Physically-based Animation of Water Pollutant Diffusion

Guijuan Zhang[1,2], Dianjie Lu[1,2], Hong Liu[1,2]
[1]School of Information Science and Engineering,
Shandong Normal University, Jinan, China
[2]Shandong Provincial Key Laboratory for Novel
Distributed Computer Software Technology, Jinan, China
Corresponding author: Dianjie Lu, ludianjie@sina.com

***Abstract*—We present a physically-based animation method for simulating the diffusion of water pollutant in this paper. Our method allows animating the dynamic evolution of biological pollutants (e.g., water hyacinth, algae) on large-scale water surface effectively. To enable the simulation of such a phenomenon in large scale virtual environment, we simplify the problem by considering the water pollutants as large patches or clusters of aquatic plants rather than each single aquatic plant. Since there exits obvious interface between the large patches of pollutants and the water surface, we model the evolution of the interface in our approach. We use 2D dynamic curves to represent the interface and the diffusion of the water pollutants can be regarded as the dynamic evolution and propagation of 2D curves. To alleviate the topological changes of 2D curves, we adopt a physically-based 2D level set model to animate the evolution and propagation of the interface. We build a level set equation to model the evolution of the interface. In addition, to handle the large scale virtual environment correctly in our physically-based level set model, an image-based 2D voxelization method is proposed in the paper. In the voxelization method, the virtual environment will be converted to boundary conditions when solving the level set equation. Finally, the water pollutants diffusion phenomenon is simulated on large scale water surface by merging the interface animation results as well as the large scale virtual environment. Animation results about the algae propagation phenomenon in Taihu Lake show that our method is intuitively to be implemented and very convenient to produce visually interesting results.**

***Index Terms*—water pollution; water pollutant diffusion; physically-based animation**

I. INTRODUCTION

Water pollution has become a global problem of our modern society. Biological pollution, as a common phenomenon of water pollution, has caused tremendously harm to the ecological stability of lakes and rivers all over the world. The wastes from industry, agriculture and even sewage treatment accelerate algae and water hyacinth growth, which ultimately results in exponential growth of water pollution (as shown in Figure 1). Thus, it is very important to carefully manage the water quality by study the phenomenon of aquatic plants diffusion.

However, modeling for the diffusion of aquatic plants on large-scale water surface is a challenging issue. Since these diffusion scenarios often occur in wide variety of contexts such as feature films, cartoons, games and propaganda on environmental protection, animation of the water pollutant diffusion can provide the intuitionistic information for the management of the water quality.

In the literature of water animation world, very few literatures focus on animating the water pollution phenomenon. Shi et al. take the first step toward the rendering of such phenomenon [1]. In this work, they take the optical properties of polluted water into consideration. A method is provided to obtain the optical properties which generally vary with the concentrations of pollutants. The proposed method focuses on underwater phenomenon and produces interesting results by employing the rendering techniques. In other works, the propagation phenomenon on water surface is considered as well as large-scale virtual environment. To animate texture advection on the water surface, Mihalef et al. propose the Marker Level Set (MLS) method. MLS mainly handles the dynamics of a liquid and its surface attributes [2]. In this method, surface markers are used to track the surface of the liquid. Then, MLS can handle the non-diffusively surface texture advection with these surface markers. The methods mentioned above are all physics-based. However, the costly overhead makes it virtually impossible to simulate the biological pollution phenomenon on large-scale water surface.

To address this problem, we propose an approach for animating the diffusion of biological pollution (e.g., algae, water hyacinth etc.) on large-scale water surface. In our method, we consider the dynamic evolution and propagation of interface between biological pollutants and the water surface. To this end, we use some assumptions and simplifications. In this paper, we only focus on the clusters or large patches of biological pollutants rather than each single aquatic plant. We build a physically-based 2D level set model to represent the interface. In order to set the boundary condition for the 2D level set model, we also present a 2D voxelization method in this paper. The method obtained the

---

Corresponding author: Dianjie Lu, ludianjie@sina.com

doi:10.4304/jcp.9.7.1657-1664

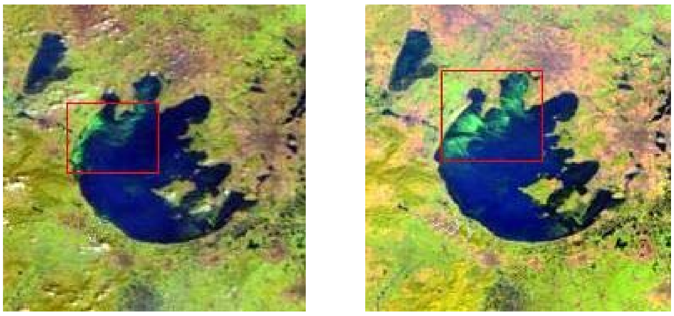


Figure 1. Satellite images show the propagation of the biological pollutants on Taihu

voxelization results from the top-view of the 3D virtual environment. Finally，we implement our method in the cases of algae diffusion in Taihu Lake. The results show that our method can animate the diffusion phenomenon of water pollutants effectively. The contributions of this paper are listed as follows.

(1) We present an animation method for simulating the evolution and diffusion of water pollutants in large-scale virtual environment.

