Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH ARTICLE

Target-Driven Cloud Evolution using Position-Based Fluids

Zili Zhang[1,2] | Yunfei Li[1] | Bailin Yang[3] | Frederick W.B. Li[4] | Xiaohui Liang*[1]

[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China
[2]Department of Computer Science and Engineering, Shijiazhuang University, Shijiazhuang, China
[3]Department of Computer and Electronic Engineering, Zhejiang Gongshang University, Zhejiang, China
[4]Department of Computer Science, University of Durham, Durham, United Kingdom

Correspondence
*Xiaohui Liang, State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China. Email: liang_xiaohui@buaa.edu.cn

Present Address
Beihang University, No.37, Xueyuan Road, Beijing, China

Summary

To effectively control particle-based cloud evolution without imposing strict position constraints, we propose a novel method integrating a control force field and a phase transition control into the Position-based Fluids (PBF) framework. To produce realistic cloud simulation, we incorporate both fluid dynamics and thermodynamics to govern cloud particle movement. The fluid dynamics is simulated through our novel driving and damping force terms. As these terms are only formulated based on cloud particle density and position, they simplify the inputs and make our method free from artificial positional constraints. The thermodynamics is implemented by our phase transition control, which can effectively simulate cloud evolution between discrepant initial and target shapes, producing plausible results. Uniquely, our method can also support target shape change during cloud simulation. Experiment results have demonstrated our method surpasses existing methods.

KEYWORDS:
cloud evolution, fluid control, target-driven, position-based fluids

1 | INTRODUCTION

Controlling fluid animation, such as smoke, clouds, and water, has been widely studied in computer graphics and related areas. The non-linear nature of fluid simulation makes it challenging to drive fluid matching with a target shape while maintaining natural fluid motions using fluid dynamics. Without a high-level control system, an artist may need to manually manipulate a lot of physical parameters to make a fluid animation following a desired shape, which is tedious and time-consuming. Some previous works have been proposed to control water and smoke motions in order to form user-specified shapes. They may be classified into grid-based methods [1–4] and particle-based approaches [5–7].

Cloud is an important element in synthetic outdoor scenes, training simulators, movie visual effects, etc. In this paper, we focus on controlling cloud evolution from an initial shape to a target one. It is difficult to simply extend previous fluid control methods to solve our problems, since they only simulate fluid dynamics without incorporating thermodynamics which plays an important role in cloud evolution. To provide cloud simulation control, Dobashi et.al. [8] proposed a feedback based framework using the grid-based method to control cloud formation, closely matching with the top contour of a cloud specified by users. This was achieved by controlling the latent heat and water vapor supply according to the height difference between the simulated cloud and the target contour during simulation. However, such a control was not sufficient for controlling cloud evolution accompanied by a series of evaporation and condensation. Moreover, it did not consider the constraints of given initial shapes. Combining fluid dynamics and thermodynamics, we present a novel method using Position-Based Fluid (PBF)[9] for controlling animated clouds matching with desired shapes.

Existing particle-based methods [5–7, 10] proposed to select some particles from the source model as control particles and determine their corresponding particles in the target model. The transformations between these particles formed strict positional constraints guiding other fluid particles to fill the target shape. Obviously, such artificial positioning of control particles contradicted natural fluid motion characteristics. This approach is also not favorable for user interaction. For instance, if animators want to change the target shape during a simulation, recalculation of the target positions of control particles is then required, which will take time and suppress user interaction. Instead of using artificial positional constraints to control fluid motions, we simulate fluid motions by determining a **control force** for each particle based on its position, velocity and density at each frame as well as combining the signed distance field generated from the target shape in a pre-computation process. This both frees our method from using positional constraints and simplifies the inputs, making our method simple and suitable for interactive graphics applications. We have also developed a **phase transition control** method to determine the transition between water vapor in the environment and liquid water in cloud particles, which is crucial for reducing the volume discrepancy between the simulated cloud and the target.

Our method samples particles from the source model and generates a signed distance field from the target shape as inputs. During the simulation, we use the negative gradient direction of the signed distance field and a color field to determine the directions of control forces. The control force for each particle is determined by a composition of multiple feedbacks, taking into account of particle position and density to preserve the fluid nature. We assume that the simulated cloud and the target have the same rest density. Inspired by the density constraints of PBF, we control the phase transition between water vapor and liquid water according to cloud particle position and density. We have evaluated our method on various evolution scenes, namely between two cloud models reconstructed from cloud images, from cloud object to animal shape, from multi-model to a single model and so on. These results demonstrate that our algorithm can robustly control cloud evolution between different objectives while preserving natural fluid motion. Our main contributions include:

- An effective PBF-based cloud animation control framework supporting cloud evolution between two models with different shapes and volumes.
- A novel particle-based fluid control approach, which determines the control force for each particle by using its position, density and velocity as control parameters and combines a signed distance field generated from the target shape. This avoids the use of positional constraints, making our method simple and suitable for interactive graphics applications.
- An adaptive phase transition control method, which smoothly adjusts the amount of water vapor and liquid water during simulation to preserve the target feature and keep constant rest density. This also makes our control method being able to effectively integrate into the PBF framework.

In the following, Section 2 presents existing work in fluid simulation, cloud simulation and fluid control. Section 3 describes the physical framework of our method. Section 4 elaborates the proposed control method. Section 5 then describes the implementation. Finally, Section 6 and 7 showcases several visual results and concludes the paper with future work, respectively.

2 | RELATED WORK

This section reviews previous works on fluid simulation, cloud simulation and fluid control.

Fluid simulation is a popular research in computer graphics. Early work have introduced stable and efficient grid-based solvers[11–14]. Bridson[15] has presented an excellent overview of grid-based methods. In contrast, our method relies on PBF[9], which belongs to particle-based methods[16–18]. To maintain incompressibility, some methods[19–24] proposed to construct a linear complementarity problem using linearized constraint functions. PBF solves the problem by formulating and solving a set of positional constraints. To improve the computational efficiency of particle-based methods, some adaptive particle frameworks have been proposed[25, 26]. For instance, Adams et al.[26] introduced a sampling condition based on geometric local feature size. Horvath et al.[27] used the adaptive method to simulate the two-scale fluid while preserving mass.

There are two major approaches for cloud simulation, namely procedural approach and physically based approach. Lots of methods have been proposed to model clouds based on procedural techniques[28–30]. Procedural approach may generate cloud with a desired shape, yet being difficult to create realistic cloud animations since a lot of parameters are required to adjust through trial and error. In contrast, physically based approaches can eliminate the need of adjusting parameters manually. Miyazaki et al.[31] used a grid-based method to formulate the cloud generation process. Harris et al.[32] proposed a similar method by using slightly different definitions for the phase transition. Recently, Barbosa et al.[33] developed a particle-based method for cloud simulation. These methods govern a cloud simulation by solving the Navier-Stokes equations. Hence, it is impossible to manually adjust many physical parameters for controlling cloud evolution to match with given targets. To solve the problem, Dobashi et.al. [8] proposed a feedback control based method to make clouds form desired shapes. However, it is difficult to extend this method to solve the problem of controlling cloud evolution, since the technique only addresses the cloud formation process without taking the constraints of a given initial shape into consideration.

In terms of fluid control, Treuille et al.[34] proposed a keyframe control technique to drive a smoke simulation matching with a desired shape, yet being computationally expensive for solving a non-linear optimization problem. Mcnamara et al.[35] used the adjoint method to solve the gradient-based nonlinear optimization for finding optimal control forces. It can deal with both smoke and water control. Pan and Manocha[3] proposed a space-time optimization solver, computing a dense sequence of control force fields to drive smoke matching with several keyframes. Ma et al.[4] presented a simulator to control a 2D coupled system with fluid bodies and rigid objects using deep reinforcement learning. To avoid expensive optimization over the entire fluid sequence, Hong and Kim[1] used a geometric potential field creating from a target shape to control smoke forming a desired shape. Shi and Yu[36] controlled fluid to follow rapidly changing target objects by applying a feedback control force field and the gradient field of a potential function. Fattal and Lischinski[37] designed a new driving force using smoke density to carry smoke towards a particular target. To prevent smoke from diffusing too much, a gathering term was added in the advection equation. Raveendran et al.[38] controlled fluid motion with the guidance of a dense sequence of control meshes generated by a volume preserving morph. These methods can effectively control fluid deforming between different target shapes.

