Journal of Information Processing Vol.21 No.4 674–680 (Oct. 2013)

[DOI: 10.2197/ipsjjip.21.674]

Regular Paper

Calculation of Velocity on an Implicit Surface by Curvature Invariance

Makoto Fujisawa[1,a] Yojiro Mandachi[2,b] Kenjiro T. Miura[2,c]

Received: November 29, 2012, Accepted: May 18, 2013

Abstract: This paper presents an accurate method for computing the surface velocity which is used to advect the vertex in mesh-based surface tracking. We propose a curvature invariance condition that accurately captures the movement of a surface, especially in the case of rotating objects. The method uses the least-squares method and mesh fairing to solve the problem that the surface velocity would not be calculated when the implicit function defining the surface does not change. We show that the method works well in scenes including rotation and deformation.

Keywords: mesh-based surface advection, implicit surface, curvature

1. Introduction

Surface tracking is an important technique in computer graphics animation, where the appearance of most objects is defined by their surface. In particular, surface tracking and surface extraction are important for fluid or viscoelastic bodies which undergo significant changes of shape. Many methods extract a surface represented by an implicit function at every step of an animation using a piecewise linear approximation, such as the marching cubes algorithm. Although this approach can deal with changes of surface geometry (e.g., merging or splitting), it is difficult to define the relation between the current and previous surfaces. This causes a problem when we want to advect the surface texture or color, so a mesh-based approach is used instead.

Mesh-based surface tracking advects the initial surface mesh, which is created at the start by triangulation, according to the velocity of the surface. There are two problems that we have to solve to achieve mesh advection: 1) how to compute the surface velocity, and 2) how to deal with changes of topology. Some methods have been proposed to solve the second problem [4], [11], [16] but the normal velocity is commonly used as the surface velocity. This means that texture or color cannot be advected along with the surface, and it also causes large deformation of the mesh. Stam and Schmidt [12] proposed a mesh advection method that includes the tangential velocity. They assumed that the gradient of an implicit function would be maintained throughout the computation. Then they calculated a total velocity based on this assumption. However, the total velocity found takes account of translation only and does not include rotation.

We propose a method for calculating surface velocity by using a curvature invariance condition. Our method includes not only translation but also rotation, and we use the least squares method and mesh fairing to solve the problem that the surface velocity would not be calculated when the implicit function defining the surface does not change. Our results show that this method works well in scenes including rotation and deformation.

In order to track a surface calculated by the fluid simulation, we have to define the velocity on the surface. In most cases, the velocity on the surface is not directly defined, for example, the particle simulation just gives the velocity at the center of a particle. Moreover, the velocity field is undefined when we want to track the surface from an observed data such as a photo or a video. Our method does not require the knowledge of the velocity field. We assume that only the implicit function representing the surface is obtained and propose the calculation method for the surface velocity using it.

The remainder of the paper is organized as follows. Section 2 describes related work and we present our method in Section 3. Results and an analysis of our implementation are given in Section 4 and we conclude our work in Section 5.

2. Related Work

Fluid simulation is widely used to model complex fluid phenomena for computer graphics animation. There are many methods that use the surface tracking technique to capture a fluid's shape. In the level set method [6], [13], the liquid surface is represented as the zero level set of a signed distance function. Enright et al. [5] proposed a particle level set method to conserve the volume of the liquid. They advect marker particles with the level set function and modify the function according to the position of the particles. Bargteil et al. [1] have incorporated a volume-conserving level set function with the mesh-based approach. There are some methods that improve the level set method, such as CLSVOF [9], BFECC [7], and USCIP [8]. These methods just define the surface, so one must extract the mesh or render directly by ray tracing at every step.

[1] University of Tsukuba, Tsukuba, Ibaraki 305–8550, Japan
[2] Shizuoka University, Hamamatsu, Shizuoka 431–8011, Japan
[a] fujis@slis.tsukuba.ac.jp
[b] f0230063@ipc.shizuoka.ac.jp
[c] tmkmiur@ipc.shizuoka.ac.jp

The mesh-based approach adopts another strategy, advecting the mesh itself rather than an implicit function. Therefore we do not need to consider the correspondence of the mesh between frames. A major problem in the mesh-based approach is dealing with changes of the surface topology due to large and abrupt deformations of the fluid. Brochu and Bridson [4] used self-intersection information to reconstruct the mesh. Wojtan et al. [14] proposed using the deep cell test for detecting the change of geometry. Müller [10] extended Wojtan's approach to take care of self-intersection and thin features. Wojtan et al. [15] used surface flow to give more details of the surface at a scale smaller than the grid. Yu et al. [16] presented a mesh-based surface tracking method using surface tension for the smoothed-particle hydrodynamics (SPH) simulation.

Although there are many methods for dealing with a topology change, few approaches have been proposed for calculating the surface velocity. In most cases, the normal velocity or the velocity given by the simulation is used to advect the mesh; however, the deformation of the mesh will be small and small features of the surface will be preserved only if we use accurate velocities. Stam and Schmidt [12] calculated the total surface velocity, including the tangential component, under the assumption that the normal direction of the implicit function does not change during a frame. This method gives the tangential velocity for the mesh advection, but it cannot deal with the rotation due to the assumption of a rigid body translation. Bojsen-Hansen et al. [3] used some constraints from the initial mesh shape and topology to create an energy function and then found the correspondences between two meshes by minimizing that function. Their method could capture the movement of the surface mesh, but it needed a long computational time. Our method is based on the total velocity [12]. We extend the method to rotating objects by using the curvature invariance instead of the normal vector to estimate the tangential velocity.

