J Vis
DOI 10.1007/s12650-015-0317-7

REGULAR PAPER

XiaoKun Wang · XiaoJuan Ban · Xu Liu · YaLan Zhang · LiPeng Wang

Efficient extracting surfaces approach employing anisotropic kernels for SPH fluids

Received: 22 July 2015 / Revised: 17 September 2015 / Accepted: 12 October 2015

Abstract Particles are disordered throughout the entire process of fluid simulation using particle-based-methods, extracting surfaces through following up particles is unlikely to achieve. Therefore, it is reasonably necessary to extract fluid surfaces called surface reconstruction which has been research focus in particle-based fluid simulation for decades. To construct more smooth surfaces and enhance reconstruction efficiency in fluid simulation, this paper addresses an efficient anisotropic surface reconstruction method for particle-based fluid simulation. First, we simplify and modify the construction of traditional anisotropic kernel function. Second, we divide particles into near-surface particles and internal particles according to the analysis of particles' eigenvectors. Finally, near-surface particles are involved in the calculation of surface reconstruction while internal particles are directly assigned color field values through the number of neighbor particles. Experimental results show that this algorithm ensures smoothness and geometric characteristics of fluid surfaces reconstructed. Compared to existing algorithms, this approach is simple and easy to implement and greatly improves the operation efficiency.

Keywords Fluid simulation · SPH · Surface reconstruction · Anisotropic kernels

1 Introduction

Fluid phenomenon widely exists in nature, daily life and industrial production. The simulation of fluid also called fluid animation has always been the significant content in the research of physical-based animation as well as computer graphics. According to the different spatial discretization way, physically based animation can be divided into two main categories: mesh-based methods and mesh-free methods. In mesh-based methods, the simulation domain is discretized into mesh grids and physical attribute values on grid points (such as velocity and density) which are obtained by solving the governing equations. Nevertheless, the fluid volume is discretized into sampled particles in mesh-free methods. Moreover, each particle has physical properties and is advected by the governing equations. The mesh-free methods have the advantages of mass conservation, flexible extensibility of unbounded domains that make them easier to simulate more complex phenomena (such as spindrift and explosion), they have obtained rapid development in recent years and have become alternative ways of grid method effectively. Among various mesh-free approaches, smoothed particle hydrodynamics (SPH) is the most popular method for simulating fluid due to computational simplicity and efficiency.

Although the SPH method has been used to simulate all kinds of fluid phenomenon, it is very difficult to extract fluid surface. In addition, how to construct a smooth and flat fluid surface is an urgent issue that

X. Wang (✉) · X. Ban · X. Liu · Y. Zhang · L. Wang
School of Computer and Communication Engineering, University Science and Technology Beijing, Beijing 100083, People's Republic of China
E-mail: wang1xiao2kun3@163.com

Published online: 17 November 2015

needs to be solved. SPH method is difficult to construct smooth surface due to its irregular particle distribution, hence color field method is always used for surface reconstruction. On account of the surface being not smooth enough and usually having bumps or indentations, it is reconstructed by the traditional color field method. At the same time, existing anisotropic approaches extracting fluid surface is seriously slow, therefore, this paper proposes an efficient surface reconstruction algorithm based on anisotropic improvement of color field function. First, we construct an anisotropic kernel function for each particle according to the neighbor particles distribution. Compared to the traditional anisotropic kernel function, the computing method of kernel function proposed in this paper is more simple. Particles can be divided into near-surface particles and internal particles according to the analysis of the particles' eigenvector. The near-surface particles are involved in calculation when processing surface reconstruction. The internal particles' color field values are directly assigned by means of their number of neighbor particles, which further improves the calculation efficiency.

2 Related work

Since extracting fluid surfaces is vital for realistic animation, approaches for reconstructing and tracking fluid surfaces have attracted widespread attention when fluid simulation was first introduced in computer graphics. The reconstructing fluid surfaces can be divided into two categories: mesh-based methods and mesh-free methods. Under the framework of mesh-based methods, researchers have presented a large number of approaches to reconstruct surfaces. Osher and Fedkiw (2002) realized the Euler fluid surfaces tracking using the level set method. To solve the problem of mass loss in level set method, particle level set method (Enright et al. 2002) and semi-Lagrangian contouring method (Enright et al. 2005) have been proposed. Muller (2009) employed marching cubes grid to sample fluid surfaces, while keeping the grid sampling points as before. Wojtan et al. (2010) rebuilt the grid using convex hull method that kept features of thin fluid. The following year, Bhattacharya et al. (2011) improved Williams's surface reconstruction method by minimizing energy consumption of the surface represented by level set. Wojtan et al. (2009) used iso-surface building methods to rebuild grids which changed topological structure. Hirt and Nichols (1981) proposed volume-of-fluid methods (VOF). Brochu and Bridson (2006) introduced explicit surface tracking method.

For the past few years, with the application of SPH framework in fluid simulation, researchers carried out an in-depth study and exploration on reconstructing surfaces based on mesh-free method. Blinn (1982) presented the classic blobby spheres approach. A new kind of implicit surface equations is employed using distance of sampling points to the scattered point as a parameter. The values of equations are used to determine whether the sampling points are on the surface; it successfully realizes surface extraction of the discrete points. One of the defects is that high or low densities of particles will cause bumps or indentations on the surface. Muller et al. (2003) proposed a reconstructing approach which is similar to level set method by calculating the color field function. Fluid free surface can be constructed simply and rapidly by this method, but the surface extracted is rough and the particles near the surface will cause bumps. Zhu and Bridson (2005) modified Blinn's algorithm to correct local particle density variations. Primarily, they calculate fluid particles' weighted average of coordinates and radius on the basis of the neighbor particles' position and radius. Then, they apply the particles' weighted average coordinates and radius to the calculation of reconstructing surfaces, thus get a relatively smooth fluid surface that do not have obvious bumps. Adams et al. (2007) put forward improved method by tracking the particle-to-surface distances over time which made the surfaces more smooth. But the surface expression using implicit function is time-consuming when implicit function is constructed. Williams (2008) achieved global smoothness on surface mesh through solving a nonlinear optimization problem iteratively. Yu et al. (2012) proposed a new reconstruction method for fluid surfaces. They only construct surfaces at the beginning of the simulation, and then for the each next frame, they merely process triangular mesh segmentation and integration of operations to previous frame surface, no longer reconstruct fluid surfaces each frame. Yu and Turk (2013) employed Muller's 2003 implicit surface method and introduced the anisotropic kernels to the implicit surface which extract more real and smooth surfaces of the fluid.

Essentially, SPH method's core idea is to use summation of neighbor particles' value in local area to replace field function and its derivative. The distribution of local area neighbor particles will directly affect the calculation results and the accuracy of SPH method in consequence. Particles on the surface have less neighbors than those in the internal which causes smaller density obtained by SPH formula. Worse still, it

results in a series of influence and eventually leads to the surface distortion. To solve this problem, Schechter and Bridson (2012) addressed ghost SPH to compensate for surface particle density variations. They create a random ghost particle layer in the external of original particle boundary that modifies SPH kernel function calculation by increasing particle number within the scope of support domain. Besides, Liu and Liu (2003) presented ASPH method to solve the SPH calculation of lacking neighbors in local region. They construct the distance covariance matrix of the particles and their neighbors to analyze the distribution characteristics of the surrounding particles, then acquire eigenvectors served as distance weight of different direction to achieve adaptive effects. Then Yu and Turk (2013), inspired by ASPH method, introduced the anisotropic kernels method that applied weighted principle component analysis (WPCA) (Koren and Carmel 2003) to orient and scale the anisotropic kernel. In their method, each particle owns a unique anisotropic kernel function obtained through the particle's neighborhood distribution.

Yu's method solved the problem of smoothness on fluid surface to some extent. Each particle has its own influence scope when calculating the color field. Unlike the traditional methods, Yu's method cannot directly iterate over all pixels to complete the color field calculation. Whereas it needs to traverse pixels based on each particle's influence range and bring repeated traversal of the pixel, the calculation was directly affected by the number of particles. Moreover, it requires respective matrix calculations since the kernel function varies for each particle, which is time-consuming and inefficient. To solve the above problems, we propose an efficient SPH-oriented fluid surfaces extracting method using anisotropic kernels on the basis of Yu's method. We modify the formulation of anisotropic kernels, then classify particles by extracting the location characteristics, and only reconstruct surfaces by the near-surface particles and directly assign color field values for internal particles. Without affecting the effect of reconstruction, the method in this paper preferably saves computing resources and greatly improves efficiency of the extracting algorithm.