(2) We give some assumptions and simplifications to enable the large-scale animation and build a physically-based diffusion model to animate the diffusion phenomena of water pollutants. The diffusion model is built upon 2D level set equation and can dynamically evolve the interface between biological pollutants and water surface effectively.

(3) We present a 2D voxelization method to provide the boundary condition for the 2D level set diffusion model so that the interface can correctly evolve in large-scale 3D virtual environment.

II. RELATED WORK

Diffusion phenomenon has been extensively investigated in computer animation in the past decades [3-5]. For example, in [3][4], the authors animate the fluid propagation in virtual 3D environment. In [5], Auer et al. animates the flow propagation on dynamic surfaces. These methods are all are physics-based and provide nice visual effect. However, it is hard to execute the large-scale animation for these methods. The reason is that the computational resources are relatively limited compared with the increasing animation scale. The costly overhead makes it impossible to get the animation results.

The particle level set method is the most common method to animate water diffusion phenomenon in fluid animation world. With this method, the dynamical water surface can be represented. In addition, faster methods are proposed to animate water diffusion in rivers and oceans [7][8]. Furthermore, some researchers present fluid control methods to guide the direction of water diffusion [9-11]. For example, in [9], the particle-based control method is used to account for the water propagation. They define the external forces for control particles via shape difference and velocity difference. In [10] and [11], Zhang et al. also control the propagation of the liquid motion by transportation method or two-layer model. In addition, GPU based methods are also presented to speed up the computation of the physically based model in fluid animation [12,13]. However, these methods haven’t taken the propagation of the objects on liquid surface into consideration. Moreover, these physics-based methods are too expensive to be used in large-scale animation.

Large-scale water animation is gaining growing attention recently. A large amount of work has been done for it such as the FFT based empirical and mathematical description of the water surface [14-17]. In these methods, the water surface are represented by a height field that formulated by FFT. In addition, Perlin noise is also involved to get the height field representation of water surface [17]. These methods play an important role in large-scale water surface animation. However, the propagation of the objects on liquid surface has not been considered.

Our work has a close relation with [1] and [2]. Shi et al. [1] render rendered the polluted water while Mihalef et al. [2] animates the texture advection on liquid surface. However, the method in [1] is not suitable for animating the evolution and propagation of the biological pollutants on large-scale water surface since it focuses on the rendering model of the polluted water and simulates the under water pollution phenomenon. As for the method in [2], it is also infeasible for large-scale animation because of the costly overhead of the physically-based animation.

To address the above problems and achieve the goal of this paper, we present an animation method for simulating the evolution and diffusion phenomenon of biological pollutants on large-scale water surface. Our method can animate large-scale phenomenon and produce the interesting diffusion and evolution results on water surface.

III. OVER VIEW

The goal of this paper is to allow animators to obtain interesting results of biological pollutants propagating on large-scale water surface. To achieve this goal, we introduce some assumptions and simplifications in this paper. Under these assumptions, we only focus on the large patches of biological pollutants and model the dynamic propagation of the interface between the pollutants and water surface. It can well represent the dynamic evolution of water pollutants effectively. Our method is performed in three steps as shown in Figure 2.

In the first step, we pre-process the 3D virtual environment in order to provide boundary condition for the physically-based diffusion model. Since we focus on

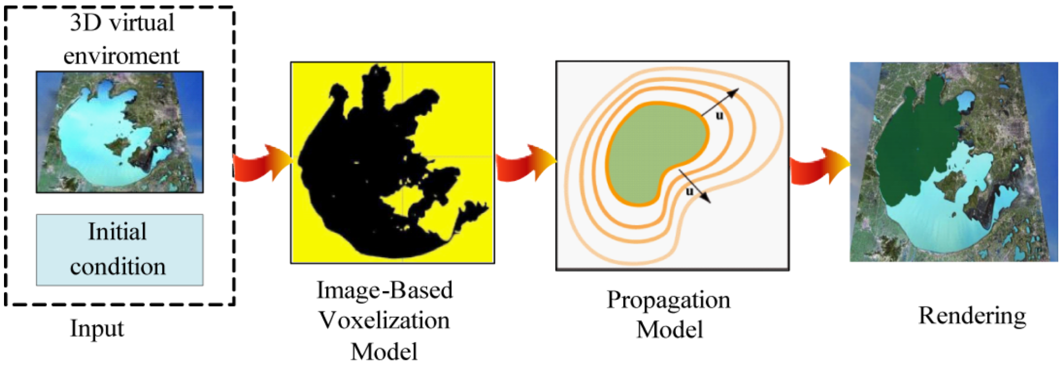


Figure 2. The framework of our method.

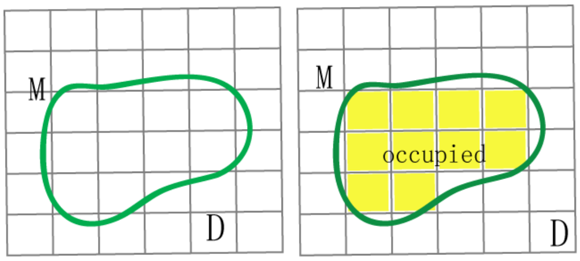


Figure 3. A model M enclosed by a 2D curve and its voxelization results. Note that the interior region of the closed curve is finally converted into a set of voxels denoted by yellow cells in computational domain D.