3. Surface Tracking

3.1 Normal Velocity Approach

The surface is defined as the zero level set of the implicit function $f = f(\boldsymbol{x}, t)$:

$$\Gamma(t) = \{\boldsymbol{x} | f(\boldsymbol{x}, t) = 0\}. \tag{1}$$

By taking the total derivative, we obtain

$$f_x x_t + f_y y_t + f_z z_t + f_t = 0, \tag{2}$$

which shows that the advance of the surface ($f = 0$) is determined by the gradient of the implicit function ∇f and the surface velocity $\dot{\boldsymbol{x}} = (x_t, y_t, z_t)^T$. The spatial and temporal partial differentials of f are described as f_x, f_y, f_z and f_t.

If the derivatives of the implicit function are known, we can get the surface velocity [12]. By putting $\boldsymbol{q} = (f_x, f_y, f_z)^T$, Eq. (2) can be rewritten as

$$\boldsymbol{q}^T \dot{\boldsymbol{x}} = -f_t. \tag{3}$$

By using the pseudo inverse $(\boldsymbol{q}^T)^+ = \boldsymbol{q}(\boldsymbol{q}^T\boldsymbol{q})^{-1} = \boldsymbol{q}/|\boldsymbol{q}|^2$, we can solve this equation.

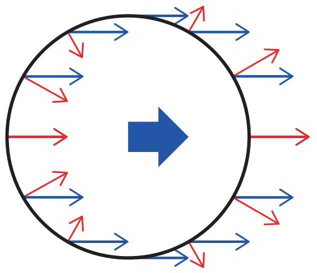
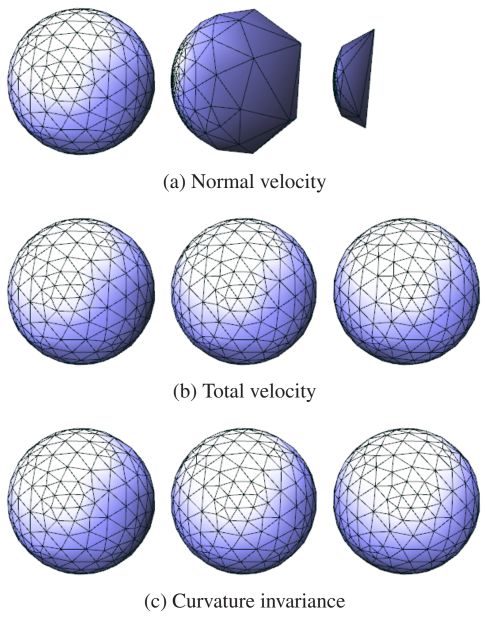


Fig. 1 Surface advection velocity (blue arrow) and its normal component (red arrow).



(a) Normal velocity

(b) Total velocity

(c) Curvature invariance

Fig. 2 Translation of a particle.

$$\dot{\boldsymbol{x}} = \frac{-f_t}{|\boldsymbol{q}|^2}\boldsymbol{q} = v_n \boldsymbol{n} \tag{4}$$

where $\boldsymbol{n} = \boldsymbol{q}/|\boldsymbol{q}|$ and $v_n = -f_t/|\boldsymbol{q}|$. This equation is a particular solution of (x_t, y_t, z_t) satisfying Eq. (2) and shows that the surface velocity $\dot{\boldsymbol{x}}$ at $f = 0$ is in the normal direction, and the tangential component v_t is zero. However, this result is inaccurate. **Figure 1** shows a circle being advected from left to right. If the normal vector points in the advection direction the normal velocity corresponds exactly to the surface velocity, while the difference becomes larger as the angle between the advection and the normal vector increases (see **Fig. 2**).

Knowing v_t is not necessary if we are interested in shape changes only. However, we must consider the tangential velocity when we want to represent the advection of texture or color. Moreover, $v_t = 0$ will cause a large deformation of the mesh. Hence it is important to get a good estimate of v_t. The remainder of this section describes our approach to determining the surface velocity, including the tangential component. The method consists of the following parts: determining a curvature invariance condition (Section 3.2), stable calculation of the surface velocity using the least squares method (Section 3.3), and mesh fairing after the advection (Section 3.4).

3.2 Curvature Invariance

To overcome the problems of the normal velocity approach, Stam and Schmidt [12] assumed that the orientation of the gradient of the implicit function $\nabla f/|\nabla f|$ is invariant over time.

$$\frac{d}{dt}\left(\frac{\nabla f}{|\nabla f|}\right) = 0. \tag{5}$$

The assumption is exactly satisfied for the translation of a rigid body but not for rotations. (see **Fig. 3**). Thus the condition in Eq. (5) cannot be used when there is a rotation. **Figure 4** (a) shows the result of a rotation of a particle with the total velocity. The shapes of the mesh gradually changed due to the error caused by this condition. Here, we propose a method to deal with rotations by using a surface curvature condition instead of the condition on the normal.