However, the above methods are grid-based simulation, being computationally expensive and limited with fixed simulation area. The amount of work in particle-based fluid control methods are much less than that of the grid-based methods. Since Thürey proposed the pioneering work[10], several methods have been proposed[5?,6]. Thürey et al.[10] proposed a detail-preserving fluid control method. It first generated control particles, following by determining attraction forces and velocity forces to pull fluid towards the control particles. Control forces were only applied to coarse-scale components of the flow to preserve small-scale details. Zhang et al.[6] used control particles automatically sampled from a target mesh to attract fluid particles to a desired shape with fast movement or large deformation. These generated control particles are deformed with skeleton motion data specified by an animator. To solve clustering and clumping at the center of control particles, Zhang et al.[6] proposed a density constraint method. Springs between fluid particles and control particles were added to solve the problem that a target shape moves or deforms rapidly. Madill et al.[5] proposed a particle-based target driving control method for smoke simulation, where the number of source control particles and that of target control particles must be the same. Feng and Liu[7] improved the matching process between control particles to preserve fluid details.

All these particle-based methods commonly required the use of control particles and corresponding target particles, imposing strong positional constraints on particles. Also, as they mainly focused on water and smoke simulation, their implementations did not include certain thermodynamics processes, such as the phase transition between water vapor and liquid water, this significantly limits their capability of simulating cloud evolution properly.

3 | THE PHYSICAL FRAMEWORK

Our physical framework takes an initial cloud shape and a target cloud shape as inputs, simulating cloud evolution through cloud motion and cloud microphysics equations. The cloud motion process is evaluated through the Navier-Stokes equations, which models the conservation of momentum and mass continuity. The cloud microphysics evaluates the interaction between the cloud and the environment during the motion process.

3.1 | Cloud Motions

Realistic cloud animations are typically governed by numerically approximating the Navier-Stokes or the Euler equations. Assuming the atmosphere to be incompressible and inviscid, the motion of cloud can then be expressed as follows:

$$\frac{d\mathbf{u}}{dt} = -\frac{1}{\rho}\nabla p + \mathbf{f} \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

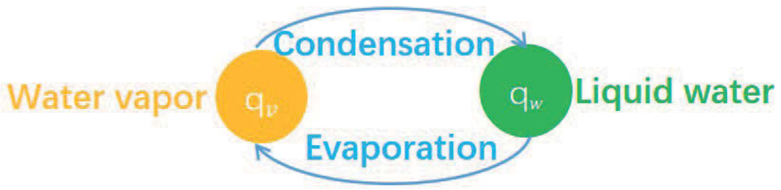


FIGURE 1 Phase transition between water vapor and liquid water.

where $\mathbf{u}$ denotes the fluid velocity vector. ρ is the density. p is the pressure. $\mathbf{f}$ accounts for the external forces affecting fluid flow, such as buoyance and wind force. Equation 1 is obtained by the conservation of momentum, and Equation 2 is the continuity equation.

The external force term in Equation 1 provides an important means of control over the fluid simulation. In this paper, we utilize the force term in order to drive the simulated cloud matching with a target shape from a given initial state. According to the method[36], the fluid control method should meet four criteria, namely control capability, ease of use, fluid-like motion and stability. Moreover, the force should be smooth among neighboring particles to maintain the motion consistency of fluid particles. Hence, we introduce a **driving force term**, which depends only on the instantaneous state of each particle, including its density and position. It is formulated as an explicit yet simple function $\mathbf{f_1} = F_{drv}(\rho, \mathbf{p})$ as described in Section 4.1, where $\mathbf{p}$ is particle position. Meanwhile, the gradual accumulation of momentum could result in unstable movement of fluid. We also introduce a **damping force term** for each particle depending on its velocity, denoting as $\mathbf{f_2} = F_{dam}(\mathbf{u})$, and will be discussed in Section 4.2. Putting the two external forces into Equation 1 gives a new momentum equation as follows:

$$\frac{d\mathbf{u}}{dt} = -\frac{1}{\rho}\nabla p + F_{drv}(\rho, \mathbf{p}) + F_{dam}(\mathbf{u}) \tag{3}$$

3.2 | Cloud Microphysics

The droplets in a natural cloud interact with their environment and with each other droplets in many ways, which affect the droplet sizes and condensation[39] during the entire motion process. According to the Kessler scheme[40], which describes the warm cloud including water vapor and cloud water, cloud formation and dissipation are estimated through condensation and evaporation (see Figure 1). Droplets grow by condensation if the environment has an excessive amount of vapor over the equilibrium value, as would be produced by chilling in pseudo adiabatic ascent. They evaporate if dry environmental air in sufficient amount is mixed with the cloudy air.

Condensation: In this process, water vapor (q_v) in the air is changed into liquid water (q_w). This links to cloud formation and evolution. The expression for condensation is implemented as:

$$C_{v\mapsto w} = min\{q_v, \frac{q_v - q_s}{1.0 + f(\theta)q_s}\} \tag{4}$$

where θ is the potential temperature of water vapor, $f(\theta)$ is a function of the potential temperature. The saturation mixing rate q_s is implemented as:

$$q_s = \frac{380.16}{p} \cdot exp(17.25\frac{\prod\theta - 273}{\prod\theta - 36}) \tag{5}$$

where p is the pressure, and Π is the Exner function which is a non-dimensional pressure quantity.

Evaporation: The evaporation process turns liquid water (q_w) in the cloud into water vapor (q_v). When the air becomes unsaturated, the process occurs to maintain a balance between liquid water in the atmosphere and water vapor. Hence, the rate of evaporation is regulated timely to keep the air at the saturation mixing ratio. Evaporation is calculated as follows:

$$C_{w\mapsto v} = min\{-C_{v\mapsto w} - q_w, \kappa\frac{q_s - q_v}{\rho q_s}\} \tag{6}$$

where κ is the control parameter determined experimentally.

Note that the phase transition between water vapor in the environment and liquid water in the clouds essentially controls the elements forming a cloud, determining the cloud volume. In order to solve the volume difference between the initial and the target cloud shapes, we therefore introduce a phase transition control method as in Section 4.3. This effectively maintains the physical nature and the constant rest density of a cloud evolution process.

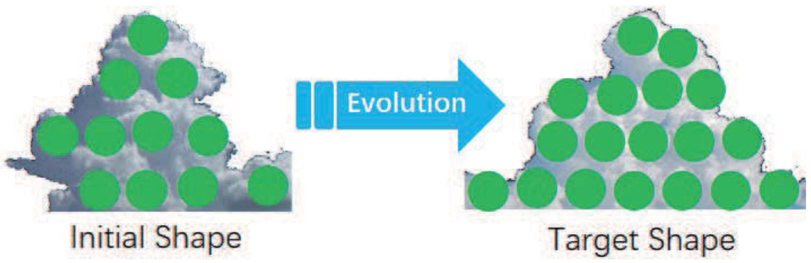


FIGURE 2 Shape and volume (indicated by different number of green circles) distinctions between initial and target shapes.

3.3 | Position Based Fluids

We use the Position Based Fluids which is a particle-based method to solve the motion equations of cloud. Similar to SPH, a physical value $\mathcal{A}(\mathbf{x})$ at location $\mathbf{x}$ is calculated by a weighted sum of contributions $\mathcal{A}(\mathbf{x}_j)$ from its neighboring particles j.

$$\mathcal{A}(\mathbf{x}) = \sum_j m_j \frac{\mathcal{A}(\mathbf{x}_j)}{\rho_j} W(\mathbf{x} - \mathbf{x}_j, h) \tag{7}$$

where m_j and ρ_j are the mass and density of particle j, respectively. The function $W(\mathbf{x}, h)$ is called the smoothing kernel, where h is the core radius. To solve the problem of fluid incompressibility, PBF proposes a density constraint for each particle associated with its neighbors. The density constraint on the particle i is defined using an equation of state:

$$C_i(\mathbf{p}_i, \mathbb{P}_i) = \frac{\rho_i}{\rho_0} - 1 \tag{8}$$

where $\mathbb{P}_i$ is the position list of the neighbors of particle i. ρ_0 denotes the rest density of fluid, and ρ_i corresponds to the fluid density defined by using Equation 7 as follows:

$$\rho_i = \sum_{\mathbf{p}_j \in \mathbb{P}_i} m_j W(\mathbf{p}_i - \mathbf{p}_j, h) \tag{9}$$

4 | CONTROL METHOD

As shown in Figure 2, in a cloud evolution scene, there are usually two kinds of distinctions between the initial and the target models, namely shape and volume, due to cloud-environment mixing[41]. We assume that the state of the environment is maintained during a simulation. Hence, the rest density of cloud should keep constant, which is also necessary for enforcing incompressibility using the PBF framework. Our method aims at controlling cloud evolution to reduce the difference between the target and the original models to zero, while ensuring that the simulated cloud keeps constant rest density during the simulation.