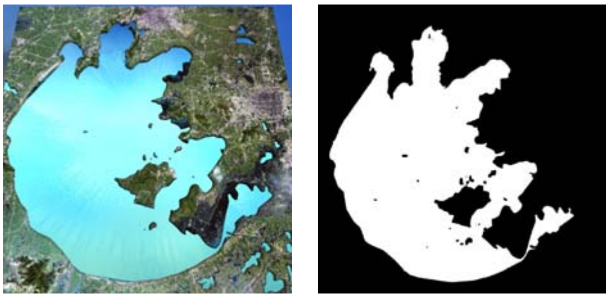


Figure 4. The top view image of the 3D virtual environment and the segmentation result of the image. Note that the white color denotes water region and black color represents the obstacles.

animating large-scale water pollution, the pre-processing step should manipulate the large 3D virtual environment effectively. To this end, we propose an image-based voxelization method to get the boundary condition for computation. The voxelization method handles the top image of the 3D virtual environment and outputs the voxelization function for each grid cell in the computational domain.

Next, we build a physically-based diffusion model to animate the evolution of the water pollutants in large scale virtual environment. The model uses 2D level set equation to represent the dynamic evolution of the pollutants interface. In the 2D level set equation, we consider both the diffusion of the pollutants and the drifting motion of the pollutants. The physically-based diffusion model allows us to animate the large-scale water pollution with high degree of visual realism effectively. The level set equation is solved according to semi-Lagrange method and the boundary condition of the equation is set according to the voxelization results obtained from the last step. Note that, if we capture the diffusion velocity and the drifting velocity of water pollutants, the method can also be used to visualize the pollution phenomenon in applications.

Finally, we render the obtained data to get the visualization result of the dynamic diffusion. The pollutants diffusion images are merged with the virtual environment. As shown in Figure 2, the final animation results contain two unique parts, the virtual environment and the dynamic diffusion of the water pollutants respectively.

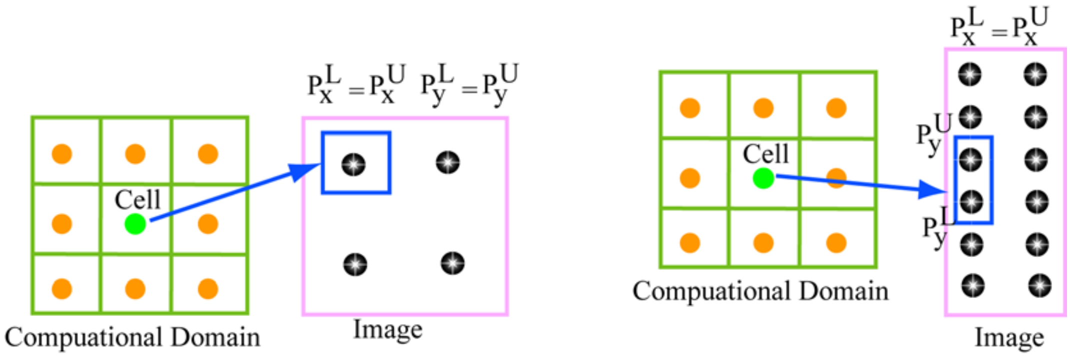
IV IMAGE-BASED VOXELIZATION METHOD

As an essential condition in water diffusion animation, the environment boundary should be correctly defined. To achieve this, we employ the voxelization technique. Since we only consider the diffusion phenomenon on water surface, we can make the top view image (the view that shows water surface) of the 3D virtual environment as the input. In this paper, we propose an image-based 2D voxelization method to get the voxelization results.

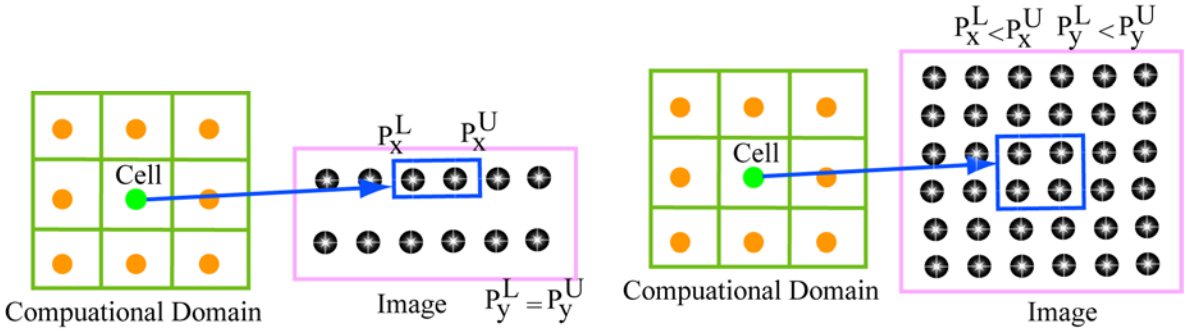
In traditional voxelization method [18], the continuous geometric representation of the object (e.g. 3D mesh model) is converted to a set of voxels that best approximates the object as shown in Figure 3. The generated data base of the discrete digitization of the object will be used for setting boundary condition in next section.

Intuitively, a proper voxelization results contains all voxels that are occupied (totally or partially) by the model. In this paper, we focus on the patches that composed by biological pollutants and the patches are evolving on the water surface. As a result, we only consider the subsurface composed by the water surface. So the 3D voxelization problem is converted into a 2D one in this paper. We compute a voxelization function for each grid cell in our computational domain according to our 2D voxelization method.