3 SPH framework for fluid simulation

In the Lagrangian description, flow-controlled partial differential equations of Navier–Stokes for fluids can be expressed as

$$\frac{\mathrm{d}\rho_i}{\mathrm{d}t} = -\rho_i \mathbf{v_i}, \tag{1}$$

$$\rho_i \frac{D\mathbf{v_i}}{Dt} = -\nabla p_i + \rho_i g + \mu \nabla^2 \mathbf{v_i}; \tag{2}$$

where $\mathbf{v_i}$ is the velocity, ρ_i is the density, p_i is the pressure, μ is the viscosity coefficient and g represents the external force field. Equation (1) is mass equation and Eq. (2) is the momentum equation.

The theory of SPH (Becker and Teschner 2007) is to utilize the form of discrete particles to characterize the successive fields and use integration to approximate the fields. For particle i at location x_i,

$$\langle A(x_i) \rangle = \sum_j m_j \frac{A_j}{\rho_j} W\left(x_i - x_j, h\right) \tag{3}$$

where m_j and ρ_i represent particle mass and density, respectively, $W\left(x_i - x_j, h\right)$ is the smoothing kernel and h the smoothing radius.

Applying Eq. (3) to the density of a particle i at location x_i yields the summation of density:

$$\rho_i = \sum_j m_j W\left(x_i - x_j, h\right) \tag{4}$$

Thus, forces between particles including pressure $\mathbf{f}_i^{\mathrm{P}}$ and viscous force $\mathbf{f}_i^{\mathrm{v}}$ can be represented as:

$$\mathbf{f}_i^P = -\sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} \right) \nabla W_{ij} \tag{5}$$

$$\mathbf{f}_i^{\mathrm{v}} = \mu \sum_j m_j \frac{\mathbf{v}_{ji} \rho_j}{\nabla^2} W_{ij} \tag{6}$$

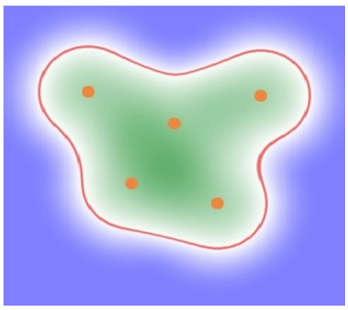


Fig. 1 In the figure, *green region* signifies interior $\phi > 0$, *blue region* signifies exterior $\phi < 0$, the color depth represents the intensity of the field, *red curve* implies iso-surface $\phi = 0$. Surface expressed by implicit surface

In this article, We employ Tait equation (Muller et al. 2003) to calculate the pressure, that is $p_i = \frac{\rho_0 c_s^2}{\gamma}\left(\left(\frac{\rho_i}{\rho_0}\right)^{\gamma} - 1\right)$. Where ρ_0= 1000 is the rest density of the fluid, γ = 7 is stiffness parameters and c_s is velocity of sound. We use the method in reference (Akinci et al. 2012) to compute viscous force.

4 Surface reconstruction based on anisotropic kernels

4.1 Surface definition

We exploit implicit surfaces to represent fluid surfaces in this paper. In the first place, we define a domain $\bar{D}$ cover the fluid space D that contained all particles, that is $D \subseteq \bar{D}$. Fluid surface is $S = \partial D$ which means a iso-surface of spatial function $\phi(\mathbf{x}) : R^d \rightarrow R$, that is

$$S = \partial D = \{\mathbf{x} | \mathbf{x} \in \bar{D}, \phi(\mathbf{x}) = 0\} \tag{7}$$

Furthermore, the internal and external regions of fluids can be expressed as follows,

$$\begin{cases} \phi(\mathbf{x}) > 0; & \forall \mathbf{x} \in \mathrm{D} \backslash \partial D \\ \phi(\mathbf{x}) = 0; & \forall \mathbf{x} \in \partial D \\ \phi(\mathbf{x}) < 0; & \forall \mathbf{x} \in \bar{D} \backslash D \end{cases} \tag{8}$$

In simulation, the definition above is easy to discretize, that is to say, we just need to divide $\bar{D}$ into small grids $\mathbf{x}(i,j,k) \in \bar{D}$, where (i, j, k) is the grid index. After discretization, Eq. (8) can be defined as:

$$\begin{cases} \phi(i,j,k) > 0; & \forall \mathbf{x}(i,j,k) \in D \backslash \partial D \\ \phi(i,j,k) = 0; & \forall \mathbf{x}(i,j,k) \in \partial D \\ \phi(i,j,k) < 0; & \forall \mathbf{x}(i,j,k) \in \bar{D} \backslash D \end{cases} \tag{9}$$

For the definition of surface, Muller (2003) used the following formula:

$$\phi(\mathbf{x}) = \sum_j m_j \frac{1}{\rho_j} W(\mathbf{x} - \mathbf{x}_j, h_j) - C \tag{10}$$

where C is a positive constant, $W(\mathbf{x} - \mathbf{x}_j, h_j)$ is kernel function that can select any suitable smoothing function (Fig. 1).

4.2 Surface reconstruction using anisotropic kernels

Normally, the color field definition (Muller et al. 2003) is as follows

$$\phi(\mathbf{x}) = \sum_j \frac{m_j}{\rho_j} W(\mathbf{x} - \mathbf{x}_j, h_j) \tag{11}$$

In Eq. (11), W is isotropic kernel function and its expression is

$$W(\mathbf{r},h)=\frac{\sigma}{h^d}P\left(\frac{||\mathbf{r}||}{h}\right) \tag{12}$$

where σ is a scaling factor, d is the dimension of the simulation, $\mathbf{r}$ is a radial vector, and P is a symmetric decaying spline with finite support.

To cope with the problem of density distributions near the surface, Yu and Turk (2013) capture the density distribution more accurately by replacing h with a $d \times d$ real positive definite matrix G, they redefine W to be an anisotropic kernel

$$W(\mathbf{r},G)=\sigma \det(G)P(||G\mathbf{r}||) \tag{13}$$

In the preceding equation, $r=\mathbf{x}-\mathbf{x}_i$, $\mathbf{x}$ can be treated as any position, the linear transformation G rotates and stretches the radial vector $\mathbf{r}$.

To analyze the impact of particles on the surrounding location, it is necessary to establish the covariance matrix of the particles. The covariance matrix C_i of particle i is formulated as

$$C_i=E((\mathbf{x}_j-\mathbf{x}_i)(\mathbf{x}_j-\mathbf{x}_i)^{\mathrm{T}}) \tag{14}$$

Yu replaces $\mathbf{x}_i$ with $\bar{\mathbf{x}}_i$ to produce a more smooth surface, thus Eq. (14) is rewritten as

$$C_i=E((\mathbf{x}_j-\bar{\mathbf{x}}_i)(\mathbf{x}_j-\bar{\mathbf{x}}_i)^{T}) \tag{15}$$

$$\bar{\mathbf{x}}_i=\sum_j w_{ij}\mathbf{x}_j/\sum_j w_{ij} \tag{16}$$

The function w_{ij} is an isotropic weighting function

$$w_{ij}=\frac{8}{\pi h^3}\begin{cases}1-\frac{6r^2}{h^2}+\frac{6r^3}{h^3} & 0<r<\frac{h}{2}\\ \frac{2(h-r)^3}{h^3} & \frac{h}{2}<r<h\\ 0 & r<0 \quad \text{or} \quad r>h\end{cases} \tag{17}$$

where $\mathbf{r}$ is the distance of particle location between $\mathbf{x}_i$ and $\mathbf{x}_j$, h is the supporting radius of smoothing kernel. To include enough neighborhood particles and obtain reasonable anisotropy information, we choose $r=2h$.

For each particle, singular value decomposition (SVD) can be performed on covariance matrix C_i since C_i is a real symmetric matrix. C_i can be written as

$$C_i=R\Sigma R^{\mathrm{T}} \tag{18}$$

$$\sum=\mathrm{diag}(\sigma_1,\ldots,\sigma_{rmd}) \tag{19}$$

In the above formulas, R is a 3×3 rotation matrix with the eigenvectors of C_i as column vectors. Each column R_i that represents distribution axis of C_i corresponds to eigenvalue σ_i in Σ. Σ is a diagonal matrix with eigenvalues $\sigma_1 \geq \cdots \sigma_{rmd}$, so the quantity of neighbor particles is large in the R_1 direction and smallest in the R_3 axis. To avoid extreme conditions and unexpected situation to occur such as σ_1 is much greater than σ_{rmd} when matrix transformation is carried out, Yu modifies covariance matrix C_i. Primarily, they check whether σ_1/σ_{rmd} is smaller than k_{rmr} with $k_{rmr}>1$. Then use $G=k_n I$ to replace anisotropic kernels for internal fluid particles as well as isolated particles. Next, they employ $\tilde{C}_i$ instead of the modificatory C_i, but to ensure the kernel W of particle i deform in line with C_i, G must be an inversion of C_i and be scaled by $1/h$ to maintain the same form with W. In summary, G can be expressed as