The core of our method is a position-based control solver, which consists of two major stages, namely **driving and damping force generation** and **phase transition control**. Figure 3 depicts an overview of our algorithm. We first sample an initial shape by voxelization to generate fluid particles and then compute a signed distance field from the target shape as input. For the control force generation stage, we design proper driving and damping forces to control particles matching with the target shape while preserving plausible fluid motions. We combine particle current position, velocity, the signed distance field and the density difference between its density and the rest density to generate the control force. For the phase transition control, we choose particle density and position as control parameters to determine whether the physical process occurs. It facilitates the preservation of target feature when the volume of the target is different from that of the initial model.

4.1 | Driving Force

In this section, we develop an appropriate formulation of the driving force term $F_{drv}(\rho, \mathbf{p})$. As the force only depends on the particle density and position, which are two essential parameters in the PBF framework, it is intuitive and simple for users to use. To control the driving force, we design two parameters, namely the direction and the amount of driving forces, to be applied.

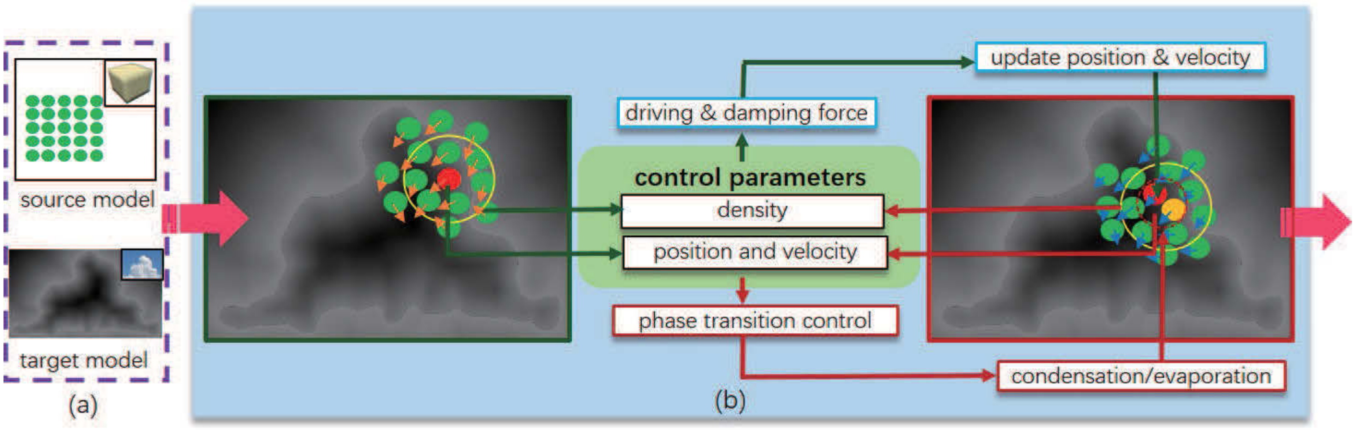


FIGURE 3 Overview of our algorithm. (a) shows the inputs to our system, namely the source and target shapes. The source shape is transformed into particles and a signed distance field is generated from the target shape. (b) shows our control mechanism. We choose particle density, position and velocity as control parameters, computing a driving and damping force for each particle to update its position and velocity. The phase transition controls the dynamical transitions between the water vapor in the air and the liquid water in cloud particles according to these parameters, simulating the processes of condensation and evaporation. The process will adjust the number of particles to keep constant rest density and preserve the details of the target shape.

Direction of Driving Force: We determine the initial direction using the gradient of the signed distance field generated from the target shape as in the method[1]. The gradient of the color field[17] is then used to update the initial direction.

For a three-dimensional domain $\mathfrak{B} \subset \mathbb{R}^3$, the unsigned distance is usually defined as the Euclidean distance from a given point $\mathbf{x} = (x, y, z)^T$ in space to the nearest point on the boundary $\partial\mathfrak{B}$:

$$d_t(\mathbf{x}) = \inf_{\mathbf{x}^* \in \partial\mathfrak{B}} \|\mathbf{x} - \mathbf{x}^*\| \tag{10}$$

Let the sign of the distance to be determined by whether a given point $\mathbf{x}$ is in $\mathfrak{B}$. The signed distance function $\phi(\mathbf{x})$[42] is then defined as follows:

$$\boldsymbol{\phi}(\mathbf{x}) = \begin{cases} -d_t(\mathbf{x}) & \mathbf{x} \in \mathfrak{B} \\ d_t(\mathbf{x}) & \text{otherwise} \end{cases} \tag{11}$$

If the cloud particle is outside the target shape, the generated force should drag the particle with a direction towards the target shape and vice versa. Therefore, we propose the initial direction of the driving force as:

$$\widehat{G(\mathbf{x})} = \begin{cases} \frac{\nabla\boldsymbol{\phi}(\mathbf{x})}{\|\nabla\boldsymbol{\phi}(\mathbf{x})\|} & \mathbf{x} \in \mathfrak{B} \\ -\frac{\nabla\boldsymbol{\phi}(\mathbf{x})}{\|\nabla\boldsymbol{\phi}(\mathbf{x})\|} & \text{otherwise} \end{cases} \tag{12}$$

where $\nabla\phi(\mathbf{x})$ is the gradient of $\phi(\mathbf{x})$. $\widehat{G(\mathbf{x})}$ is pre-computed according to the target shape, which is constant during a simulation.

Practically, motion conflict situation may occur, as illustrated by the red cloud particle in Figure 4. This particle is inside the target shape and the initial driving force moves the particle towards the target shape boundary. On the contrary, some of its neighbors are outside the target shape and their initial driving forces move them back to the target shape. We tackle this situation based on the color field idea, which assigns 1 to each target cloud particle positions and 0 to else were. We apply the negative gradient of smooth color field[17] to solve the problem as follows:

$$N(\mathbf{p}_i) = -\sum_j m_j \frac{1}{\rho_j} \nabla W(\mathbf{p}_i - \mathbf{p}_j, h) \tag{13}$$

According to the signed distance $\Phi(\mathbf{x})$ and the dot product between $N(\mathbf{x})$ and $\widehat{G(\mathbf{x})}$, we update the initial direction as follows:

$$G(\mathbf{x}) = \begin{cases} \widehat{G(\mathbf{x})} & N(\mathbf{x}) \cdot \widehat{G(\mathbf{x})} \geq 0 \ or \ \Phi(\mathbf{x}) \leq 0 \\ -\widehat{G(\mathbf{x})} & \text{otherwise} \end{cases} \tag{14}$$

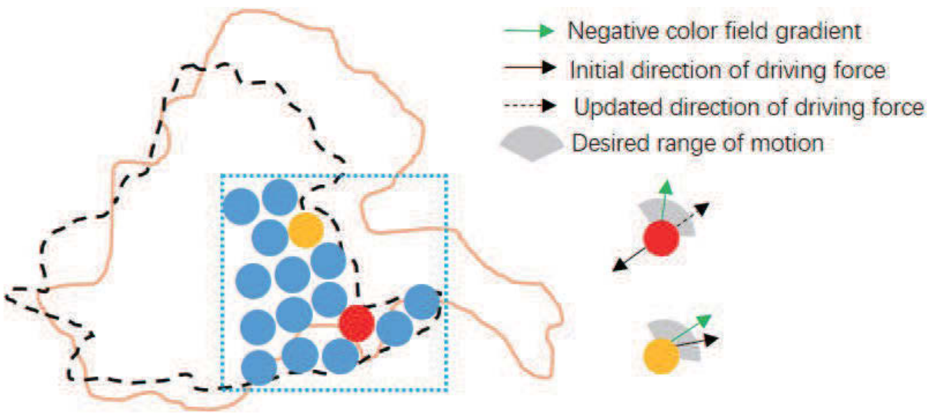

Negative color field gradient
Initial direction of driving force
Updated direction of driving force
Desired range of motion

FIGURE 4 Determining the direction of driving force. The black contour is the initial shape and the orange one denotes the target shape. The dotted box shows a local part of the simulated domain. The red and yellow circles are two particles at different positions.