Our voxelization process can be divided into two steps. In the first step, we rotate the 3D model to the top view of the virtual environment which can show the water surface well. Then, the Grabcut [19] method can be introduced to get the segmentation results since the water surface shows different color from the remaining part. As



(a) $I_x \le R_x$ and $I_y \le R_y$ (b) $I_x \le R_x$ and $I_y > R_y$



(c) $I_x > R_x$ and $I_y \le R_y$ (d) $I_x > R_x$ and $I_y > R_y$

Figure 5. The mapping between the computational domain and the input image.

we can see in Figure 4, the foreground and background pixels are correctly segmented. The image is also colored by white and black color which denotes the water region and the obstacles respectively. So far, we can get the input image of the top view.

In the second step, we calculate the voxelization function. We define the grid resolution of the computational domain D as $R_x \times R_y$ and the input image resolution as $I_x \times I_y$. The pixel $p_{i,j}$ with $T_{i,j}$ can be tagged by

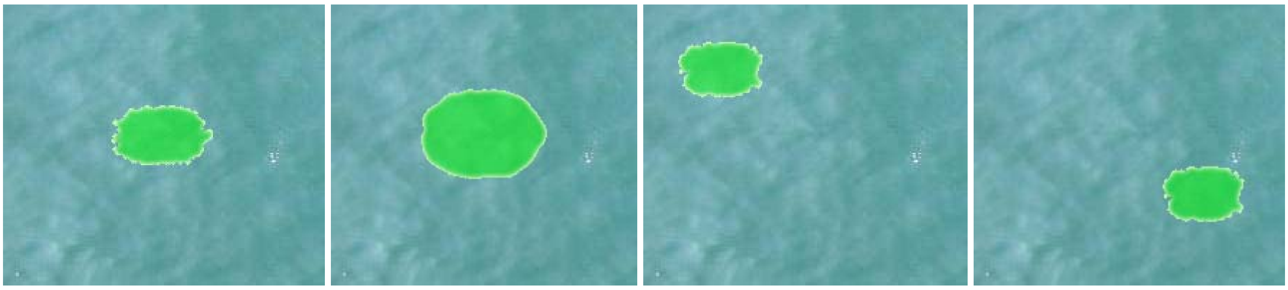
$$T(i,j) = \begin{cases} 0 & \text{if } p_{i,j} \in foreground \\ 1 & \text{if } p_{i,j} \in background \end{cases}, \qquad (1)$$

where $i \le I_x$ and $j \le I_y$. As for a cell (x,y) in the computational domain D, we first find out a pixel patch in the image that contains all pixels corresponding to the cell (x,y). The pixel patch is an integer set and can be obtained by computing its lower and upper bound respectively.

Denote P_x^L and P_y^L as the lower bound of the row indices and column indices, P_x^U and P_y^U as the upper bound of the row indices and column indices. For example, if $I_x \le R_x$, we can easily get $P_x^L = I_x / R_x \times x$ and $P_x^U = I_x / R_x \times x$. Same method can be used to compute the lower and upper bound of column indices. If $I_y \le R_y$, $P_y^L = I_y / R_y \times y$ and $P_y^U = I_y / R_y \times y$ can be obtained.

Otherwise, there exist more than one pixels in the set that corresponding to the cell (x,y) in the computational domain. As for the row indices, we can get $P_x^L = I_x / R_x \times x$ and $P_x^U = I_x / R_x \times (x+1)$. Similarly, we can also calculate the column indices P_y^L and P_y^U easily. Specifically, $P_y^L = I_y / R_y \times y$ and $P_y^U = I_y / R_y \times (y+1)$.

Let $S_x = P_x^U - P_x^L + 1$ and $S_y = P_y^U - P_y^L + 1$. So the size of the patch is $S_x \times S_y$. As shown in Figure 5, the patch has four different cases. First, when $I_x \le R_x$ and $I_y \le R_y$, the size of the patch is 1. It means that one or more cells are mapped to the same pixel in the input image as shown in Figure 5 (a). The value of voxelization function in the current grid cell can be calculated with the corresponding pixel. When $I_x \le R_x$ and $I_y > R_y$, the size of the patch is $1 \times I_y / R_y$. Then, we can get the voxelization of the current grid cell through averaging the results obtained from each pixel in the patch (shown in Figure 5(b)). The other two cases are demonstrated in Figure 5(c) and (d). When $I_x > R_x$ and $I_y \le R_y$, the size of the patch is $I_x / R_x \times 1$. When $I_x > R_x$ and $I_y > R_y$, the



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

Figure 7. The diffusion phenomena of biological pollutants

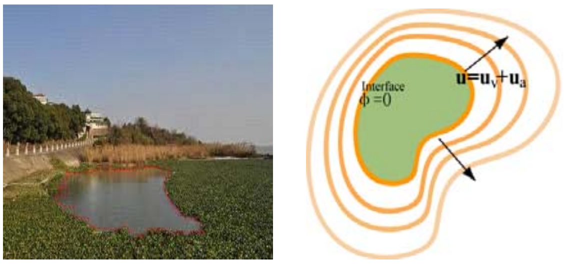


Figure 6. The interface between the pollutants and the water surface. The left image shows the interface exist in a real world. We highlight it with a red curve. The right image gives the representation of the implicite function. Note that Φ=0 denotes the interface. The exterior brown curves are the propagation results of the interface at different time.

patch size is $\left(I_X/R_X\right)\times\left(I_Y/R_Y\right)$. With the mapping between the computational domain and the input image, we can compute the voxelization value for a specific grid cell.