The curvature of an implicit surface can be calculated from the second-order derivative.

$$\kappa(\boldsymbol{u}) = -\frac{(\boldsymbol{u}, \nabla^2 f\boldsymbol{u})}{|\nabla f|}, \tag{6}$$

where $\boldsymbol{u}$ is a unit vector on the tangent plane, $\nabla^2 f$ is the Hessian matrix and $(\boldsymbol{u}, \nabla^2 f\boldsymbol{u})$ denotes the dot product of $\boldsymbol{u}$ and $\nabla^2 f\boldsymbol{u}$. The Hessian is defined as

$$\nabla^2 f = \begin{bmatrix} f_{xx} & f_{xy} & f_{xz} \\ f_{yx} & f_{yy} & f_{yz} \\ f_{zx} & f_{zy} & f_{zz} \end{bmatrix} \tag{7}$$

We assume the curvature is time-independent.

$$\frac{d\kappa}{dt} = 0. \tag{8}$$

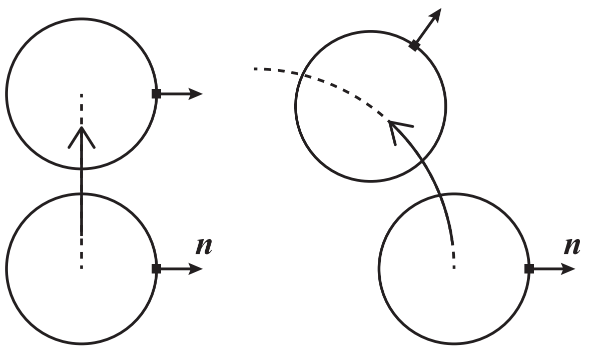


Fig. 3 Change of the normal direction in case of a translation (left) and a rotation (right).

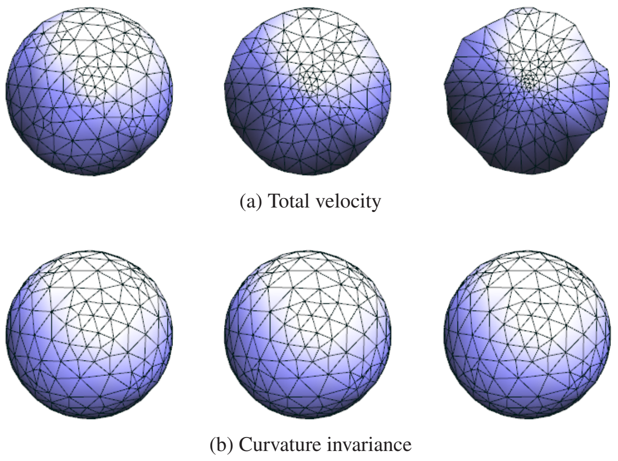


Fig. 4 Mesh advection of a particle through 10, 20, or 30 rotations.

We call Eq. (8) the curvature invariance condition. For three dimensions, two conditions are needed to evaluate the surface velocity on the tangent plane. The Gauss curvature K and the mean curvature H are defined by the following formulas [6].

$$\begin{aligned} K &= \frac{(\Delta f)^2 - tr(\nabla^2 f)^2}{2|\nabla f|^2} \\ &\quad + \frac{(\nabla f, (\nabla^2 f)^2 \nabla f) - \Delta f(\nabla f, \nabla^2 f \nabla f)}{|\nabla f|^4}, \\ H &= \frac{1}{2}\left(\frac{\Delta f}{|\nabla f|} - \frac{(\nabla f, \nabla^2 f \nabla f)}{|\nabla f|^3}\right). \end{aligned} \tag{9}$$

By taking total derivatives of K and H,

$$\begin{aligned} \frac{dK}{dt} &= 0, \\ \frac{dH}{dt} &= 0 \end{aligned} \tag{10}$$

equations for the surface velocity $\dot{\boldsymbol{x}} = (x_t, y_t, z_t)$ can be derived (see Appendix A.1). Equations (2) and (8) (or (10) for three dimensions) lead to a linear system for the velocity. These equations derived from the curvature invariance give the surface velocity, including the tangential component, even if the rigid body rotates.

The calculation formulae of the Gauss and mean curvature from an implicit function are not only above one. For example, Belyaev et al. [2] also derived other formulae to the calculate curvature. In order to show that our method works well if another formulae are used, we have implemented the method with Belyaev's formulae (see Appendix A.2) and confirmed that our curvature invariance approach also works well.

The curvature invariance condition can also be applied to a deformable body by assuming the curvature is locally invariant. However, surfaces where the curvature does not change along the tangent plane could not be tracked, for example, a sphere turning around its center or a plane moving along a tangent direction (see **Fig. 5**). These shapes and movements cause an indeterminacy of the linear system. We use a least squares method and mesh fairing to increase the robustness of our method against these possible problems.