If the angle between vector $N(\mathbf{x})$ and vector $G(\mathbf{x})$ is larger than $\pi/2$, it shows that the initial direction of control force collides with the desired movement of the fluid particle. We use the gradient of the color field to update the initial direction of the control force (ref. the red particle in Figure 4).

Amount of Driving Force: In the following, we will describe the way to determine the amount of driving force. Simply setting a constant will lead to obvious artifacts and result in gradually growing momentum. Hence, the simulated fluid may not reach a rest state with $\mathbf{u} = 0$, even the target has been reached. To make the controlled movement of the fluid stable and natural, we use particle density and position as the control parameters to determine the driving force.

The amount of driving force is the product of the signed distance and the density difference between the particle's density and the rest density. To avoid obtaining an excessive driving force, we use the sigmoid function to clamp the product. Hence, the amount of control force is calculated as follows:

$$f_i(\rho_i, \mathbf{p}_i) = \alpha \cdot sigmoid(|\phi(\mathbf{p}_i)(\rho_i - \rho_0)|) \tag{15}$$

where α is the scaling parameter. To achieve smooth evolution, we usually set $\alpha = 0.16$ in our experiments. ρ_i and ρ_0 are the density of fluid particle i and the rest density, respectively. The difference between ρ_i and ρ_0 controls the volume distinction between two shapes. If the particle density is higher than the rest density and the difference between them is also large, the driving force is then large, driving particles to meet incompressibility quickly. According to the definition of function $\phi(\mathbf{p_i})$, the farther the fluid particle drifts away from the target shape, the larger the driving force becomes.

When the target shape is reached, the density of each particle inside the target shape nears the rest density, i.e., $|\rho_i - \rho_0| = 0 + \delta$, where δ is a minimum. Hence, the driving forces acting on the particles will become zero. Meanwhile, the signed distance, $\phi(\mathbf{x})$, is close to zero at the boundary of the target shape, and that the particles at the boundary will not be driven further by the control forces. It shows that the simulated object will reach a rest state when it reaches the target shape. Finally, the driving force for particle i is calculated as follows:

$$\mathbf{F}^i_{drv}(\rho_i, \mathbf{p}_i) = f_i(\rho_i, \mathbf{p}_i) G(\mathbf{p}_i) \tag{16}$$

The proposed driving force is similar to but different from the control forces, including the boundary control force and the shape control force, presented by Yang et al.[43]. The gradients of the sign distance field is employed to estimate the boundary control force. The shape control force is based on the medial axis point clouds, which are extracted using the Laplacian criteria after computing the signed distance field of the target shape. Due to the lack of the density constraints in the control forces of[43], their method produces unevenly distributed cloud particles, with more cloud particles gathering around the medical axis and areas away from the medical axis becoming hollow because of lacking cloud particles, which is undesirable. Meanwhile, we use the signed distance field of the target shape to calculate the driving force controlling both the boundary and the shape, which is a much easier way of implementation for users than the approach in[43].

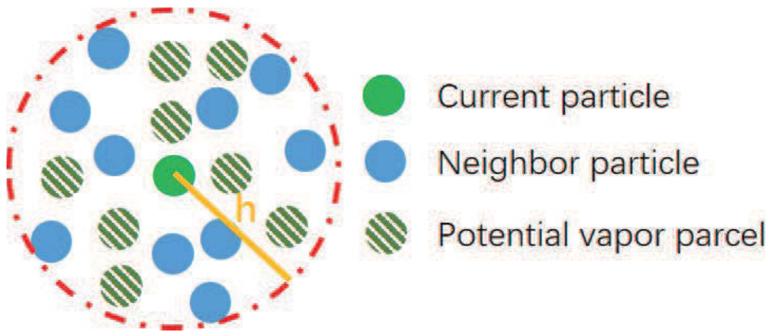


FIGURE 5 Example for phase transition from vapor to cloud.

4.2 | Damping Force

To avoid the accumulation of momentum, which may keep fluid away from reaching the rest state, we design a damping force using the velocity of particles. It is calculated as follows:

$$F^{i}_{dam}(\mathbf{u}_i) = -\beta \cdot sigmoid(||\mathbf{u}_i||^2)\frac{\mathbf{u}_i}{||\mathbf{u}_i||} \tag{17}$$

where β is the gain for proportional control. The larger the value of β is, the larger the damp force becomes. To drive particles filling the target more rapidly while restricting the fast-growing of momentum, we usually set $\beta = 0.76$ in our experiments. Note that the force follows the opposite direction of $\mathbf{u}_i$ to decrease the momentum of the simulated fluid.

4.3 | Phase Transition Control

To preserve the fluid nature of the constant density during evolution, we simulate the microphysics processes of clouds, which formulate the interaction between clouds and their environment, to solve the volume discrepancy between the initial and the target shapes.

If the volume of the target shape is larger than the initial one, we then assume that there are lots of vapor parcels in the environment. This means some water vapor in the air parcels should be condensed into cloud droplets to make the cloud grow. To make such growth more natural, only those vapor parcels which are either inside or at the boundary of the cloud are involved in the process. If the volume of the target shape is smaller than the the initial one, we assume that the air is unsaturated. Hence, some liquid water in cloud droplets is evaporated into air parcels, which reduces the volume difference. Since the two processes only change the state of the water in air parcels and cloud droplets, the total mass of the air and the cloud is remained the same during evolution.

According to Equations 4 and 6, both the rate of condensation $C_{v\mapsto w}$ and that of evaporation $C_{w\mapsto v}$ are nonlinear functions of the latent temperature θ and saturation mixing rate q_s. However, it is difficult to adjust these parameters manually, making cloud evolution match with a user-specified target shape. Therefore, we present two phase transition controllers, including condensation controller and evaporation controller, based on the shape feedback to automatically adjust the two phase transition rates.

Condensation Controller: In order to control the condensation, we need to determine the location where there are more saturated water vapor. As all cloud particles need to meet the incompressibility constraint at each frame, if the density of one cloud particle is smaller than the rest density, it will mean there are more water vapor around it. Hence, we can use the difference between the particle density and the rest density as the criterion to determine such a location. According to the density difference, we design the method to determine the potential locations.

Given that a density threshold ζ ranges from 0 to 1, if the density of particle i, denoted as ρ_i, is lower than the rest density $(1-\zeta)\rho_0$, the vapor parcels around the particle i are then potential particles forming part of the cloud. To make particles better preserve the constant rest density and reduce the fluctuation of the density, the parameter ζ should be assigned with a low value. We therefore set ζ to 0.1 in our experiments. The position $\mathbf{p}_i$ is regarded as a potential location around where particles may transit into vapor, and these potential locations constitute a set, denoted as $\mathbb{L}_c$. Meanwhile, new cloud particles should not conflict with other particles, as illustrated in Figure 5. Note that there are potentially many potential valid positions, from which we pick some random air parcels.

For those particles located at the boundary of the target shape, the phase transition rate should be smaller in order to stop the growth. When the ith fluid particle's density is much smaller than the rest density, we should make the cloud grow to match with the target shape. Meanwhile, the cloud density of new fluid particle turned from air parcel should be near to the particle's cloud density in order to get smoothed cloud density, which is similar to[33]. Finally, the condensation controller is defined as:

$$C_{v\mapsto w} = K_c M(-\phi(\mathbf{p}_i))M(\rho_0 - \rho_i)(\frac{w_i}{w_i + v_i})q_v \tag{18}$$

$$M(r) = \begin{cases} 1 & r > 0 \\ 0 & r \le 0 \end{cases} \tag{19}$$

where w_i and v_i are the liquid water density and the vapor density in cloud particle i, respectively. According to the method calculating cloud density[33], the different distributions of the two parameters influence the cloud density and hence the rendering results. K_c denotes the smooth control parameter and is set to 0.6 in our experiments. When the surface of the original cloud reaches the boundary of the target shape, $\phi(\mathbf{p}_i)$ of the particle i on the surface is zero. So the term $M(-\phi(\mathbf{p}_i))$ becomes zero, meaning that the phase transition will be stopped.

Evaporation Controller: Evaporation can be considered as an inverse process of condensation. We use the physical process to solve the gathering of particles which makes the particle density higher than the rest density. If the density of particle i is larger than the rest density ρ_0, and the difference between ρ_i and ρ_0 is larger than $\zeta\rho_0$, i.e., $\rho_i - \rho_0 > \zeta\rho_0$, its neighbor particles could be turned into vapor. Consequently, the neighborhood of the particle i is considered as a potential region and its location is set to be the position of the particle i, i.e., $\mathbf{p}_i$. Finally, all these potential locations form a set $\mathbb{L}_v$.