$$\sigma_i{}'=\begin{cases}\sigma_i & \sigma_1<k_r\sigma_i\\ \frac{\sigma_1}{k_r} & \sigma_1>k_r\sigma_i\end{cases} \tag{20}$$

$$G=\frac{1}{h}R\tilde{\Sigma}^{-1}R^{\mathrm{T}} \tag{21}$$

4.3 Advanced anisotropic surface reconstruction

Seen from the previous paragraph that matrix G realizes the functions of anisotropic kernels. We can control the scaling of r on three axes that R included through the matrix G and determine the value of $W(\mathbf{r},h)$ by $\|G\mathbf{r}\|$. In this part, we will explain the working principle of $\|G\mathbf{r}\|$. Eigenmatrix R of $\|G\mathbf{r}\|$ has three

orthogonal components R_1, R_2, R_3 which are all unit vector. As a result, r can be alternated by R_1, R_2, R_3 in three-dimensional space. That is,

$$\mathbf{r} = c_1 \cdot R_1 + c_2 \cdot R_2 + c_3 \cdot R_3 \tag{22}$$

Therefore, $G\mathbf{r}$ can be written as

$$\begin{aligned} G\mathbf{r} &= \frac{1}{h} R\tilde{\Sigma}^{-1} R^{\mathrm{T}} \cdot (c_1 \cdot R_1 + c_2 \cdot R_2 + c_3 \cdot R_3) \\ &= \frac{1}{h} R\tilde{\Sigma}^{-1} (c_1 \cdot R^{\mathrm{T}} \cdot R_1 + c_2 \cdot R^{\mathrm{T}} \cdot R_2 + c_3 \cdot R^{\mathrm{T}} \cdot R_3) \end{aligned} \tag{23}$$

Because

$$R^{\mathrm{T}} = [R_1, R_2, R_3]^{\mathrm{T}} = \begin{bmatrix} R_1{}^T \\ R_2{}^T \\ R_3{}^{\mathrm{T}} \end{bmatrix} \tag{24}$$

$$c_1 \cdot R^T \cdot R_1 + c_2 \cdot R^T \cdot R_2 + c_3 \cdot R^T \cdot R_3 = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} \tag{25}$$

From the above, we can find $R \cdot \mathbf{r}$ as the projection of $\mathbf{r}$ in the direction of R_1, R_2, R_3, which is consistent with the concept of vector inner product. It also implies, convert coordinates of $\mathbf{r}$ from the coordinate system ((1, 0, 0), (0, 1, 0), (0, 0, 1)) to R_1, R_2, R_3, that is the coordinate from (x, y, z) to (c_1, c_2, c_3). Figure 2 illustrates the transformation process of coordinate system, the solid line is the initial coordinates, dotted line is the coordinates after transformation.

In conclusion, $G\mathbf{r}$ can further be expressed as:

$$G\mathbf{r} = \frac{1}{h} R\tilde{\Sigma}^{-1} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \frac{1}{h} R \begin{bmatrix} \frac{1}{\sigma_1} & & \\ & \frac{1}{\sigma_2'} & \\ & & \frac{1}{\sigma_3'} \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \frac{1}{h} R \cdot \begin{bmatrix} \frac{c_1}{\sigma_1} & & \\ & \frac{c_2}{\sigma_2'} & \\ & & \frac{c_3}{\sigma_3'} \end{bmatrix} \tag{26}$$

From the preceding equation, we can detect the scaling effect of $\mathbf{r}$ through eigenvalue and σ_i' is modified by Eq. (20).

But

$$R \cdot \begin{bmatrix} \frac{c_1}{\sigma_1} & & \\ & \frac{c_2}{\sigma_2'} & \\ & & \frac{c_3}{\sigma_3'} \end{bmatrix} = \frac{c_1}{\sigma_1} \cdot R_1 + \frac{c_2}{\sigma_2'} \cdot R_2 + \frac{c_3}{\sigma_3'} \cdot R_3 \tag{27}$$

It is almost the inverse operation of the formula on (25). That is, converting the transformed coordinates to the original coordinates. Actually, Eq. (27) is redundant in numerical analysis. Because in the actual calculation, we are only concerned with the changes of L1 norm value after stretching. The following brief proof

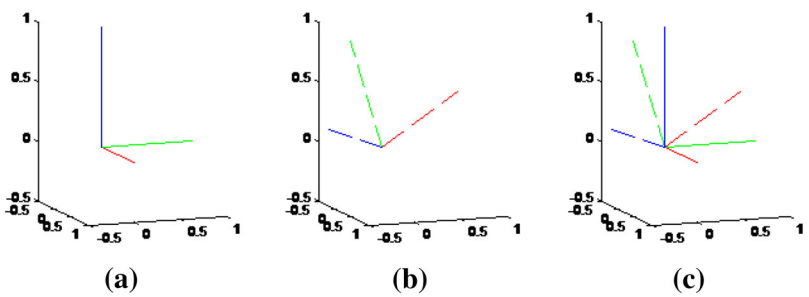


Fig. 2 **a** Standard coordinate system **b** transformed coordinate system **c** compared the two coordinate systems the coordinate transformation

$$\begin{aligned}\|G\mathbf{r}\| &= \frac{1}{h}\cdot\left\|R\tilde{\Sigma}^{-1}R^{\mathrm{T}}\cdot\mathbf{r}\right\| \\ &= \frac{1}{h}\cdot\left\|\left(\frac{c_1}{\sigma_1}\cdot R_1+\frac{c_2}{\sigma_{2'}}\cdot R_2+\frac{c_3}{\sigma_{3'}}\cdot R_3\right)\right\| \\ &= \frac{1}{h}\cdot\sqrt[2]{\left(\frac{c_1}{\sigma_1}\right)^2+\left(\frac{c_2}{\sigma_{2'}}\right)^2+\left(\frac{c_3}{\sigma_{3'}}\right)^2}=\frac{1}{h}\cdot\left\|\tilde{\Sigma}^{-1}R^{T}\cdot\mathbf{r}\right\|\end{aligned} \tag{28}$$

Thus, we can replace G with G',

$$G' = \tilde{\Sigma}^{-1}R^{\mathrm{T}} \tag{29}$$

Compared with Eq. (21), the expression of formula (29) is simpler with higher computational efficiency and same results. The definition of W is then modified as:

$$W(\mathbf{r}, G') = \sigma|\det(G')|P(||G'\mathbf{r}||) \tag{30}$$

Notice that formula (30) taken the absolute value of the determinant of G'. The reason is that the absolute value of the determinant of G' and G is equal which is guaranteed by the determinant's characteristics. The determinant of G can be guaranteed positive according to Eq. (31). However, determinant of G' needs to be guaranteed positive through Eq. (32).

$$\det G = (\det R)\det(\Sigma)(\det R^{\mathrm{T}}) = (\det R)^2\det(\Sigma) \tag{31}$$

$$\det G' = \det(\Sigma)(\det R^{\mathrm{T}}) = (\det R)\det(\Sigma) \tag{32}$$

Under the three-dimensional coordinates, determinant indicates the volume composed by column vectors of matrix R. Because column vectors of R are unit vectors and perpendicular to each other, determinant of R is 1 or -1. Besides, determinant of diagonal matrix is the product of diagonal elements, referring to formula (33). Hence, Eq. (30) can be further represented by formula (35).

$$\det\Sigma = \Pi\Sigma_{ii} \tag{33}$$

$$\det G = abs(\det G') = \Pi\Sigma_{ii} \tag{34}$$

$$W(\mathbf{r}, G') = \sigma\cdot\Pi\sum_{ii}\cdot P(||G'\mathbf{r}||) \tag{35}$$

Overall, Eq. (35) is the color field function proposed in this article. Compared with the original formula (13), the approach deduced in this paper simplifies matrix operations and reduces the computational costs without changing the final results. The results will be shown in part 5.