To determine the voxelization result of each cell in the computational domain, we first calculate the proportion of background pixels in the pixel patch. If more than half of the pixels in the patch belong to background, we tag the cell as an occupied cell. It denotes that the cell is occupied by the virtual environment. So, $F(x,y)$ for each cell can be computed by

$$F(x,y)=\sum_{i=1}^{S_x}\sum_{j=1}^{S_y}\frac{T(i,j)}{S_x\times S_y}, \tag{2}$$

where T(i, j) is defined in equation (1). (x,y) is a grid cell, $S_x=P_x^U-P_x^L+1$ and $S_y=P_y^U-P_y^L+1$ is the size of the image patch that corresponds to the grid cell (x,y). Finally, the voxelization function of each grid cell (x,y) is

$$V(x,y)=\begin{cases}1 & F(x,y)\geq 0.5\\ 0 & else\end{cases}. \tag{3}$$

V DIFFUSION MODEL OF THE POLLUTANTS

A. Physically-based Level Set Model

To model the diffusion phenomenon on large-scale water surface, some assumptions and simplifications are given as follows.

We assume that the diffusion of biological pollutants is viewed in the distance. It is reasonable because we are more interested in the global distribution and evolution of the biological pollutants. As a result, we can assume that the water surface is a static plane. In addition, we only focus on the biological pollutants that dynamically evolved on water surface rather than underneath. Biological pollutants are composed by aquatic plants (e.g., algae, water hyacinth etc), so there exists an obvious interface between the pollutants and water (see Figure 6). As a result, we model the evolution and diffusion of the interface to simplify the problem in this paper.

We adopt a physically-based method to simulate the evolution and diffusion of the water pollutants. To this end, we use the isocontour ($\phi=0$) of an implicit function ϕ to represent the interface between biological pollutants and water surface. It is demonstrate in the right image of Figure 6. Here, ϕ is the signed distance function with $|\nabla\phi|=1$. So the region of $\phi\leq 0$ denotes pollutants and the region of $\phi>0$ represents the water surface without pollutants covering on it. According to 2D level set method, the dynamic evolution of the interface is

$$\frac{\partial\phi}{\partial t}+\boldsymbol{u}\cdot\nabla\phi=0 \tag{4}$$

where $\boldsymbol{u}$ is the evolution velocity, and $\nabla=\left(\partial/\partial x,\partial/\partial y,\partial/\partial z\right)$ is the gradient operator. Given a velocity field, the level set equation allows us to compute the evolution and diffusion of water pollutants.

The velocity $\boldsymbol{u}=\boldsymbol{u}_n+\boldsymbol{u}_a$ includes two different parts. $\boldsymbol{u}_n$ is the growing velocity of the biological aquatic plant. It points to the normal of the interface and denotes the extension velocity of the plant growing. The normal of the interface can be approximated easily since we use the implicit function ϕ to represent it. Here, we denoted it as $\boldsymbol{n}=\nabla\phi/|\nabla\phi|$. $\boldsymbol{u}_a$ is the advection velocity which represents the biological pollutants floating on the water surface.

Figure 7 shows the diffusion results of water pollutants with different settings in equation (4). In Figure 7(a)(b), only $\boldsymbol{u}_n$ is considered while in Figure 7(c)(d), only $\boldsymbol{u}_a$ is considered. The results demonstrate difference roles of $\boldsymbol{u}_n$ and $\boldsymbol{u}_a$. Apparently, $\boldsymbol{u}_n$ controls the growing of the

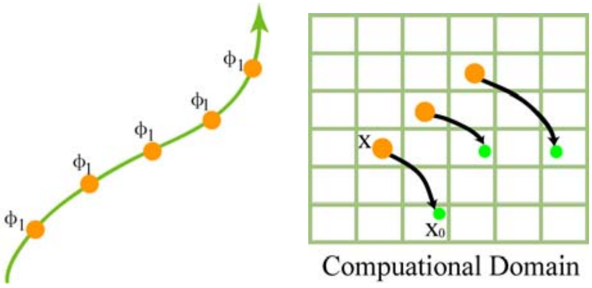


Figure 8. Semi-Lagrange method.

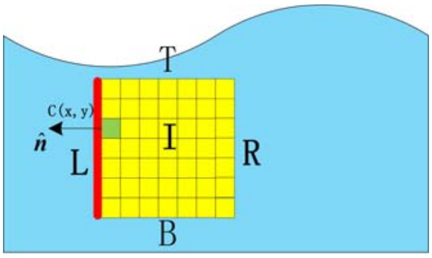


Figure 9. Setting the boundary condition

water pollutants and $\boldsymbol{u}_a$ controls floating phenomenon of the pollutants.