To rapidly make fluid particles meet the rest density, the larger the density difference is, the larger the condensation rate could be. So we define the evaporation controller for the phase transition from cloud to vapor as:

$$C_{w\mapsto v} = K_v \cdot max(0, sigmoid(\rho_i - \rho_0))w_i \tag{20}$$

where w_i is the cloud density of particle i, and K_v is scaling parameter, which is usually set to 0.8 in our experiments. We pick some random neighbors of particle i, e.g., $\widehat{p}_1, \widehat{p}, \cdots, \widehat{p}_l$, to apply the physical process with the same phase transition rate $\overline{c}_i$. We then choose two particles, $\widehat{p}_m$ and $\widehat{p}_n$, whose distance is the smallest than any pairs of particles in the neighborhood, to merge into a fluid particle and an air parcel iteratively. The new fluid particle is placed in the middle between the neighboring particles. Meanwhile, the mass of the new fluid particle is set to the average of the original fluid particles, and its cloud density is set to the sum of cloud density of the two particles. Similar to the phase transition from vapor to cloud, we use the same density constraint criteria to control the amount of phase transition.

4.4 | Density Constraints

As the phase transition will influence the number of neighbors of some particles, it could result in density fluctuations in some local regions. The problem can be solved by controlling the amount of phase transition. Hence, we propose a criterion according to the density constraint of PBF to control the two processes.

When a particle p_i does not satisfy the density constraint, $C_i(\mathbf{p}_i, \mathbb{P}_i) \neq 0$, the phase transition will be performed to adjust the number of its neighbors. This aims to update the neighborhood $\widetilde{\mathbb{P}}_i$, generating the density correction $\Delta\rho_i$ to turn $C_i(\mathbf{p}_i, \widetilde{\mathbb{P}}_i)$ to zero. Hence, the density correction $\Delta\rho_i$ should satisfy:

$$\Delta\rho_i = |\rho_i - \rho_0| \tag{21}$$

5 | IMPLEMENTATION

To properly parallelize the process of evaporation control, we construct individual small sets, denoted as $\widetilde{\mathbb{L}}_v^1, \widetilde{\mathbb{L}}_v^2, \cdots, \widetilde{\mathbb{L}}_v^m$ from the set $\mathbb{L}_v$. For each set $\widetilde{\mathbb{L}}_v^n$, the distance of any two regions in the set should be larger than $2 \times h$, as shown in Figure 6. The initial set shown in the left sub-figure is divided into three sets, colored in purple, yellow and blue, respectively. Therefore, the merging process can be implemented on each set $\widetilde{\mathbb{L}}_v^n$ in parallel.

To control the phase transition between water vapor in the air and liquid water in the fluid particles, we construct two neighbor lists, namely water vapor list N_v and liquid water list N_c, for each fluid particle. Integrating control force and phase transition control into the PBF framework, the simulation loop is outlined in Algorithm 1. Our control solver is from line 2 to line 17. As

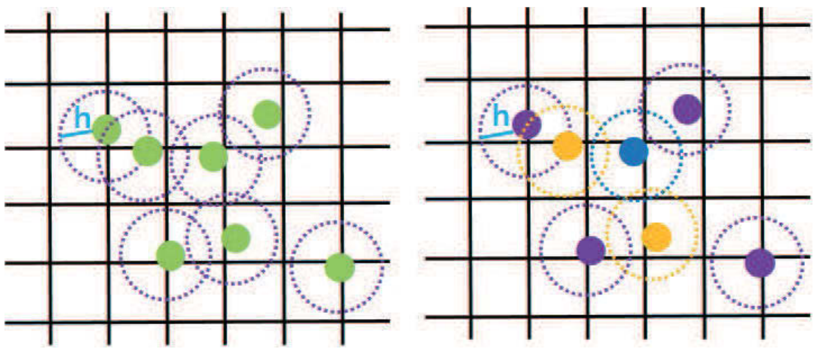


FIGURE 6 Parallelizing the process of evaporation control. Left sub-figure shows regions tagged to evaporate. Right sub-figure displays new sets constructed from the initial set on which the evaporation control can be parallelized. The three sets are colored in purple, yellow and blue, respectively.

density controlling could not make all particles fulfill incompressibility, the last step of our control step performs the position updating process according to the density constraints of PBF.

Algorithm 1 Control flow.

```
1:  for all particles $i$ do compute $N_c(\mathbf{p}_i)$ and $\rho_i$
2:  for all patiles $i$ do
3:      compute force $F^i_{con} = F^i_{drv} + F^i_{dam}$ using Eqns.16 and 17
4:      update velocity $v_i \leftarrow v_i + \Delta t F^i_{con}$ and $\mathbf{p}_i$
5:  end for
6:  for all particles $i$ do compute $N_c(\mathbf{p}_i)$ and $\rho_i$
7:  generate the two sets $\mathbb{L}_v$ and $\mathbb{L}_c$
8:  for all potential locations $\mathbf{p}_j \in \mathbb{L}_c$ do
9:      for all air parcel $k \in N_v(\mathbf{p}_j)$ do
10:         apply the process of condensation using Eqn. 18
11:     end for
12: end for
13: for all potential locations $\mathbf{p}_j \in \widetilde{\mathbb{L}}^i_v$ do
14:     for all fluid particle $k \in N_c(\mathbf{p}_j)$ do
15:         apply the process of evaporation using Eqn. 20
16:     end for
17: end for
18: for all particles $i$ do
19:     update $\mathbf{p}_i$ by using density constraints of Eqn.8
20: end for
```

6 | RESULTS

In the following, we will describe the examples that we have run and compare with existing works. We implement our method using the C++ programming language. All results are obtained on an Intel® Xeorn® CPU with 64GB memory and a NVIDIA® Quadro M4000 GPU.

We now demonstrate how our method can efficiently control cloud evolution between two non-volume-preserving models through a variety of scenes. In all of the examples shown in this section, we choose some mesh models as initial and target

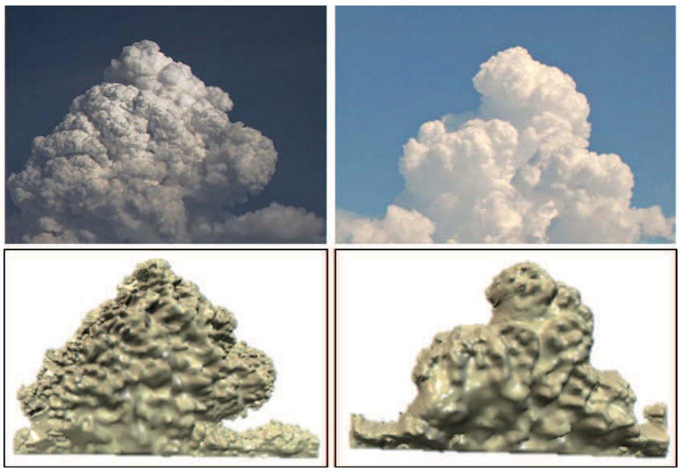


FIGURE 7 Cloud models (bottom row) reconstructed from cumuliform cloud images (top row).

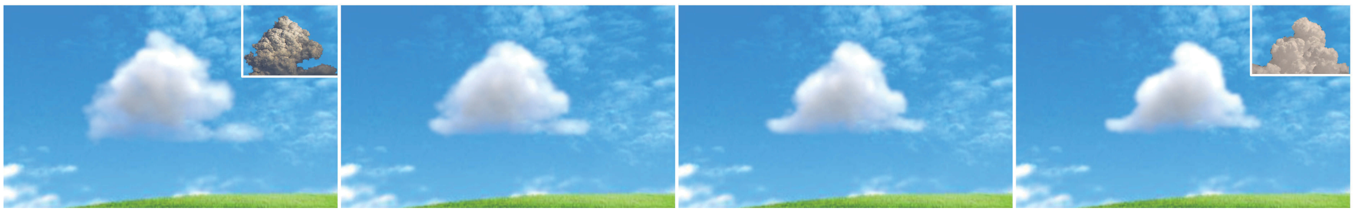


FIGURE 8 Evolution between two cumulus clouds models generated from given images.

models. For our system, we first sample particles uniformly from the initial shape as inputs and set all particles with the same radius and their mass to be $0.0512g$. We set the density of each air parcel to the rest density of $1kg/m^3$. The initial values of the liquid water density w and the vapor density v in each cloud particle are set to $0.8kg/m^3$ and $0.2kg/m^3$, respectively. As the signed distance field generated from the target shape is used to determine the driving force, its accuracy will affect the control results. Consequently, we generate the signed distance field (SDF) from the given target shape using the method[42], which can efficiently construct a grid-based SDF using hierarchical $hp-$refinement based on piecewise polynomial fitting. Our system is implemented in CUDA®, and we use a kd-tree for finding neighboring particles. The final rendering is done using Mitsuba[44]. The fluid density field is generated by splatting fluid particles onto a regular grid with the Poly6 kernel function.