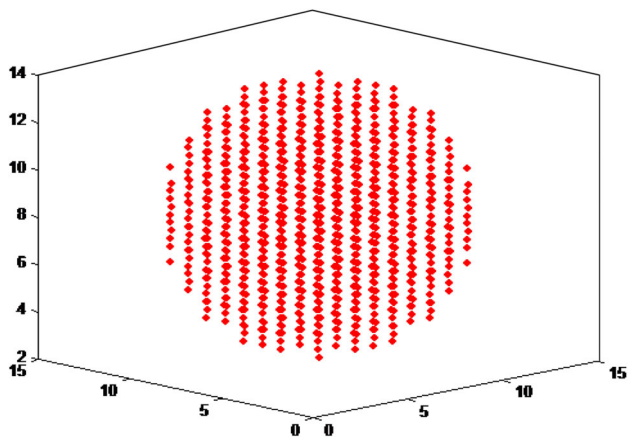


Fig. 3 Schematic diagram of a sphere consists of 925 particles

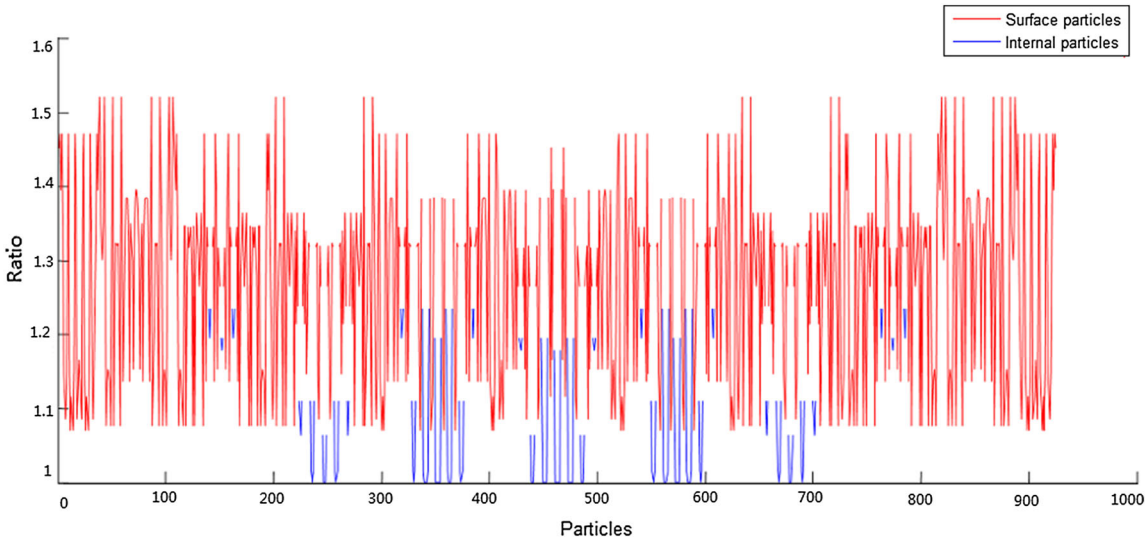


Fig. 4 The eigenvalue ratio σ_3/σ_1 of the spheres 925 particles, *red color* represents the surface particles and *blue color* represents the internal particles

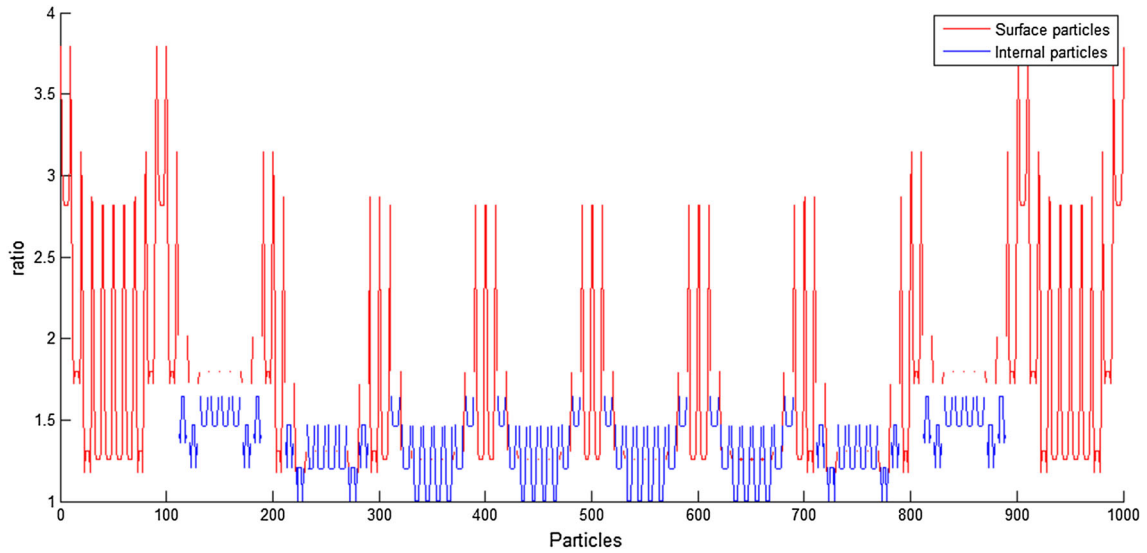


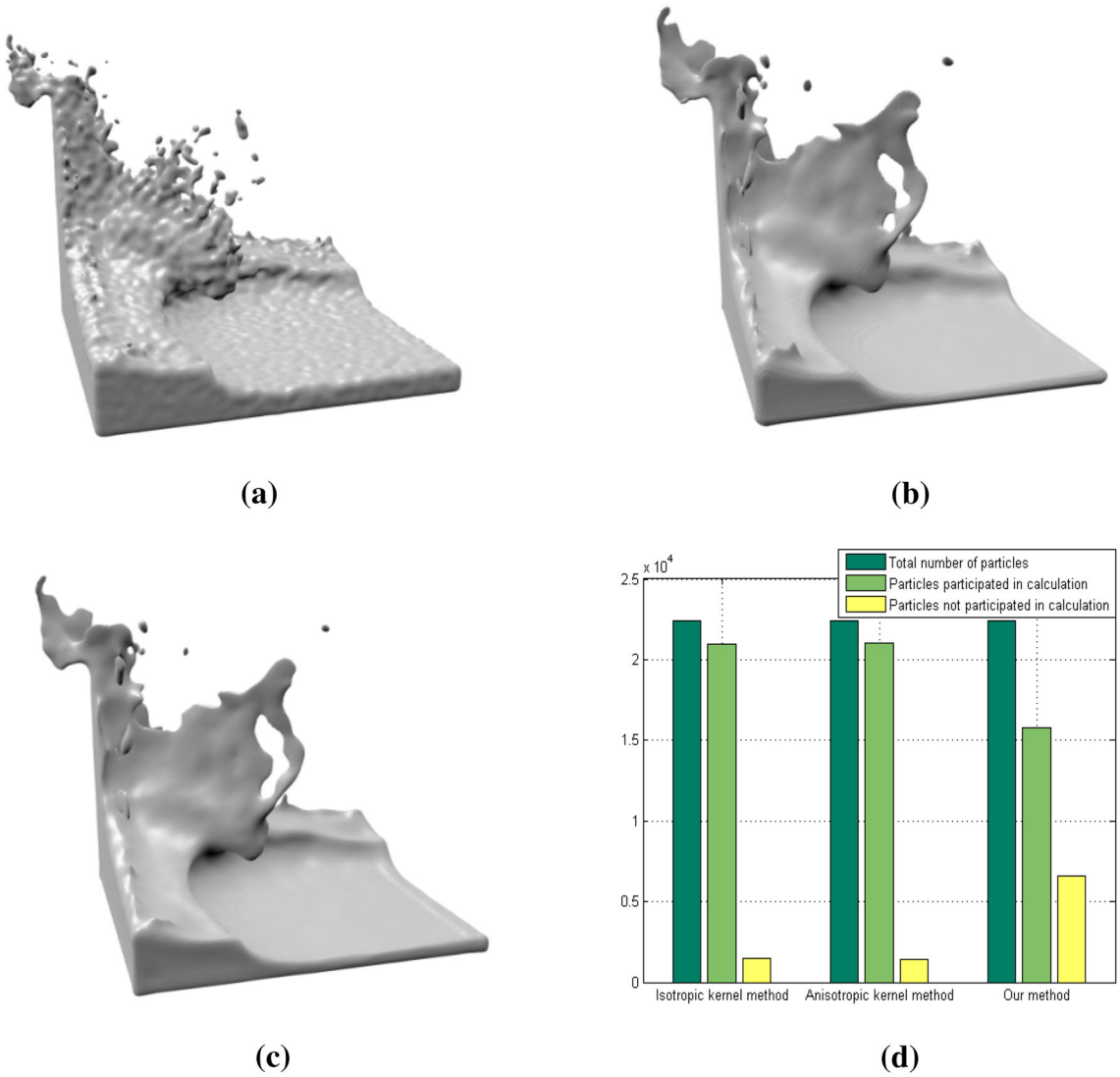
Fig. 5 The eigenvalue ratio σ_3/σ_1 of the cubes 1000 particles, *red color* represents the surface particles and *blue color* represents the internal particles