B. Solve the Level Set Equation

Equation (4) is essentially an advection equation and we use semi-Lagrange advection method to solve it. In the equation, the implicit function value of each grid point is moved with the velocity field. As shown in the left image of Figure 8, the value ϕ on the advection path is unchanged during animation. So we can trace the value back along the path to compute the implicit function value of current grid point. Let $\boldsymbol{x}$ be the grid position in current step and $\boldsymbol{x}_0$ be the position that the grid point locates in the last time step. See the right image of Figure 8, given a grid point $\boldsymbol{x}$, we track back along the advection path to get the point $\boldsymbol{x}_0$. After interpolating the implicit function value at $\boldsymbol{x}_0$, we can obtained the value of ϕ for the grid point $\boldsymbol{x}$. So the new value after advection equals to the value along the velocity direction a time Δt ago. We interpolate to get the ϕ value

$$\begin{aligned} \phi^{t+\Delta t}(\boldsymbol{x}) &= \phi^{t}(\boldsymbol{x}_0) \\ \boldsymbol{x}_0 &= \boldsymbol{x} - \boldsymbol{u}^{t+\Delta t}(\boldsymbol{x})\Delta t \end{aligned} \tag{5}$$

where $t+\Delta t$ is current time step, t is the last time step, $\boldsymbol{x}$ is the grid position in current step and $\boldsymbol{u}^{t+\Delta t}(\boldsymbol{x})$ is the velocity at $\boldsymbol{x}$ in current time step, $\boldsymbol{x}_0 = \boldsymbol{x} - \boldsymbol{u}^{t+\Delta t}(\boldsymbol{x})\Delta t$ denotes the position that the current grid point locates in the last time step.

C. Setting the Boundary Condition

The above subsection only considers what's happening in the interior of the computational domain. To solve equation (4) correctly, we are also required to consider the boundary condition because we simulate the phenomenon in a virtual environment. So we consider the solid wall boundary condition for computing the diffusion of the pollutants. Setting the solid wall boundary condition is necessary because we assume that the pollutants will not be flowing into the solid or out of it. According to the boundary condition of the solid wall, the velocity of the pollutants diffusion should be constrained by

$$\boldsymbol{u}\cdot\hat{\boldsymbol{n}} = \boldsymbol{u}_{solid}\cdot\hat{\boldsymbol{n}} \tag{6}$$

where $\boldsymbol{u}$ is the diffusion velocity of water pollutants in equation (4), $\hat{\boldsymbol{n}}$ is the normal of the solid wall, $\boldsymbol{u}_{solid}$ is the velocity of the solid wall.

When setting boundary condition for equation (4), we first let $\boldsymbol{u} = \boldsymbol{u}_N + \boldsymbol{u}_T$. Note that $\boldsymbol{u}_N = \boldsymbol{u}\cdot\hat{\boldsymbol{n}}$ denotes the velocity component that normal to the solid wall and $\boldsymbol{u}_T = \boldsymbol{u} - \boldsymbol{u}_N$ denotes the tangential component of $\boldsymbol{u}$. There are two cases when setting the boundary condition for equation (4).

In the first case, we consider the grid cells that locate in the interior of the solid wall. In this condition, we set the velocity value for a grid cell (i,j) that is occupied by the virtual environment, that is, $V(i,j)=1$. If all the neighbor's cells of (i,j) are occupied by the virtual environment

$$\begin{aligned} &V(i-1,j)=1 \quad and \\ &V(i+1,j)=1 \quad and \\ &V(i,j-1)=1 \quad and \\ &V(i,j+1)=1 \end{aligned}, \tag{7}$$

then we can conclude that the cell (i,j) locates in the interior of the solid wall. We will set $\boldsymbol{u}_{i,j} = \boldsymbol{u}_{i,j}^{solid} = 0$ here.

In the second case, we consider the grid cells on the boundary of solid wall. This is the toughest part in boundary setting operation. In this condition, $V(i,j)=1$ and there is at least one neighbor of the cell (i,j) is occupied by water surface, that is

$$\begin{aligned} &V(i-1,j)=0 \quad or \\ &V(i+1,j)=0 \quad or \\ &V(i,j-1)=0 \quad or \\ &V(i,j+1)=0 \end{aligned}. \tag{8}$$

To set boundary condition, we consider the normal component $\boldsymbol{u}_N = \boldsymbol{u}\cdot\hat{\boldsymbol{n}}$ and the tangential component $\boldsymbol{u}_T = \boldsymbol{u} - \boldsymbol{u}_N$ respectively. As for the tangential component $\boldsymbol{u}_T$, the slip boundary condition is used here. As a result, $\boldsymbol{u}_T$ remains unchanged when setting the boundary condition. Differently, the normal component will be set $\boldsymbol{u}\cdot\hat{\boldsymbol{n}} = \boldsymbol{u}_{solid}\cdot\hat{\boldsymbol{n}}$ according to equation (6). Specifically, Figure 9 gives an example. The solid wall