Image can easily capture real clouds and is highly accessible. Hence, as in the first example (ref. Figure 8), we have simulated a cloud evolution between two given cloud images. First, we reconstructed the cloud models from user-specified cumuliform cloud images using the method[45]. The initial model and the target model are named as **cloud1** (left) and **cloud2** (right) as shown in Figure 7, respectively. Number of particles filled into the two models are shown in the first row of table 1. Note that cloud1 has a smaller volume than cloud2. Hence, condensation control will be performed during the simulation. The cloud gradually evolved from one shape (cloud1) to another (cloud2) with plausible visual effect, as shown in video demonstration.

In Figure 9, we demonstrate our method by controlling the dragon model to match with the bunny model. Despite having a very different and complex target model, our method manages to well approximate the target shape. Meanwhile, as shown in the second row of table 1, the dragon model contains more particles than the bunny model. Therefore, evaporation control is effective to adaptively reduce the number of particles during the evolution in this example.

Figure 10 shows that our method can support interactive user intervention. During the simulation, a user can change the target shape at any frame and the simulation can continue to be executed without any additional operating procedure. At the beginning, we control cloud3 (1st picture) to match with cloud1. At frame 30 (2nd picture), we choose the bunny model as the new target shape. Consequently, the evolving cloud at this frame becomes a new initial shape. Results demonstrate that our method can adapt target shape changes without any extra input or interruption. It also shows the scalability of our control method.

Figure 11 shows an example with a large source and target volume difference, with the source shape being a box (see the small inset of the 1st picture) and the target being cloud2. As shown in table 1, the box only contains less than one-half of the

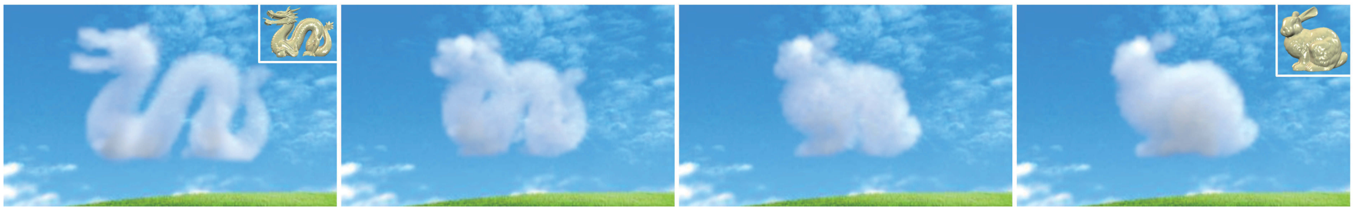


FIGURE 9 A dragon model deforms to a bunny model.

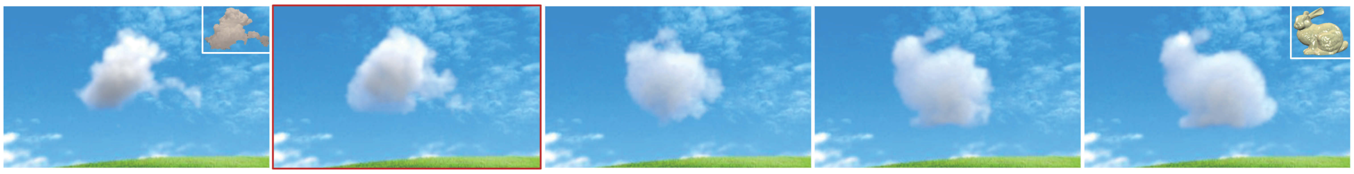


FIGURE 10 User intervention example. We first control cloud3 (1st picture) to match with cloud1. At frame 30 (2nd picture), the evolving cloud becomes a new initial shape, as the target shape is changed to the bunny model.

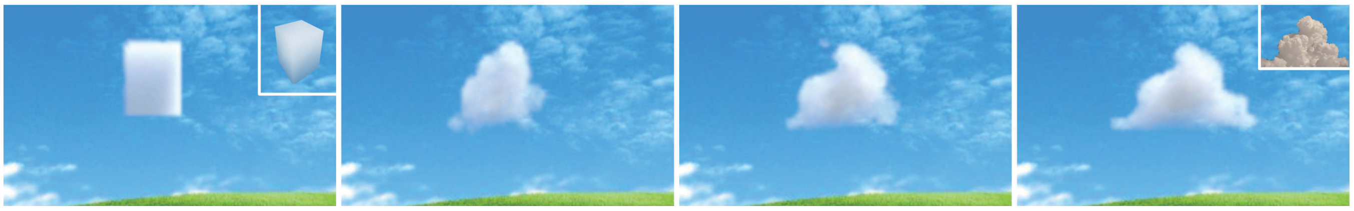


FIGURE 11 Examples with large source and target volume difference. The initial shape shown at the top right corner of the left image contains nearly half of the fluid particles for the target model.

number of fluid particles of the target model. As more air parcels around the cloud participate the condensation process during the simulation, we observe that the box can better transform into the given target shape and preserve the constant rest density.

Since each cloud particle is featured with liquid water density w and vapor density v, these density parameters consequently affect the control results. Also, according to[33], the cloud rendering result will be affected by different w and v value settings. We have conducted an experiment by lowering the value of w and increasing the value of v of the cloud simulation as shown in Figure 8, with $w = 0.4$ and $v = 0.6$. As shown in Figure 12, we then obtain a cloud with lower density.

To demonstrate the effectiveness of our method, we compare our method with one of the existing control particle based method[5]. We choose the two scenes as shown in Figures 8 and 9 to be the test examples, and show their results in Figure 13 and Figure 14, respectively. As control particles play an important role for methods based on control particles, we respectively choose $10,000$ and $13,000$ control particles, as well as their target particles by using the method[5] for the two scenes. Figure 13 shows a comparison of results generated by different methods. Particularly, comparing with method[5] (left picture), our method

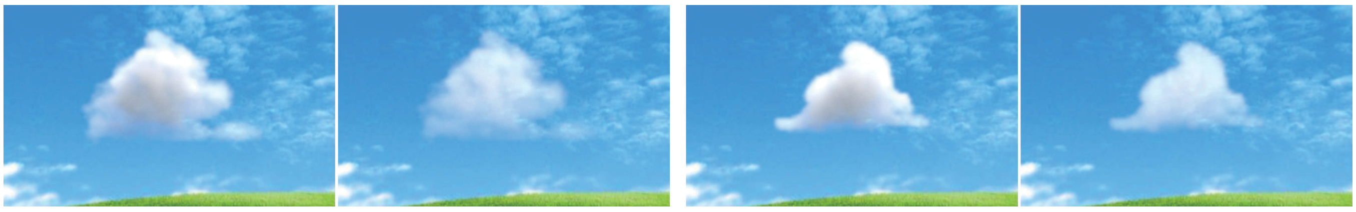


FIGURE 12 Comparison of results obtained using the different values for the parameters w and v. In each pair, the left one is obtained by using a larger w than the right one.

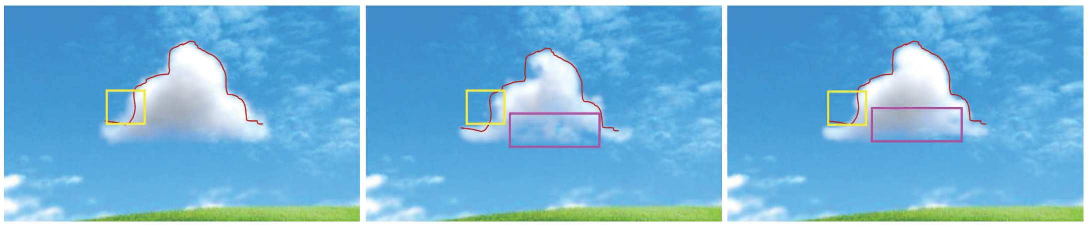


FIGURE 13 Comparison of results obtained by Madill's method[5] (left), Yang's method[43](middle) and our method (right) for the scene1. The red line is the top contour of the target shape.