4.4 Extracting surface and near-surface particles

The interior fluid particles have no substantial contribution to reconstructing surfaces in practice. To further increase the computational efficiency and save computing resources, unlike traditional method based on neighbor particles classification, we classify the particles according to its eigenvalues. Since the characteristic values have been solved in the previous calculation, the classification of particles brings no additional costs. For the sake of ensuring the weak compressibility, SPH method usually forces the particle to keep an average particle spacing with little change while running in the simulation. According to the state equation, the average distance changes between general particles do not exceed 1/100 (Liu and Liu 2003). Consequently, we have analyzed two typical scenarios that are: cube and sphere. A sphere comprised 925 particles which is shown in Fig. 3 and its ratio curve of σ_3/σ_1 is referred in Fig. 4. We calculate the eigenvalues $\sigma_1, \sigma_2, \sigma_3 (\sigma_1 < \sigma_2 < \sigma_3)$ and ratio σ_3/σ_1, σ_3/σ_2 for each particle. We found that the more they are near to the center of the geometry, the values of $\sigma_1, \sigma_2, \sigma_3$ is more closer and the equal

(a) (b)

Fig. 6 **a** Not processing internal particle, leaving a cavity **b** assign values for internal particle, eliminating cavity comparison of handling internal particles, **a** not calculated internal particle **b** calculated the internal particle



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

Fig. 7 **a** The isotropic kernels method, **b** the anisotropic kernels method, **c** our method and **d** comparison of particles number Surface reconstruction in comparison with free surface

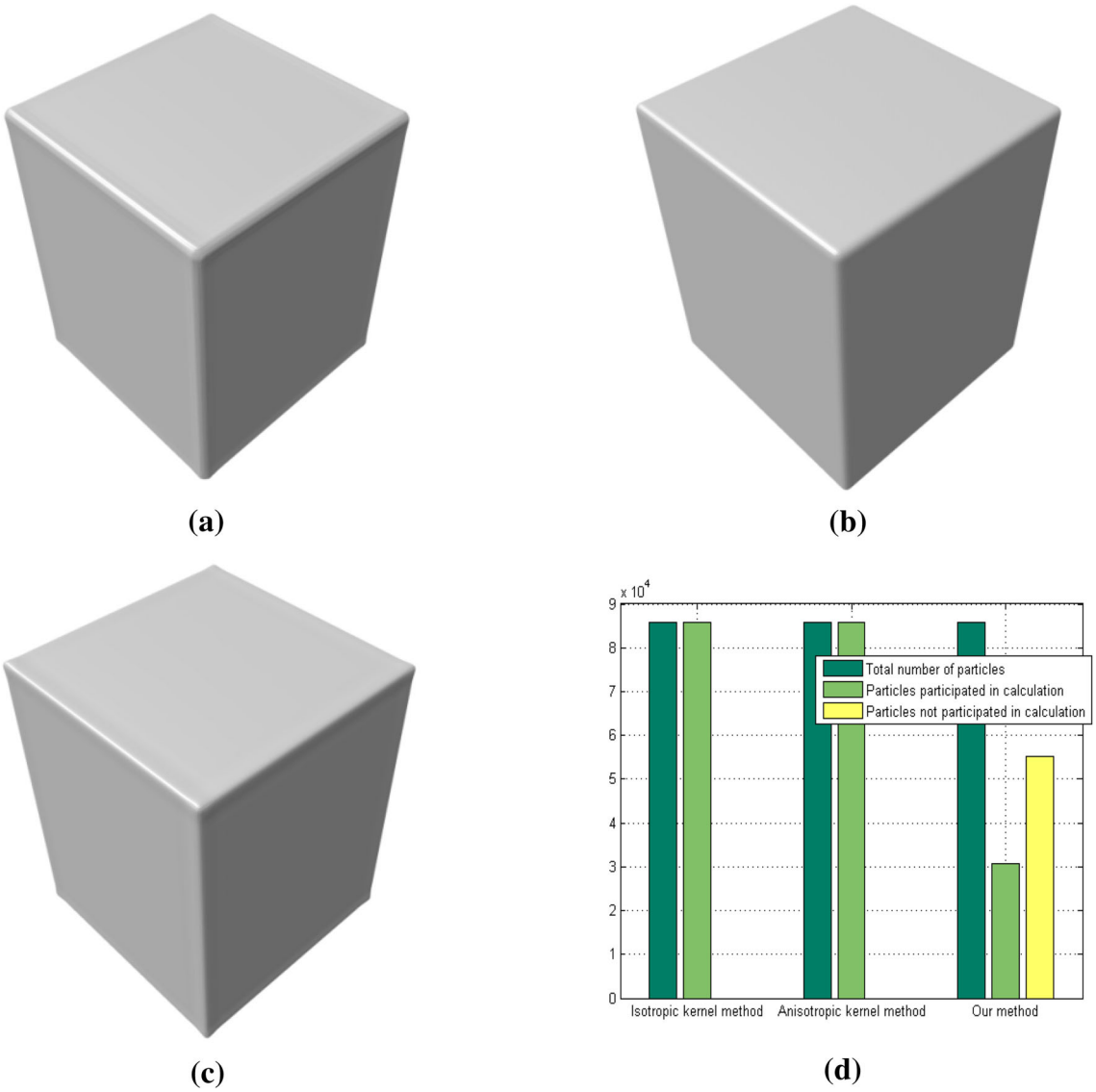


Fig. 8 **a** The isotropic kernels method, **b** the anisotropic kernels method, **c** our method and **d** comparison of particles number Surface reconstruction in comparison with cube fluids

trend of σ_1, σ_2 is more faster. Furthermore, σ_3/σ_1 is equal with σ_3/σ_2 at the center axis. This phenomenon is well suited for particle classification strategy. Through the experiments, we discover σ_3/σ_1 is greater than or equal to σ_3/σ_2 and if σ_3/σ_1 is very close to σ_3/σ_2 it is probably an internal particle. In other words, σ_3/σ_1 can be used to distinguish surface particles and internal particles, and σ_3/σ_2 can be used to classify internal particles.

From the viewpoint of the particle classification, we just need to separate the surface particles and internal particles. However, due to the non-uniform distribution of particles when using SPH method, reconstructing surfaces only with surface particles may be incomplete and invariably have bumps for lacking particles. It is necessary to classify the particles near the surface to compensate for lacking of surface particles. So, we can treat surface particles and near-surface particles as one category while internal particles as another category. It seems that the ratio σ_3/σ_1 shown in Fig. 4 fluctuates near the surface that can exactly provide assistance for the classification. Owing to require regarding surface particles and near-surface particles as one category, in the case of fluctuations, we merely analyze ratio threshold of boundary particles that will naturally draw the near-surface particles to surface particles. From the figure, we can find that the

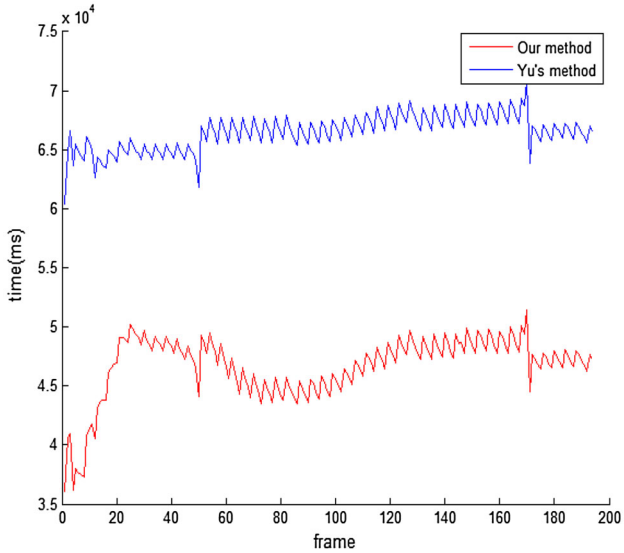


Fig. 9 Comparison of computation time in double breaking dam experiment, *blue line* expresses computing time of the anisotropic kernels method each frame, while *red line* shows computing time of our method each frame

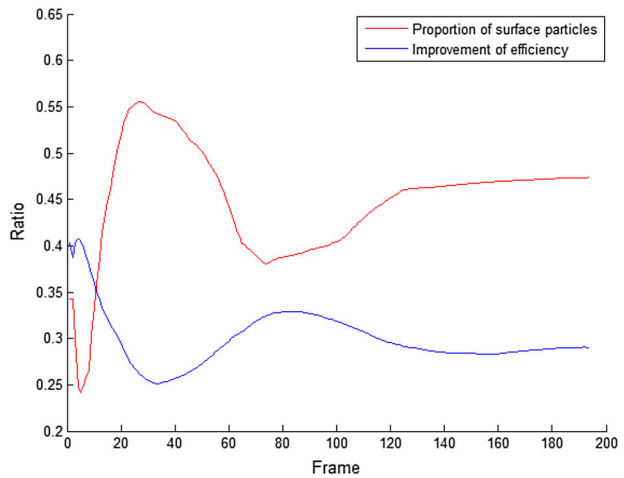


Fig. 10 Efficiency analysis for double breaking dam experiment, the *red line* implies surface particles proportion of all particles each frame, the *blue line* represents the increasing rate of efficiency compared with anisotropic kernels method each frame

Table 1 Comparison of anisotropic kernels method and our method

Scene	Method	#p (K)	Total comp. time (s)	Speed up
Double breaking dam	Isotropic kernels	205	2744.1	–
	Anisotropic kernels	205	12903	–
	Our method	205	9059.4	1.42
1 cube fluid–rigid coupling	Anisotropic kernels	172	10310	–
	Our method	172	6344.7	1.62
16 cubes fluid–rigid coupling	Anisotropic kernels	272.8	28000	–
	Our method	272.8	16943	1.65

ratio of eigenvalues near the surface is slightly larger than the surface particles. Figure 4 confirms the above ideas, when we have chosen the minimum ratio of boundary particles as threshold, near-surface particles can be separated from the internal particles and served as surface particles.