3.3 Calculation of Surface Velocity

The linear system might be indeterminate if either the Gauss curvature or the mean curvature is nearly zero on the tangent plane or the curvature does not change on the tangent plane. To define the surface velocity for these areas is very difficult and the mesh could be advected in the wrong direction. In order to reduce the error caused by these areas, we define an error function

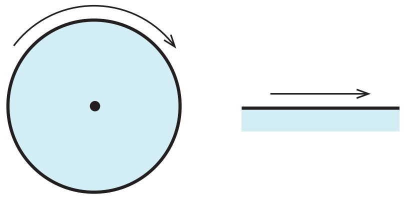


Fig. 5 The cases of the curvature does not change. The direction of the velocity corresponds to each tangent plane at the surface.

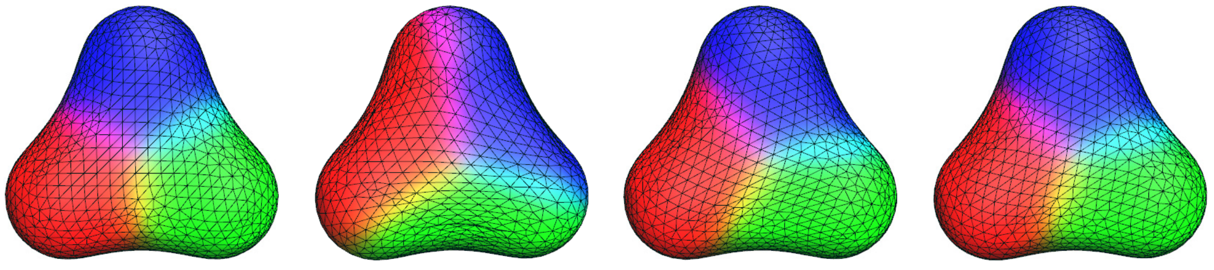


Fig. 6 Rotation of three particles.

$$e = \sum_{i=1}^{6} (F_i)^2, \tag{11}$$

where the F_i are the values of Eqs. (2), (10) and three components of the total velocity. The least squares method is then used to solve an error minimization problem. e is evaluated for each vertex and we just locally solve the linear system because the right hand side of Eq. (11) consists of the temporal differentials of K, H and the spatial and temporal differentials of f which are calculated only from the f. More accurate results and stable computation are achieved by using Eq. (11) because the area with an undefined velocity is reduced by including the total velocity.

3.4 Mesh Fairing

Although the least squares method gives a stable solution, there are still regions where we cannot get the exact solution because of the fixed implicit value, for example, a rotating sphere or a plane translation along a tangential direction. This can cause a large deformation of the mesh. One solution is to use the initial relationship with surrounding triangles to advect. We use the following mesh fairing technique based on a mass-spring system [12].

(1) For each vertex of the mesh, one-ring neighborhood vertices are projected onto the tangent plane.
(2) Projected vertices are fixed and the mass-spring system is applied with the rest length of the spring equal to initial length of the edge.
(3) The vertex is fitted onto the advected surface by calculating [11]

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n - \frac{f(\boldsymbol{x})\nabla f(\boldsymbol{x})}{|\nabla f(\boldsymbol{x})|^2}. \tag{12}$$

The procedure is applied to all vertices and iterated a user-specified number of times. In the mesh fairing, the well advected vertices, which are the exact solution of the linear system, drift due to pulling from surrounding vertices. Our method minimizes the drift effect because we are using both the total velocity and the curvature invariance.

4. Results

This section describes the results of applying the proposed method to a particle. Each particle has a position and an effective radius and we construct an implicit field [12]

$$f(\boldsymbol{x}, t) = \sum_{i=1}^{N} w_i(1 - r^2)^3 - T, \tag{13}$$

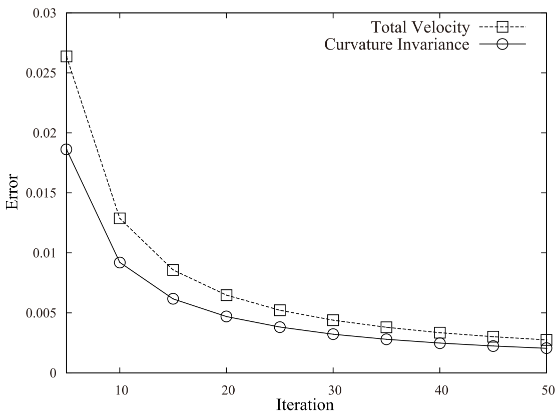
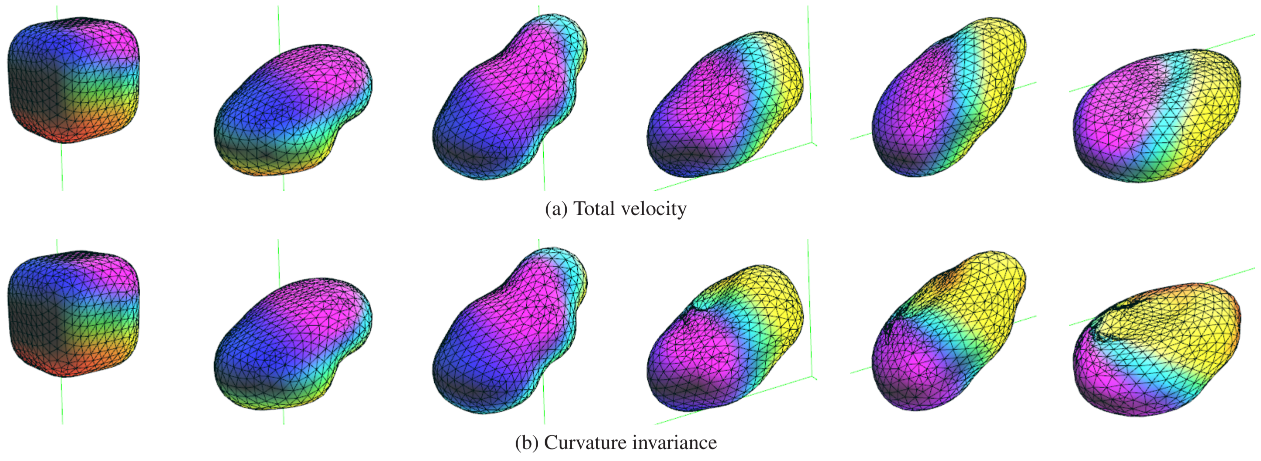