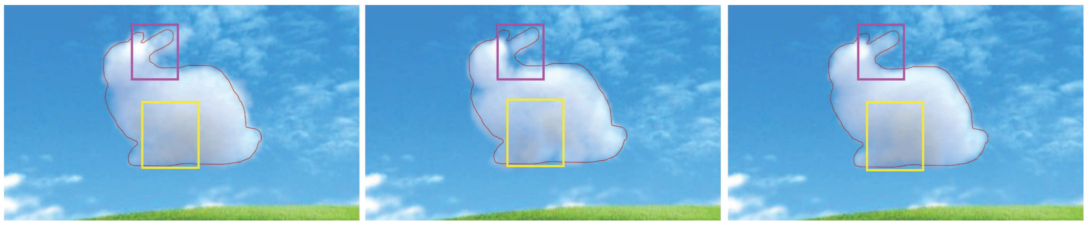


FIGURE 14 Comparison of results obtained by Madill's method[5] (left), Yang's method[43](middle) and our method (right) for the scene2. The red line is the contour of the target shape in each image. The ears and legs are highlighted with the yellow and purple squares in each images.

(right picture) produces a better cloud simulation, as the resulting cloud can better fit the target shape boundary (highlighted by the yellow box). As shown in Figure 14, our method can also better preserve details, such as the ears (highlighted by the red box) and legs (highlighted by the yellow box) of the bunny. The improvement made by our method comes from the control force as well as the shape and the density feedbacks. Specifically, as the control particle based method[5] overstrains the control particles into the target positions, the control particles are then moved roughly along straight lines. We have demonstrated such an unnatural motion in the supplementary video.

To compare our method with the unified smoke control method[43], which is based on the Eulerian approach, we have implemented the method using the PBF. We then simulate the two scenes as shown in Figures 8 and 9 using[43]. As the control forces in[43] only contain two terms, namely the medial axis point cloud constraints and the target boundary constraints, it results that the particle density is higher among these medial axis points than other particles. Moreover, due to the volume difference between the initial shape and the target model, the resulting shapes from the target-driven method lack details and make the structures appear some hollows as shown in the middle pictures of both Figure 13 and Figure 14.

Our method involves the phase transition control to meet the assumption of keeping the rest density constant during evolution. Figure 15 shows the change of average density of all particles at each frame for the scene as shown in Figure 8. In the example, we set the rest density to be $1000g/m^3$. Figure 15 shows that the proposed method can make the average density of particles near the rest density. Our method performs better comparing to the control particle based method[5].

We choose particle density and position as control parameters to determine the driving force for each particle. Hence, the amount of momentum generated by the driving force will decrease as the initial shape approaches the target shape. Figure 16 shows that the change of the average momentum for the example as shown in Figure 9. To test the system stability, we continue running the simulation process even when the simulated object have matched with the target. We use the strength of the average momentum as the test's recorded values. The orange curve shows the average momentum change for the method when using only geometric potential to determine forces, while our own result is depicted with the blue curve. Note that with the method using only geometric potential as control parameters, the average momentum could still fluctuate when the initial shape has already matched with the target. It demonstrates that our method can make the simulated fluid better satisfy the rest state requirement.

Table 1 summarizes the timing of our control algorithm in different scenarios. The shape and volume differences between two models are the two main factors affecting the performance of our control solver. We show the particle numbers of the initial shapes and the target shapes in column 1 and 2, respectively. Column 3 shows the particle number of the final generated target models.

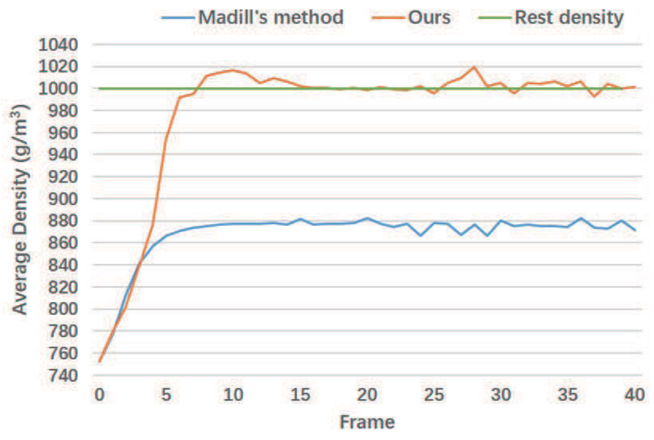


FIGURE 15 Comparisons of fluid rest density. It shows the change of average density for scene1.

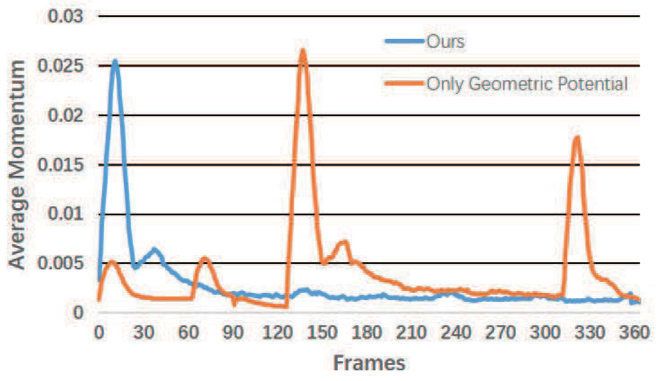


FIGURE 16 Momentum comparison between our method and the method of using geometric potential as control parameters to determine driving forces.

TABLE 1 Performance results for all the presented cases. Note that the rendering time is not included in the total time.

Scene	Initial Model Particles	Target Model Particles	Generated Target Model Particles	Time[msec/frame]
cloud1-cloud2	91,800	102,000	101,650	71.48
dragon-bunny	106,200	104,600	103,920	72.34
one box-cloud2	50,500	102,000	101,930	71.06
two boxes-cloud2	101,000	102,000	102,100	71.51
cloud3-cloud1-bunny	83,600	91,800/104,600	104,430	72.18

Figure 17 is an example demonstrating that we can support two initial shapes transforming into one single target. We choose two boxes as initial shapes and the cloud2 shown in the right of Figure 8 as the target shape. The number of particles for the two models are shown in the third row of table 1. The driving force is very effective at directing the two shapes to accurately form the cloud2.

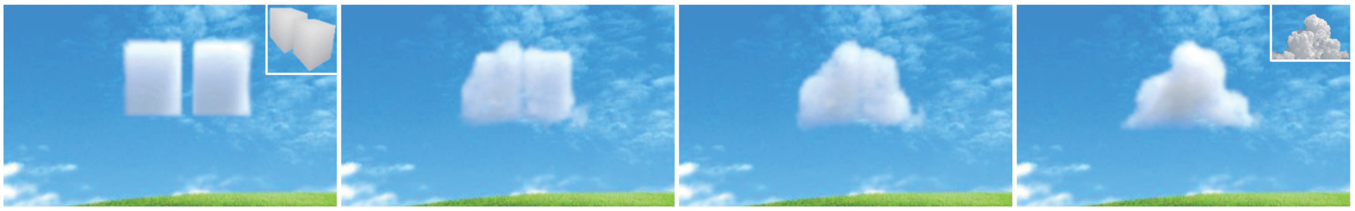


FIGURE 17 The cloud2 model is formed by two squares.

7 | CONCLUSION AND FUTURE WORK

In this paper, we propose a simple and effective target-driven cloud evolution method based on the PBF framework, enabling the simulated cloud to match with a given target and keep constant rest density. Our method only requires a user to provide two mesh models as the initial and the target shapes, respectively. Different from the control particles based method, our method assigns a control force to each particle by taking into account its position, density and velocity as well as a signed distance field generated from the target. Uniquely, this new approach allows our method to support interactive user intervention.

The control force will reduce when the simulated cloud approaches to the target, fulfilling the rest requirement. To solve the difference in volumes and integrate the proposed control method with the PBF framework, we present the phase transition control to adjust the cloud particle number by applying phase transition between vapor and cloud under the density constraints. The control force and the phase transition controller enable our method to preserve the details and keep constant rest density. Integrating PBF framework makes the method fast and stable. It can be easily integrated into existing fluid simulation frameworks and will only add a small ratio of computation time to the original simulation.

The major limitation of this work is that the signed distance field generating from the target shape must be pre-computed. Particularly when supporting interactive user intervention, such a pre-computation allows our method to revise the driving force interactively. In future work, it might be interesting to develop a method to drive the simulated fluid to match with a moving target by updating the signed distance field along with the motion of the target. Combining with the rapidly changing target matching method[6] to improve our method for supporting path control and shape matching may be a promising direction.