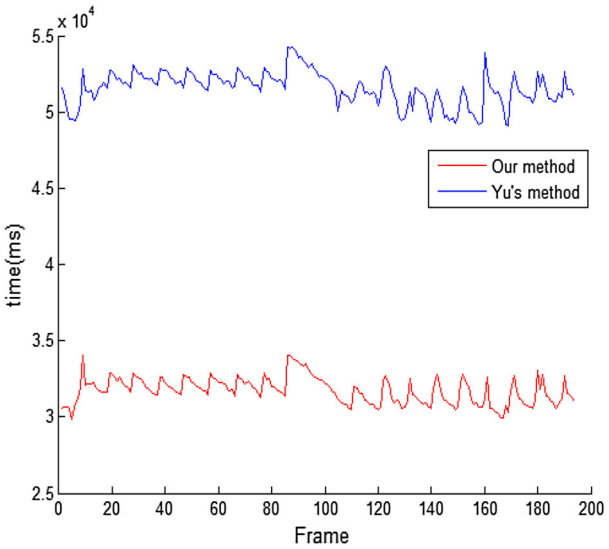


Fig. 11 Comparison of computation time in 1 cube fluid–rigid coupling experiment, *blue line* expresses computing time of the anisotropic kernels method each frame, while *red line* shows computing time of our method each frame

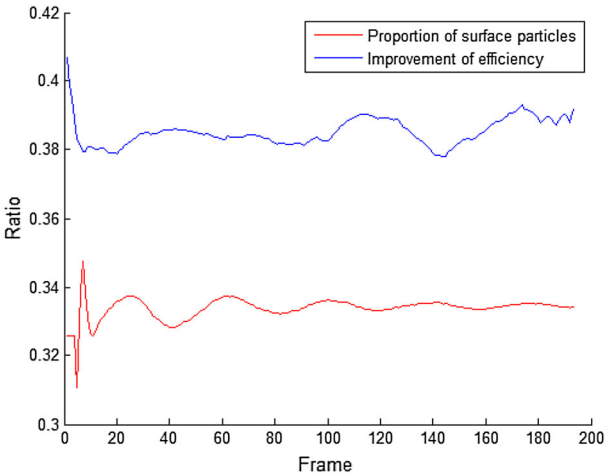


Fig. 12 Efficiency analysis for 1 cube fluid–rigid coupling experiment, the *red line* implies surface particles proportion of all particles each frame, the *blue line* represents the increasing rate of efficiency compared with anisotropic kernels method each frame

Figure 5 shows another scene of cube that we analyzed. Since the analysis is done on the sphere, we directly provide the cube's final curve of eigenvalues ratio σ_3/σ_1. The picture shows the similar results with sphere, so we can select the minimum eigenvalues ratio of boundary particles as the threshold to classify surface particles and near-surface particles with internal particles.

$$\text{particle\_type} = \begin{cases} \text{surface\_particle} & \sigma_3/\sigma_1 > \text{threshold}_s \\ \text{inner\_particle} & \text{otherwise} \end{cases} \tag{36}$$

where threshold_s is $\inf\{\sigma_3/\sigma_1\}$. Particle distribution information is hardly to be known in advance, so it is impossible to distinguish the boundary particles and internal particles. But by the previous analysis, it is not necessary to choose very accurate threshold, so we select $threshold_s = 1.1$ in this article. Due to the weak compressibility of the fluid, the analysis of cubic and spherical fluids for selecting threshold can expand to other form of fluids.

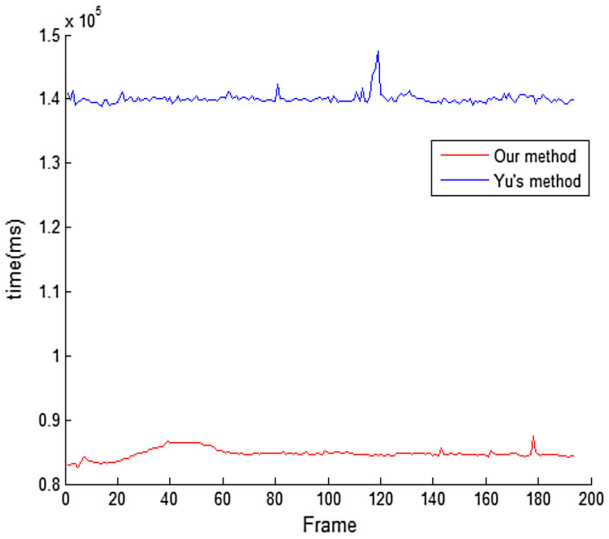


Fig. 13 Comparison of computation time in 16 cubes fluid–rigid coupling experiment, *blue line* expresses computing time of the anisotropic kernels method each frame, while *red line* shows computing time of our method each frame

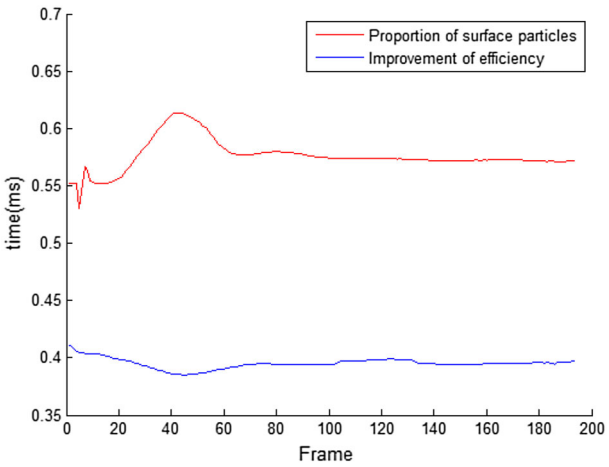


Fig. 14 Efficiency analysis for 16 cubes fluid–rigid coupling experiment, the *red line* implies surface particles proportion of all particles each frame, the *blue line* represents the increasing rate of efficiency compared with anisotropic kernels method each frame

After classification, we have to calculate the color field of particles and we can utilize Eq. (37) for surface and near-surface particles. We assign values for internal particles using formula (39). If we only calculate the color field of surface particles and do not handle internal particles, it will have inner and outer surfaces seen by the marching cubes algorithm, shown in Fig. 6. This situation would be difficult to render the fluid, so we use the following expression to assign the color field values for internal particles.

$$W_{\text{inner}} = \text{iso}/\text{neighbor\_num} \tag{37}$$

From the above formula, we can apparently find out that the expression has nothing to do with the distance and other variables, it only relates to the number of particle neighbors. As a consequence, the calculation is very simple and will not bring extra computational overhead. Moreover, to ensure the smoothness of the surface, Yu uses Gaussian smoothing of the formula (14) for all particles while we simply apply formula (14) for surface particles that further enhance the efficiency.