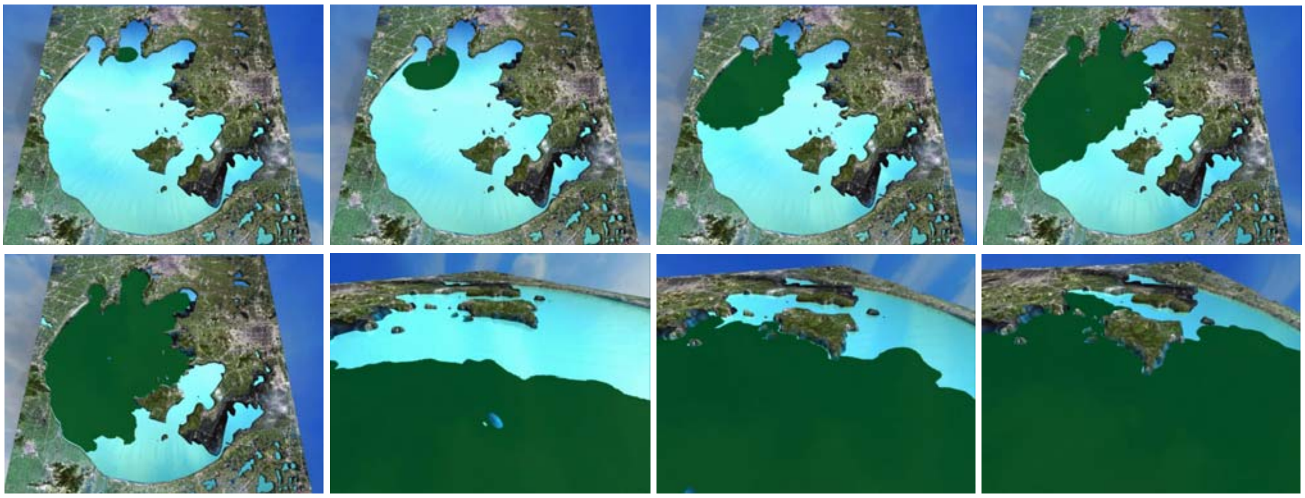


Figure 10. A sequence of animation results. We animate algae propagation on Taihu Lake. Note that the region covered by algae is represented by green color. The last three images show the results with different camera position.

(virtual environment) is denoted by yellow color. The red line is the boundary between water surface and the solid wall. We use L, B, R, T, I to represent the left boundary, bottom boundary, right boundary, top boundary and the interior of the solid wall respectively. Let $\boldsymbol{u}_{i,j}=\left(u_{i,j},v_{i,j}\right)$ and $\boldsymbol{u}_{i,j}^{solid}=\left(u_{i,j}^{solid},v_{i,j}^{solid}\right)$. For a cell (i,j) in the I region, we set $\boldsymbol{u}_{i,j}=\boldsymbol{u}_{i,j}^{solid}$ ($u_{i,j}=u_{i,j}^{solid}$ and $v_{i,j}=v_{i,j}^{solid}$). For the cells locating at the four boundaries of solid wall, the normal component of the velocity should be set with the boundary condition $\boldsymbol{u}\cdot\hat{\boldsymbol{n}}=\boldsymbol{u}_{solid}\cdot\hat{\boldsymbol{n}}$ while the tangetial component of the velocity remains unchanged. For example, for a cell (i,j) in the L region (left boundary), we will set $u_{i,j}=2u_{i,j}^{solid}-u_{i-1,j}$ so as to ensure that the velocity on the left bound of the solid wall is $u=\left(u_{i,j}+u_{i-1,j}\right)/2=u^{solid}$. Differently, the tangential velocity value $v_{i,j}$ will remain unchanged due to the slip boundary condition. Similarly, for a cell (i,j) in the R region (right boundary), we will set $u_{i,j}=2u_{i,j}^{solid}-u_{i+1,j}$ and $v_{i,j}$ remains unchanged. For the cell (i,j) locates in the T region (top boundary), we set $v_{i,j}=2v_{i,j}^{solid}-v_{i,j+1}$ and $u_{i,j}$ remains unchanged while for a cell in the B region (bottom boundary), we set $v_{i,j}=2v_{i,j}^{solid}-v_{i,j-1}$ and $u_{i,j}$ remains the same.

After obtaining the interface evolution results, we rendered it to an image with alpha channel. Note that the uncovered region of the water surface is set transparency in the image while the other region is set with a predefined texture (or color). Finally, the image that contains transparency information will be used as the 2D texture when rending the diffusion results.

VI RESULTS

We give an example of algae diffusion in this section. All the results in this paper are gathered on a PC with 2.6 GHz Intel Core 2 Duo CPU and 2GB memory. The 3D virtual environment is rendered using Povray library, which is available from http://www.povray.org/.

Figure 10 shows a sequence of animation results. The virtual 3D environment is built from GIS information of Taihu Lake. The diffusion of the biological pollutants is computed from the 2D level set equation. To make the computational domain in an available size, we set the 2D grid resolution as 1000×1000. We set $|\boldsymbol{u}_n|=0.02$ and $|\boldsymbol{u}_a|=0.001$ in this example.

The voxelization result is used as boundary condition in this example. We can see that when the interface propagating, it can avoid solid obstacles (e.g., island) well. By adjusting the value of $|\boldsymbol{u}_n|$ and $\boldsymbol{u}_a$, we can get more interesting results. For example, increasing $|\boldsymbol{u}_n|$ means that the biological plants grow at a higher speed.

In addition, if we collect these data from sensors, we can animate the results in more practical applications. For example, by visualizing the diffusion phenomenon of algae, appropriate method can be taken to prevent algae from growing in lake.

Our diffusion model can also be used in scientific analysis if we collect these data from sensors. It can be easily implemented by replacing the velocity $\boldsymbol{u}=\boldsymbol{u}_n+\boldsymbol{u}_a$ with the collected data. Specifically, $\boldsymbol{u}_n$ can be replaced by the obtained growing velocity of algae and $\boldsymbol{u}_a$ can be replaced by the measured wave velocity.