Fig. 7 Average error of vertex position by the number of iterations used for the mesh fairing.

where N is the number of neighborhood vertices, w_i is the weight function, T is the user specified threshold, $r = |\boldsymbol{x} - \boldsymbol{x}_i|/\sigma_i$ is the normalized distance to the neighborhood vertex and σ_i is the effective radius.

Figure 2 shows a translation of a particle. The shape advected by the normal velocity (Fig. 2 (a)) dynamically broke up because a vertex that has its normal perpendicular to the direction of motion remains at its original position. Note that these vertices still lie on the surface because of the movement toward the normal direction. On the other hand, the shapes are maintained during advection for the total velocity (Fig. 2 (b)) and the curvature invariance (Fig. 2 (c)) approaches. Both methods generate the same result in the case of a translation.

Figure 6 depicts the same experiment as described in Ref. [12]. Three particles are placed on a circle and rotate around their center. Figure 6 (a) is the initial condition and (b)–(d) show the results after 3 rotations with each advection method. The number of iterations for the mesh fairing is 10. The ideal result is that all vertices of the mesh return exactly to their initial positions. The vertices hardly follow the rotation at all when using the normal velocity approach. The total velocity advects the vertices with a slight drift and our curvature invariance reduces this drift effect. **Figure 7** shows a graph of the average error of the vertex position after the rotations for various numbers of iterations in the mesh fairing. Our method generates more accurate results, even if the number of iterations is small. The computation time was approximately 32 ms per step for the total velocity and 73 ms per step for the curvature invariance including the mesh fairing with 10 iterations with 3.2 GHz Intel Core i7 CPU. In both cases, the computation time of the mesh fairing was about 30 ms. Most of the


(a) Total velocity

(b) Curvature invariance

Fig. 10 Sequence of frames from the viscosity fluid simulations with SPH. The fluid flows downstairs.

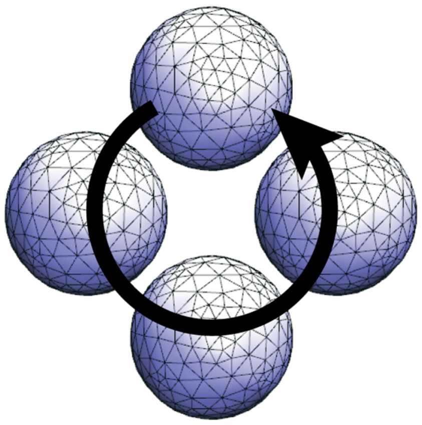


Fig. 8 Rotation of a particle.

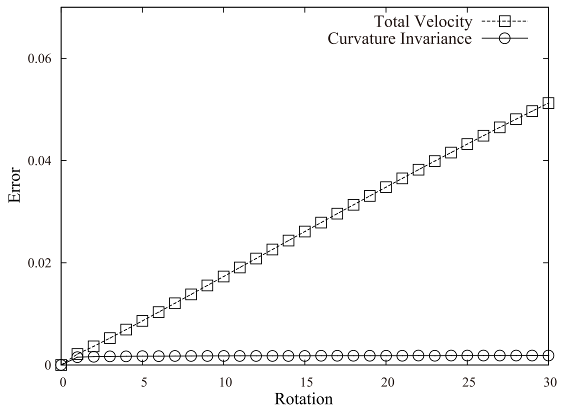


Fig. 9 Average error of vertex position by the number of rotations.

processing time for our method is involved in the minimization of Eq. (11). We used the Powell's method [13] for the minimization. More efficient algorithms such as BFGS will accelerate the computation.

The curvature invariance condition works well for rotations. **Figure 8** is the 1-particle version of Fig. 6 without the fairing. For the total velocity approach, the error gradually increases with the number of rotations, while our method maintains the original mesh shape as shown in Fig. 4. **Figure 9** shows a graph of the average error of the vertex position for various numbers of rotations.