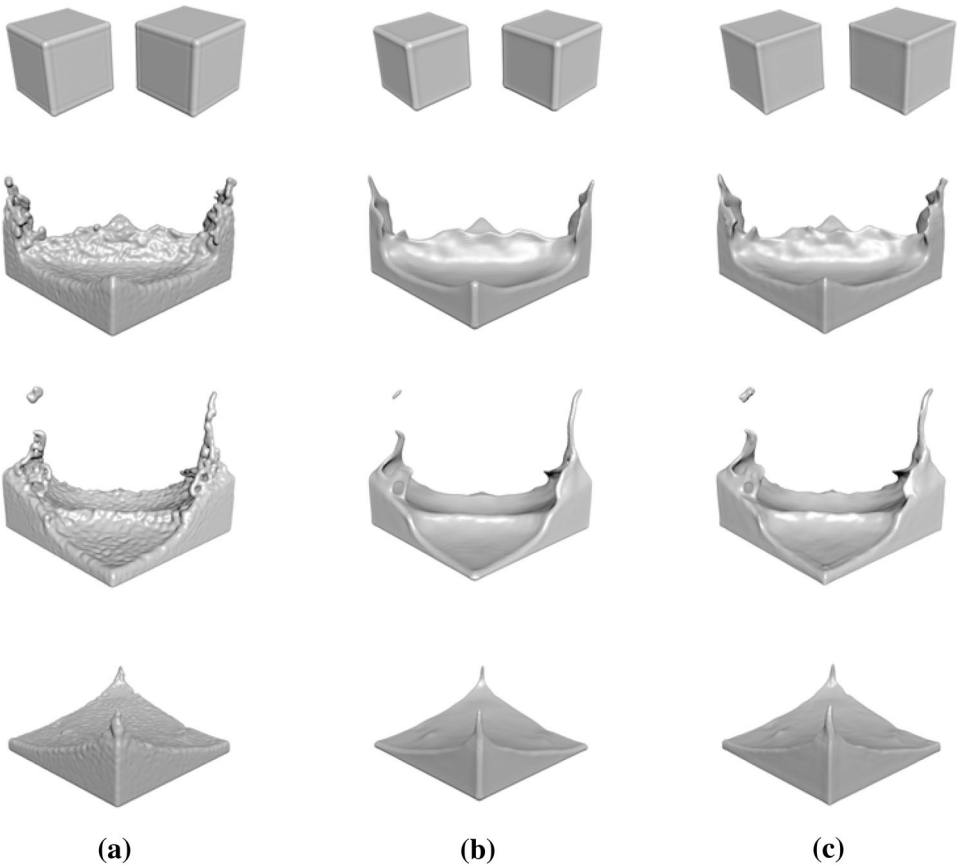


Fig. 15 **a** The isotropic kernels method, **b** the anisotropic kernels method and **c** our method surface reconstruction of double breaking dam experiment

5 Implementation and results

The experiments verified the validity of the algorithms in this article. The simulation is performed on a quad-core Intel i5-3470 (6M Cache, 3.2 GHz) with 8GB memory. Bullet is used for simulating rigid objects and OpenMP is used for parallelize particle computations. The simulation and surface reconstruction are actualized with C++ language and multi-threading technology. The searching process of approaching particle in simulation algorithm uses space background grid to carry out Hash lookup. Surface reconstruction employs the anisotropic kernel function to construct color field that is proposed in Sect. 4, and then uses the marching cubes algorithm to reconstruct surfaces, where singular value decomposition of the matrix is based on the JAMA and TNT open source maths library of NIST. The real-time simulation and results of surface structure is displayed by OpenGL 3D Graphics Library, while the video is recorded by OpenCV Library and rendering is implemented in Blender v2.73.

A set of surface reconstruction in breaking dam experiments is shown in Fig. 7. The figure shows, respectively, the isotropic kernel method of Muller (2003), the anisotropic kernel method of Yu and Turk (2013) and our approach proposed in this paper. The histogram displays the number of particles involved in the calculation. In this set of experiments, the isotropic kernel method takes 1363 ms, but the surface is rough. The anisotropic kernel method takes 1363 ms and the surface seen from the figure is very smooth. While the method described in this article takes 5398 ms, it is not different from the sense of reality with Yu's anisotropic kernel approach. As can be seen from the chart, this scene has a total of 22,397 points after removing 1399 scattered and solitary points.

Methods of isotropic kernel and anisotropic kernel all use 20,998 points to reconstruct the surfaces, while we merely use 16,327 points in surface reconstruction after the classification of surface points. From the viewpoint of time efficiency, we gain relative 11 % efficiency improvement to anisotropic kernel method.

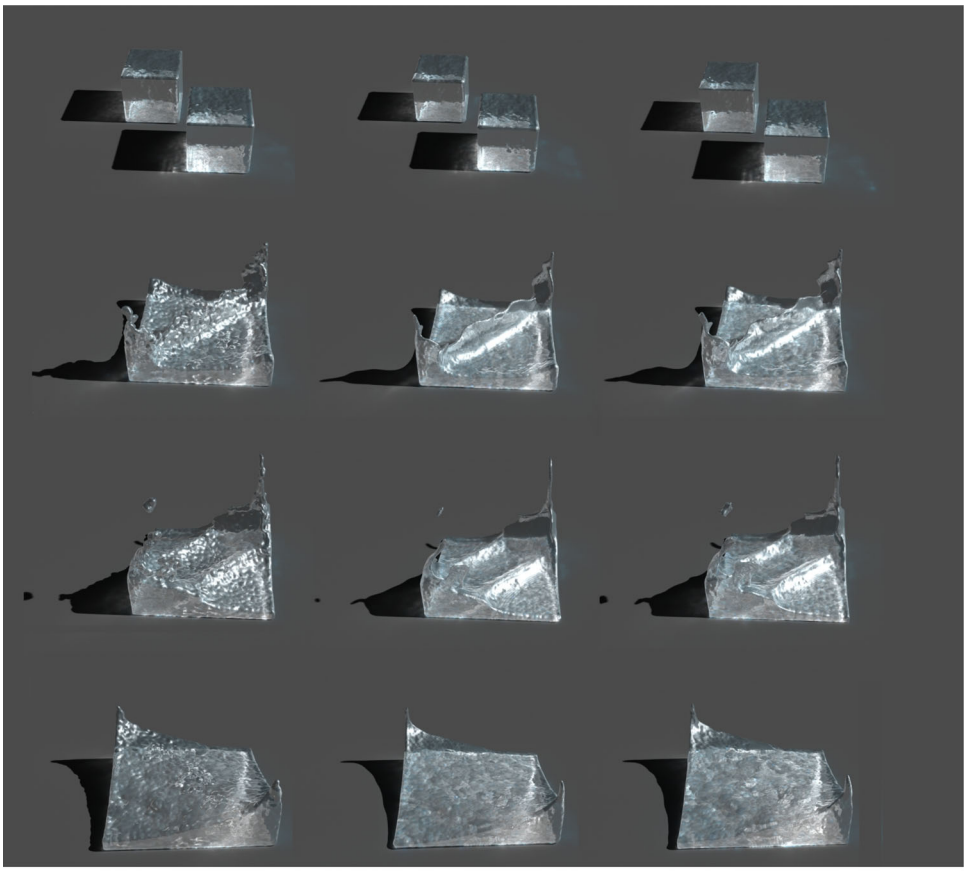


Fig. 16 **a** The isotropic kernels method, **b** the anisotropic kernels method and **c** our method rendering comparison of double breaking dam experiment

As the number of particles is small in this experiment and particles are mostly surface particles, the efficiency gained is comparatively low.

Figure 8 shows the surface reconstruction effects of the cube and a total of 85,731 particles in experiments. Muller's isotropic kernels method (Muller et al. 2003) costs 5514 ms, Yu's anisotropic kernels method (Yu and Turk 2013) uses 24,660 ms, while we use 30,598 vertices as surface points in surface reconstruction, cost 14,320 ms and obtain 41.9 % efficiency improvement. This scenario has more particles, thus obtaining a quite good efficiency. In addition, it can be discovered from the figure that our method produces more clear, sharp and realistic edges which suggests our method extraction of geometric features of surfaces better.

Figure 15 illustrates surface reconstruction results of isotropic kernels method, anisotropic kernel method and our method in double breaking dam experiment, this scenario employ 205082 particles. Figure 9 shows the computation time taken by each frame of anisotropic kernel method and our method throughout the double breaking dam experiment. As shown, we can see that the whole computation time of our method is lower than that of anisotropic kernel method and we get 1.42 times speedup throughout the experiment. Observed from Fig. 15, anisotropic kernel method and our approaches for constructing the surfaces are more smooth and realistic than the isotropic kernels method, nevertheless our algorithm saves more computing time and accesses to a larger efficiency gains than anisotropic kernel method. Figure 10 analyzes the efficiency promotion of our approach, the red line hints the proportion of surface particles to all the particles in each frame, the blue line represents the increasing rate of efficiency compared with anisotropic kernel method for each frame. As it is shown in the figure, the trends of the two lines are just opposite, that is efficiency can be greatly improved when having a small proportion of surface particles. This is fully consistent with the expected results and the whole process acquires a 30 percent enhancement of efficiency.