VII CONCLUSIONS

Biological pollution is frequently appeared in our world, and the wastes have caused serious harm to the lakes and rivers. These phenomena are also common in feature films, propaganda on environmental protection. We present a method to simulate the diffusion of biological pollutants on large-scale water surface in this paper. To make the computational resource available in the large-scale simulation, we focus on large patches of aquatic plants and model the interface between the patches and water surface. The dynamic evolution and diffusion of the interface is modeled by physically-based diffusion model. We construct a 2D level set equation to animate the evolution and diffusion of the water pollutants. To handle boundaries in large-scale virtual environment, we use an image-based voxelization method to provide correct boundary condition. Application results of algae diffusion in Taihu Lake show the effectiveness of our method.

Currently, our method only animates the diffusion of biological pollutants on static plane water surface. As for future work, we will extend it to animate the diffusion phenomenon on large-scale complex and dynamic water surface.

ACKNOWLEDGMENT

This work is supported by National Natural Science Foundation of P.R. China under Grant Nos. 61202225, 61272094, Project of Shandong Province Higher Educational Science and Technology Program under Grant No. J13LN13, Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant No. 20133704120009 and Shenzhen Basic Research Foundation under Grant No. JC201105190934A.

REFERENCES

[1] J. Shi, D. Zhu, Y. Zhang, and Z. Wang. Realistically rendering polluted water. Vis. Comput., 28(6-8):647–656, June 2012.

[2] V. Mihalef, Dimitris N. Metaxas, and Mark Sussman. Textured liquids based on the marker level set. Computer Graphics Forum, 26(3):457–466, 2007.

[3] T. Kim, J.Tessendorf, N.Thürey. Closest point turbulence for liquid surfaces. ACM Transactions on Graphics, 32(2):15:1-15:13,2013.

[4] D. Gerszewski, A. W. Bargteil,Physics-based Animation of Large-scale Splashing Liquids. ACM Transactions on Graphics, 32(6):185:1-185:6, 2013.

[5] S. Auer, R. Westermann: A Semi-Lagrangian Closest Point Method for Deforming Surfaces. Computer Graphics Forum 32(7): 207-214 , 2013.

[6] D. Enright, S. Marschner, and R. Fedkiw. Animation and rendering of complex water surfaces. ACM Transactions on Graphics, 21(3):736–744, July 2002.

[7] N. Chentanez and M. Müller. Real-time simulation of large bodies of water with small scale details. In Proceedings of the 2010 ACM SIGGRAPH/ Eurographics Symposium on Computer Animation, SCA ’10, pages 197–206, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[8] N. Chentanez and M. Müller. Real-time eulerian water simulation using a restricted tall cell grid. ACM Transactions on Graphics, 30(4):82:1–82:10, July 2011.

[9] N. Thürey, R. Keiser, M. Pauly, and U. Rüde. Detail-preserving fluid control. In Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation, SCA ’06, pages 7–12, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[10] G. Zhang, D. Zhu, X. Qiu, and Z. Wang. Skeleton-based control of fluid animation. Vis. Comput., 27(3):199–210, March 2011.

[11] G. Zhang, D. Lu, D.Zhu, L. Lv, H. Liu, and X.Meng. Rigid-motion-inspired liquid character animation. Computer Animation and Virtual Worlds, 24(3-4):205–213, 2013.

[12] Gui Y, Zhang G. An Improved Implementation of Preconditioned Conjugate Gradient Method on GPU. Journal of Software, 2012, 7(12): 2695-2702.

[13] Liu Z, Peng M J, Zhao Q. Study on Simulation Method of the Smoke-dust in Nuclear Decommissioning Virtual Simulation[J]. Journal of Computers, 2011, 6(2): 343-350.

[14] A. Fournier and William T. Reeves. A simple model of ocean waves. SIGGRAPH Computer Graphics, 20(4):75–84, August 1986.

[15] D. Jonathan, E. Bruneton. Real-time animation and rendering of ocean whitecaps. SIGGRAPH Asia 2012 Technical Briefs. p.15, ACM, 2012.

[16] Darles, E., Crespin, B., Ghazanfarpour, D.. Accelerating and enhancing rendering of realistic ocean scenes. In Proceedings of WSCG, pp. 287-294 , 2007.

[17] Q. Yu, F. Neyret, E. Bruneton, and N. Holzschuch. Scalable realtime animation of rivers. In Computer Graphics Forum, volume 28, pages 239–248. Wiley Online Library, 2009.

[18] Z. Dong, W. Chen, H. Bao, H. Zhang, and Q.Peng. Real-time voxelization for complex polygonal models. In Proceedings of the Computer Graphics and Applications, 12th PacificConference, PG ’04, pages 43–50,Washington, DC, USA, 2004. IEEE Computer Society.

[19] C. Rother, V. Kolmogorov, and A. Blake. ”grabcut”: interactive foreground extraction using iterated graph cuts. ACM Trans. Graph., 23(3):309–314, August 2004.

Guijuan Zhang is an assistant professor at School of Information Science and Engineering, Shandong Normal University, China. Her current research interests include fluid animation, computer animation, and visualization.