Figure 10 shows a scene with deformed objects: The fluid is

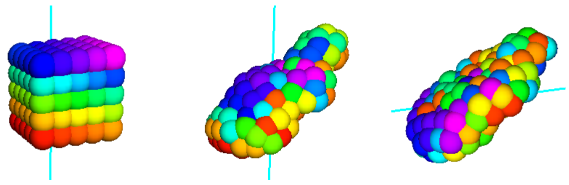


Fig. 11 Movement of particles from the simulation with SPH.

represented by particles that fall downstairs with surface rotation. The SPH method was used to simulate the behavior of the fluid. **Figure 11** shows the same scene but the particles used for the simulation are being displayed. The colors of the particles need not be exactly the same as the color of the surface (Fig. 10) because the particles inside the object would appear on the surface. The proposed method captures the surface movement better than the result obtained using the total velocity approach. However, some displaced polygons appear in Fig. 10 (b) due to the fact that convergence on the wrong local minimum during the least squares method caused the incorrect advection of vertices.

5. Conclusions and Future Work

We have presented a method for calculating the surface velocity by using curvature invariance. This enables more accurate computation of the surface velocity, even if the shape rotates. In addition, we use the least squares method and mesh fairing to reduce errors caused by the area for which the surface velocity cannot be adequately defined. The results show that our method could compute the surface velocity as accurately as previous methods even if the number of iterations used in the fairing is small. We also successfully applied our method to the SPH simulation of deformable viscous fluids.

The main limitation of our method is that the method cannot calculate the surface velocity if the value of the implicit function f does not change. For example, a sphere rotating around its center, a flat plane moving along the its tangential plane. The mesh fairing will work well if these movements locally appear. But if the large area of the surface remain unchanged, our method would cause a large error. In this case, another method like [3] or the velocity calculated by the simulation should be used to solve this problem.

In applying our method to a deformable object, some errors appeared (see Fig. 10), owing to convergence on the wrong local minimum during the least squares calculation. We will have to tackle this problem by detecting the area in which the surface velocity cannot be adequately defined. Using variable spring constants for the mesh fairing would prevent the slight drift in Fig. 6. Moreover, we will adopt a mesh reconstruction method, such as Refs. [10] or [14], for more complex scenes involving topological changes of the surface.

Acknowledgments We thank the anonymous reviewers for their helpful comments. This work was supported by research project of Graduate School of Library, Information and Media Studies, University of Tsukuba.

References

[1] Bargteil, A.W., Goktekin, T.G., O'Brien, J.F. and Strain, J.A.: A semi-Lagrangian contouring method for fluid simulation, *ACM Trans. Graphics*, Vol.25, No.1, pp.19–38 (online), DOI: http://doi.acm.org/10.1145/1122501.1122503 (2006).

[2] Belyaev, A.G., Pasko, A.A. and Kunii, T.L.: Ridges and Ravines on Implicit Surfaces, *Proc. Computer Graphics International 1998, CGI '98*, Washington, DC, USA, IEEE Computer Society, pp.530–535 (1998).

[3] Bojsen-Hansen, M., Li, H. and Wojtan, C.: Tracking surfaces with evolving topology, *ACM Trans. Graph.*, Vol.31, No.4, pp.53:1–53:10 (online), DOI: 10.1145/2185520.2185549 (2012).

[4] Brochu, T. and Bridson, R.: Robust Topological Operations for Dynamic Explicit Surfaces, *SIAM J. Scientific Computing*, Vol.31, No.4, pp.2472–2493 (online), DOI: 10.1137/080737617 (2009).

[5] Enright, D., Marschner, S. and Fedkiw, R.: Animation and rendering of complex water surfaces, *Proc. SIGGRAPH 2002*, New York, NY, USA, ACM Press, pp.736–744 (online), DOI: http://doi.acm.org/10.1145/566570.566645 (2002).

[6] Kanatani, K.: *Keijo CAD to zukei no kagaku* (*in Japanese*), Kyoritsu Shuppan (1998).

[7] Kim, B., Liu, Y., Llamas, I. and Rossignac, J.: Advections with Significantly Reduced Dissipation and Diffusion, *IEEE Trans. Visualization and Computer Graphics*, Vol.13, No.1, pp.135–144 (online), DOI: http://dx.doi.org/10.1109/TVCG.2007.3 (2007).

[8] Kim, T., Thürey, N., James, D. and Gross, M.: Wavelet turbulence for fluid simulation, *SIGGRAPH '08: ACM SIGGRAPH 2008 Papers*, New York, NY, USA, ACM, pp.1–6 (online), DOI: http://doi.acm.org/10.1145/1399504.1360649 (2008).

[9] Mihalef, V., Unlusu, B., Metaxas, D., Sussman, M. and Hussaini, M.Y.: Physics based boiling simulation, *Proc. SCA2006*, pp.317–324 (2006).

[10] Müller, M.: Fast and robust tracking of fluid surfaces, *SCA '09: Proc. 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, New York, NY, USA, ACM, pp.237–245 (online), DOI: http://doi.acm.org/10.1145/1599470.1599501 (2009).

[11] Ohtake, Y., Belyaev, A., Alexa, M., Turk, G. and Seidel, H.: Multi-level Partition of Unity Implicits, *ACM Trans. Graph.* (*Proc. ACM SIGGRAPH 2003*), Vol.22, No.3, pp.463–470 (online), DOI: http://doi.acm.org/10.1145/1201775.882293 (2003).