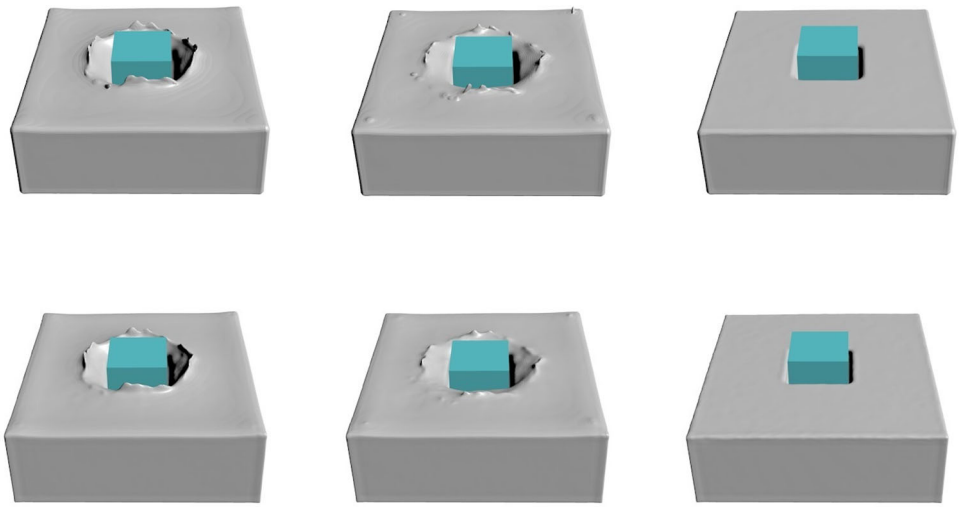


Fig. 17 Surface reconstruction of 1 cube fluid–rigid coupling experiment. *First row* is results of anisotropic kernels method; *Second row* is results of our method

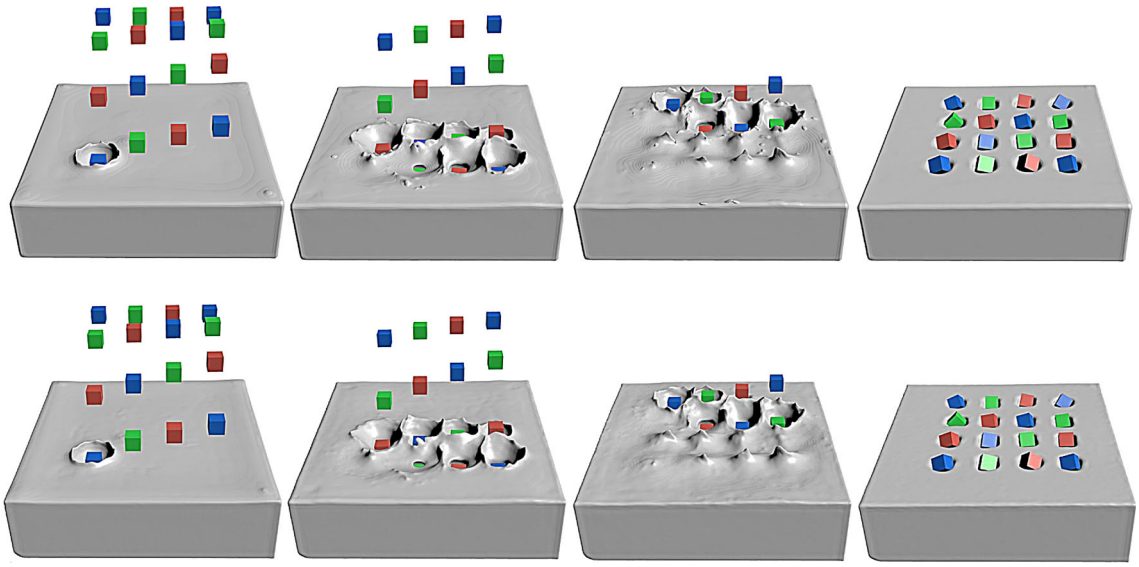


Fig. 18 Surface reconstruction of 16 cubes fluid–rigid coupling experiment. *First row* is results of anisotropic kernels method; *Second row* is results of our method

Table 1 shows comparison of anisotropic kernels method and our method in double breaking dam and fluid–rigid coupling scene. Figure 16 is the rendering comparison of three methods in double breaking dam. The rendering pictures chosen are the same frames as in Fig. 15. In this experiment, we can find that surfaces of water have very good realistic effects.

In addition, we test our algorithm in fluid–rigid coupling scene that also received good results. Figure 17 shows surface reconstruction results of anisotropic kernel method and our method in 1 cube fluid–rigid coupling experiment, this scenario employs 171,941 particles. Figure 11 displays the computation time cost by each frame of anisotropic kernel method and our method. We get 1.62 times speedup in this experiment. Like Figs. 10 and 12 analyzes the efficiency promotion of our approach, red line hints the proportion of surface particles to all the particles in each frame, blue line represents the increasing rate of efficiency compared with anisotropic kernel method and this experiment acquires a 38 % enhancement of efficiency.

Figure 18 illustrates surface reconstruction results of anisotropic kernel method and our method in 16 cubes fluid–rigid coupling experiment, this scenario uses 272,861 particles. Figure 13 shows the computation time of anisotropic kernel method and our method. In this experiment, we get 1.65 times speedup. Figure 14 analyzes the efficiency promotion of our approach, red line hints the proportion of surface particles, blue line represents the increasing rate of efficiency compared with anisotropic kernel method and this experiment acquires a 40 percent enhancement of efficiency.

6 Conclusions

An efficient anisotropic surface reconstruction algorithm for SPH fluid is proposed in this article. Primarily, we simplify the construction of the traditional anisotropic kernel function based on geometry. Then we divide particles into near-surface particles and internal particles according to the analysis of particles eigenvectors. In addition, the near-surface particles are involved in calculation when processing surface reconstruction, while the internal particles' color field values are directly assigned by means of its number of neighbor particles. Experimental results show that our method ensures smoothness and geometric characteristics of the fluid surfaces and enhances the calculation efficiency. In the experiments described, we received 30–40 % improvement on efficiency compared to Yu's anisotropic kernel algorithm in 2013. Efficiency improvements are related to the number of particles participating in the simulation. In most scenes, internal particles increase with the total number of particles. The more the number of particles are, the more efficiency the algorithm will gain. For that reason, the large-scale scenes can achieve higher speedup. Future work would be extending the proposed approach to large-scale scenarios and other SPH fluid solvers such as PCISPH or IISPH.

References

Adams B, Pauly M, Keiser R, Guibas LJ (2007) Adaptively sampled particle fluids. ACM Trans Graph, 26(3): Article 48

Akinci N, Ihmsen M, Akinci G, Solenthaler B (2012) Versatile rigid-fluid coupling for incompressible SPH. ACM Trans Graph, 31(4): Article 62

Becker M, Teschner M (2007) Weakly compressible SPH for free surface flows. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on computer animation. ACM & EG, San Diego, pp 209–217

Bhattacharya H, Gao Y, Bargteil AW (2011) A level-set method for skinning animated particle data. Symp Comput Anim 2(7):17–24

Blinn JF (1982) A generalization of algebraic surface drawing. ACM Trans Graph 1(3):235–256

Brochu T, Bridson R (2006) Fluid animation with explicit surface meshes. In: SCA proceedings of the ACM Siggraph/Eurographics symposium on computer animation. ACM & EG, Vienna. Poster session

Enright D, Fedkiw R, Ferziger J et al (2002) A hybrid particle level set method for improved interface capturing. J Comput Phys 183(1):83–116

Enright D, Losasso F, Fedkiw R (2005) A fast and accurate semi-Lagrangian particle level set method. Comput Struct 83(6):479–490

Hirt CW, Nichols BD (1981) Volume of fluid /VOF/ method for the dynamics of free boundaries. J Comput Phys 39(81):201–225

Koren Y, Carmel L (2003) Visualization of labeled data using linear transformations. In: Proceedings of IEEE information visualization. Seattle, Washington, p 16

Liu GR, Liu MB (2003) Smoothed particle hydrodynamics: a meshfree particle method. World Scientific, Singapore

Muller M, Charypar D, Gross M (2003) Particle-based fluid simulation for interactive applications. In: SCA proceedings of the ACM Siggraph/Eurographics symposium on computer animation. ACM & EG, San Diego, CA, pp 154–159

Muller M (2009) Fast and robust tracking of fluid surfaces. In: SCA Proceedings of the ACM Siggraph/Eurographics symposium on computer animation. ACM & EG, New Orleans, pp 237–245

Osher S, Fedkiw R (2002) Level set methods and dynamic implicit surfaces. Surfaces 44:77229–77240

Schechter H, Bridson R (2012) Ghost SPH for animating water. ACM Trans Graph, 31(4): Article 61

Williams B W (2008) Fluid surfaces reconstruction from particles. M.D. Dissertation, The University Of British Columbia

Wojtan C, Threy N, Gross M, Turk G (2010) Physics-inspired topology changes for thin fluid features. ACM Trans Graph 29(4):157–166

Wojtan C, Threy N, Gross M, Turk G (2009) Deforming meshes that split and merge. ACM Trans Graph**28**(3): Article 76

Yu J, Wojtan C, Turk G, Yap C (2012) Explicit mesh surfaces for particle based fluids. Comput Graph Forum 32(2 pt 4):815–824

Yu J, Turk G (2013) Reconstructing surfaces of particle-based fluids using anisotropic kernels. ACM Trans Graph, 32(1): Article 5

Zhu Y, Bridson R (2005) Animating sand as a fluid. ACM Trans Graph 24(3):965–972