused on physically based computer animation.

Reiji Tsuruno

Reiji Tsuruno is an associate professor in faculty of design at Kyushu University, Japan. His research interests include visual computing,computer graphics, NPR, physically based animation and content designing system. He is a member of TheSociety for Art and Science of Japan, IPSJ, ACM and Eurographics.

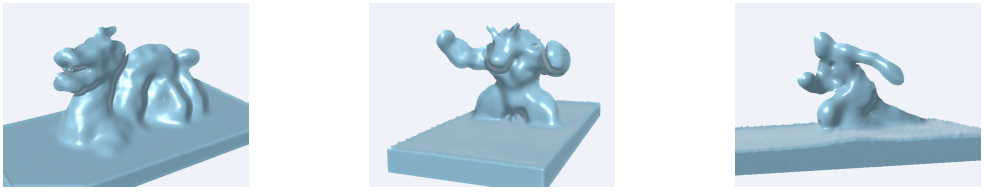
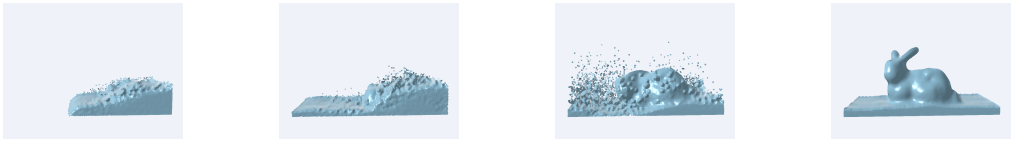


Fig 9: We demonstrated our technique using various target shapes

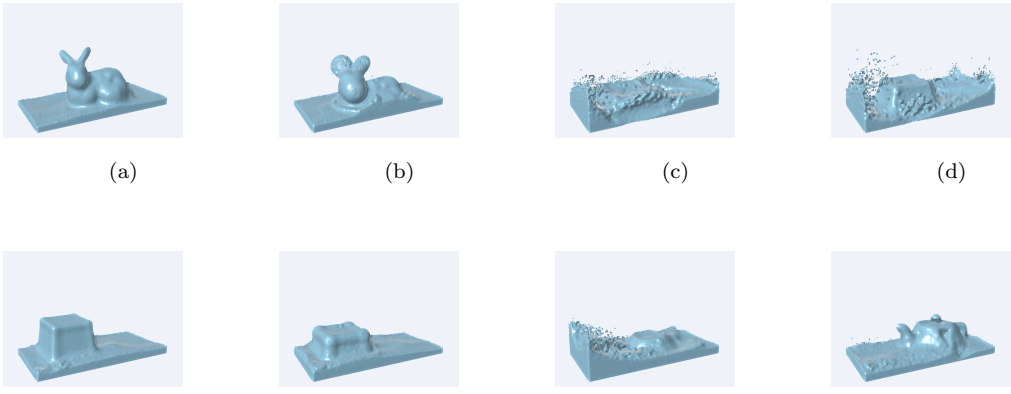


(a) (b) (c) (d)

Fig 10: Applying control to a fluid. Target shape is the Stanford bunny

(a) (b) (c) (d)

Fig 11: Visualizing control particles during the simulation, corresponding to Figure 10.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig 12: We interactively changed target shapes (bunny → cube → teapot)