[12] Stam, J. and Schmidt, R.: On the velocity of an implicit surface, *ACM Trans. Graph.*, Vol.30, pp.21:1–21:7 (online), DOI: http://doi.acm.org/10.1145/1966394.1966400 (2011).

[13] Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P.: *Numerical Recipes in C++: The Art of Scientific Computing*, Cambridge University Press (2002).

[14] Wojtan, C., Thürey, N., Gross, M. and Turk, G.: Deforming meshes that split and merge, *SIGGRAPH 2009*, pp.1–10 (online), DOI: http://doi.acm.org/10.1145/1576246.1531382 (2009).

[15] Wojtan, C., Thürey, N., Gross, M. and Turk, G.: Physics-inspired topology changes for thin fluid features, *SIGGRAPH 2010*, pp.1–8 (online), DOI: http://doi.acm.org/10.1145/1833349.1778787 (2010).

[16] Yu, J., Wojtan, C., Turk, G. and Yap, C.: Explicit Mesh Surfaces for Particle Based Fluids, *EUROGRAPHICS 2012*, Vol.30, pp.41–48 (2012).

Appendix

A.1 Equations of Curvature Invariance with Kanatani's Formula

Equations for the curvature invariance can be derived from the total derivatives in Eq. (9). In this section, we use the following notation to represent the equations compactly.

$$\begin{aligned}\frac{df}{dt} &= \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial z}\frac{dz}{dt} + \frac{\partial f}{\partial t}\\ &= (f_x, f_y, f_z, f_t)\cdot(x_t, y_t, z_t, 1)\\ &= \boldsymbol{d}f\cdot\boldsymbol{V}.\end{aligned}$$

The total derivative of the Gauss curvature is

$$\begin{aligned}\frac{dK}{dt} &= \Big\{\Delta f|\nabla f|^4(\boldsymbol{d}f_{xx} + \boldsymbol{d}f_{yy} + \boldsymbol{d}f_{zz})\\ &\quad -(\Delta f)^2|\nabla f|^3(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\Big\}\cdot\boldsymbol{V}\\ &\quad -\frac{1}{2}|\nabla f|\Big\{\frac{d}{dt}\{tr(\nabla^2 f)^2\}|\nabla f|^3\\ &\quad -2tr(\nabla^2 f)^2(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\cdot\boldsymbol{V}\Big\}\\ &\quad +\frac{d}{dt}\left\{(\nabla f)^T(\nabla^2 f)^2\nabla f\right\}|\nabla f|^2\\ &\quad -4(\nabla f)^T(\nabla^2 f)^2\nabla f(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\cdot\boldsymbol{V}\\ &\quad -\frac{d}{dt}\left\{\Delta f(\nabla f)^T\nabla^2 f\nabla f\right\}|\nabla f|^2\\ &\quad\quad +4(\nabla f)^T\nabla^2 f\nabla f(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\cdot\boldsymbol{V},\end{aligned}$$

where Δf is the Laplacian defined as the following.

$$\Delta f = f_{xx} + f_{yy} + f_{zz}.$$

The total derivative of the mean curvature is

$$\begin{aligned}\frac{dH}{dt} &= \Big\{|\nabla f|^4(\boldsymbol{d}f_{xx} + \boldsymbol{d}f_{yy} + \boldsymbol{d}f_{zz})\\ &\quad -|\nabla f|^2\Delta f(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\\ &\quad -\{2(\boldsymbol{d}f_x, \boldsymbol{d}f_y, \boldsymbol{d}f_z)^T\nabla^2 f\nabla f\\ &\quad +(\nabla f)^T\begin{bmatrix}\boldsymbol{d}f_{xx} & \boldsymbol{d}f_{xy} & \boldsymbol{d}f_{xz}\\ \boldsymbol{d}f_{xy} & \boldsymbol{d}f_{yy} & \boldsymbol{d}f_{yz}\\ \boldsymbol{d}f_{xz} & \boldsymbol{d}f_{yz} & \boldsymbol{d}f_{zz}\end{bmatrix}\nabla f\}|\nabla f|^2\\ &\quad +3(\nabla f)^T\nabla^2 f\nabla f(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\Big\}\cdot\boldsymbol{V}.\end{aligned}$$

The equations of curvature invariance are $dK/dt = 0$ and $dH/dt = 0$.

A.2 Equations of Curvature Invariance with Belyaev's Formula

Belyaev et al. [2] give the following equations for the curvature of an implicit surface.

$$K = -\frac{1}{|\nabla f|^4}\begin{vmatrix}f_{xx} & f_{xy} & f_{xz} & f_x\\ f_{xy} & f_{yy} & f_{yz} & f_y\\ f_{xz} & f_{yz} & f_{zz} & f_z\\ f_x & f_y & f_z & 0\end{vmatrix} = -\frac{1}{|\nabla f|^4}\det(\boldsymbol{M}),$$

$$H = \frac{1}{2}\left\{\frac{\Delta f}{|\nabla f|} - \frac{\sum_{i=1}^{3}\sum_{j=1}^{3} f_i f_j f_{ij}}{|\nabla f|^3}\right\},$$

where $f_1 = f_x$, $f_2 = f_y$, $f_3 = f_z$.

By using their definitions, another version of the curvature invariance equations can be derived. The equation for the Gauss curvature is

$$\frac{dK}{dt} = -\frac{\frac{d}{dt}(\det(\boldsymbol{M}))}{|\nabla f|^4} + 4\frac{\det(\boldsymbol{M})(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\cdot\boldsymbol{V}}{|\nabla f|^6},$$

where

$$\begin{aligned}\frac{d}{dt}(\det(\boldsymbol{M})) = &\begin{vmatrix} \boldsymbol{d}f_{xx} & \boldsymbol{d}f_{xy} & \boldsymbol{d}f_{xz} & \boldsymbol{d}f_x \\ f_{xy} & f_{yy} & f_{yz} & f_y \\ f_{xz} & f_{yz} & f_{zz} & f_z \\ f_x & f_y & f_z & 0 \end{vmatrix}\cdot\boldsymbol{V} \\ &+\begin{vmatrix} f_{xx} & f_{xy} & f_{xz} & f_x \\ \boldsymbol{d}f_{xy} & \boldsymbol{d}f_{yy} & \boldsymbol{d}f_{yz} & \boldsymbol{d}f_y \\ f_{xz} & f_{yz} & f_{zz} & f_z \\ f_x & f_y & f_z & 0 \end{vmatrix}\cdot\boldsymbol{V} \\ &+\begin{vmatrix} f_{xx} & f_{xy} & f_{xz} & f_x \\ f_{xy} & f_{yy} & f_{yz} & f_y \\ \boldsymbol{d}f_{xz} & \boldsymbol{d}f_{yz} & \boldsymbol{d}f_{zz} & \boldsymbol{d}f_z \\ f_x & f_y & f_z & 0 \end{vmatrix}\cdot\boldsymbol{V} \\ &+\begin{vmatrix} f_{xx} & f_{xy} & f_{xz} & f_x \\ f_{xy} & f_{yy} & f_{yz} & f_y \\ f_{xz} & f_{yz} & f_{zz} & f_z \\ \boldsymbol{d}f_x & \boldsymbol{d}f_y & \boldsymbol{d}f_z & 0 \end{vmatrix}\cdot\boldsymbol{V},\end{aligned}$$

and the equation for the mean curvature is

$$\frac{dH}{dt} = \frac{1}{2}\left\{\frac{d}{dt}\frac{\Delta f}{|\nabla f|} - \frac{d}{dt}\frac{\sum_{i=1}^3\sum_{j=1}^3 f_i f_j f_{ij}}{|\nabla f|^3}\right\},$$

where

$$\begin{aligned}\frac{d}{dt}\frac{\Delta f}{|\nabla f|} \quad = \quad &\frac{1}{|\nabla f|^3}\Big\{|\nabla f|^2(\boldsymbol{d}f_{xx} + \boldsymbol{d}f_{yy} + \boldsymbol{d}f_{zz}) \\ &-\Delta f(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_z)\Big\}\cdot\boldsymbol{V},\end{aligned}$$

$$\begin{aligned}&\frac{d}{dt}\frac{\sum_{i=1}^3\sum_{j=1}^3 f_i f_j f_{ij}}{|\nabla f|^3} \\ &\quad= \frac{1}{|\nabla f|^3}\left\{\sum_{i=1}^3\sum_{j=1}^3\left(\boldsymbol{d}f_i f_j f_{ij} + f_i\boldsymbol{d}f_j f_{ij} + f_i f_j\boldsymbol{d}f_{ij}\right)\cdot\boldsymbol{V}\right\} \\ &\qquad-\frac{3}{|\nabla f|^5}\left\{\left(\sum_{i=1}^3\sum_{j=1}^3 f_i f_j f_{ij}\right)\left(f_x\boldsymbol{d}f_x + f_y\boldsymbol{d}f_y + f_z\boldsymbol{d}f_x\right)\cdot\boldsymbol{V}\right\}.\end{aligned}$$

Makoto Fujisawa is currently an assistant professor in the Faculty of Library, Information and Media Science, University of Tsukuba since 2011. He received B.Eng., M.Eng., and Ph.D. degrees in mechanical engineering from Shizuoka University in 2003, 2005, and 2008 respectively. He worked for Nara Institute of Science and Technology from 2008 to 2010 as an assistant professor. His research interests include computer graphics and physics simulation. He is a member of ACM, IEEE CS, IPSJ and VRSJ.

Yojiro Mandachi is currently a master course student in the Graduated School of Engineering, Shizuoka University. He received B.Eng. degree in mechanical engineering from Shizuoka University in 2012. His research interests include image processing and computer graphics.

Kenjiro T. Miura is currently a professor in the Graduate School of Science and Technology, Shizuoka University. He received B.Eng., and M.Eng. degrees from the Department of Precision Engineering, the University of Tokyo in 1982 and 1984, respectively, and Ph.D. degree in mechanical engineering from Cornell University in 1991. He has been a professor in the Department of Mechanical Engineering, Shizuoka University, since 2004. His research interests include computer aided geometric design and intelligent optical measurement. He is a member of ACM, IPSJ, JSME, and JSPE.