ACKNOWLEDGMENT

This paper is supported by the National Key R&D Program of China (No. 2017YFB1002702) and the National Natural Science Foundation of China (No. 61572058). We would like to thank anonymous reviewers for helpful comments.

References

1. Jeong-mo Hong ChK. Controlling fluid animation with geometric potential. Controlling fluid animation with geometric potential. 2004;**15**:147–157.

2. Cui Q, Wu Z, Xing C, Zhou Z, and Wu W. Target Temperature Driven Dynamic Flame Animation. In: Eurographics (Short Papers); 2015. p. 45–48.

3. Pan Z, and Manocha D. Efficient optimal control of smoke using spacetime multigrid. arXiv preprint arXiv:160801102. 2016;.

4. Ma Pingchuan PZRB Tian Yunsheng, and Dinesh M. Fluid directed rigid body control using deep reinforcement learning. ACM Transactions on Graphics. 2018;**37**(4).

5. Madill J, and Mould D. Target particle control of smoke simulation. In: Graphics Interface; 2013. p. 125–132.

6. Zhang S, Yang X, Wu Z, and Liu H. Position-based fluid control. In: Symposium on Interactive 3d Graphics & Games; 2015. p. 61–68.

7. Feng G, and Liu S. Detail-preserving SPH fluid control with deformation constraints. Computer Animation and Virtual Worlds. 2018;**29**(1):e1781.

8. Dobashi Y, Kusumoto K, Nishita T, and Yamamoto T. Feedback control of cumuliform cloud formation based on computational fluid dynamics. In: ACM Transactions on Graphics (TOG). vol. 27. ACM; 2008. p. 94.

9. Macklin M. Position based fluids. Acm Transactions on Graphics. 2013;**32**(4):1–12.

10. Thürey N, Keiser R, Pauly M, and Rüde U. Detail-preserving fluid control. Graphical Models. 2009;**71**(6):221–228.

11. Foster, Nick, Metaxas, and Dimitri. Realistic animation of liquids. Graphical Models & Image Processing. 1996;**58**:471–483.

12. Foster N, and Metaxas D. Controlling fluid animation. In: Proceedings Computer Graphics International. IEEE; 1997. p. 178–188.

13. Stam, and Jos. Stable fluids. Acm Transactions on Graphics. 1999;**1999**:121–128.

14. Fedkiw R, Stam J, and Jensen HW. Visual simulation of smoke. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques. ACM; 2001. p. 15–22.

15. Bridson R. Fluid simulation for computer graphics. AK Peters/CRC Press; 2015.

16. Desbrun M, and Gascuel MP. Smoothed particles: A new paradigm for animating highly deformable bodies. In: Computer Animation and Simulation. Springer; 1996. p. 61–76.

17. Müller M, Charypar D, and Gross M. Particle-based fluid simulation for interactive applications. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation. Eurographics Association; 2003. p. 154–159.

18. Ihmsen M, Orthmann J, Solenthaler B, Kolb A, and Teschner M. SPH fluids in computer graphics. The Eurographics Association; 2014.

19. Becker M, and Teschner M. Weakly compressible SPH for free surface flows. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation. Eurographics Association; 2007. p. 209–217.

20. Solenthaler B, and Pajarola R. Predictive-corrective incompressible SPH. In: ACM transactions on graphics (TOG). vol. 28. ACM; 2009. p. 40.

21. Bodin K, Lacoursiere C, and Servin M. Constraint fluids. IEEE Transactions on Visualization and Computer Graphics. 2012;**18**(3):516–526.

22. Ihmsen M, Cornelis J, Solenthaler B, Horvath C, and Teschner M. Implicit incompressible SPH. IEEE Transactions on Visualization and Computer Graphics. 2014;**20**(3):426–435.

23. Cornelis J, Ihmsen M, Peer A, and Teschner M. IISPH-FLIP for incompressible fluids. In: Computer Graphics Forum. vol. 33. Wiley Online Library; 2014. p. 255–262.

24. Peer A, Gissler C, Band S, and Teschner M. An Implicit SPH Formulation for Incompressible Linearly Elastic Solids. Comput Graph Forum. 2018;**37**:135–148.

25. Desbrun M, and Cani MP. Space-time adaptive simulation of highly deformable substances. INRIA; 1999.

26. Adams B, Pauly M, Keiser R, and Guibas LJ. Adaptively sampled particle fluids. In: ACM Transactions on Graphics (TOG). vol. 26. Acm; 2007. p. 48.

27. Horvath CJ, and Solenthaler B. Mass preserving multi-scale SPH. Pixar Technical Memo 13–04, Pixar Animation Studios. 2013;.

28. Wither J, Bouthors A, and Cani MP. Rapid sketch modeling of clouds. In: Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBIM). Eurographics Association; 2008. p. 113–118.

29. Man P. Generating and real-time rendering of clouds. In: Central European seminar on computer graphics. Citeseer; 2006. p. 1–9.

30. Gardner GY. Visual simulation of clouds. Acm Siggraph Computer Graphics. 1985;**19**(3):297–304.

31. Miyazaki R, Dobashi Y, and Nishita T. Simulation of Cumuliform Clouds Based on Computational Fluid Dynamics. In: Eurographics (Short Papers). Citeseer; 2002. .

32. Harris MJ. Real-time cloud simulation and rendering. University of North Carolina at Chapel Hill; 2003.

33. Ferreira Barbosa CW, Dobashi Y, and Yamamoto T. Adaptive cloud simulation using position based fluids. Computer Animation and Virtual Worlds. 2015;**26**(3-4):367–375.

34. Treuille A, McNamara A, Popović Z, and Stam J. Keyframe control of smoke simulations. Acm Transactions on Graphics. 2003;**22**(3):716–723.

35. Mcnamara A, Treuille A, and Stam J. Fluid control using the adjoint method. Acm Transactions on Graphics. 2004;**23**(3):449–456.

36. Shi L, and Yu Y. Taming liquids for rapidly changing targets. In: Acm Siggraph/ Eurographics Symposium on Computer Animation; 2005. p. 229–236.

37. Fattal R, and Lischinski D. Target-driven smoke animation. Acm Transactions on Graphics. 2004;**23**(3):441–448.

38. Raveendran K, Thuerey N, Wojtan C, and Turk G. Controlling liquids using meshes. In: ACM Siggraph/eurographics Symposium on Computer Animation; 2012. p. 255–264.

39. Yau MK, and Rogers RR. A short course in cloud physics. Elsevier; 1996.

40. Kessler E. On the continuity and distribution of water substance in atmospheric circulations. Atmospheric research. 1995;**38**(1-4):109–145.

41. Goswami P, and Neyret F. Real-time landscape-size convective clouds simulation and rendering. In: Eurographics-European Association for Computer Graphics; 2017. .

42. Koschier D, Deul C, Brand M, and Bender J. An hp-adaptive discretization algorithm for signed distance field generation. IEEE transactions on visualization and computer graphics. 2017;**23**(10):2208–2221.

43. Yang B, Liu Y, You L, and Jin X. A unified smoke control method based on signed distance field. Computers & Graphics. 2013;**37**(7):775–786.

44. Jakob W. Mitsuba renderer. Http://www.mitsuba-renderer.org; 2010.

45. Yuan C, Liang X, Hao S, Qi Y, and Zhao Q. Modelling Cumulus Cloud Shape from a Single Image. Computer Graphics Forum. 2014;**33**(6):288–297.

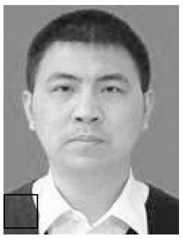
AUTHOR BIOGRAPHY

Zili Zhang received his master degree in computer science and engineering from Yunnan University, P.R. China. He is currently a Ph.D. student at the school of computer science and engineering at Beihang University. His main research interests include computer graphics, visualization and fluid simulation.

Yunfei Li received his bachelor's degree in computer science and technology from Northeastern University, P.R. China. He is a graduate student at the School of Computer Technology, Beihang University, China. His research interest is fluid simulation and interaction.

Bailin Yang received his Ph.D. degree in the Department of Computer Science from Zhejiang University in 2007. He is currently a professor in the Department of Computer and Electronic Engineering of Zhejiang Gongshang University. His research interests are in mobile graphics, real- time rendering and mobile game.

Frederick W. B. Li received a B.A. and an M.Phil. degree from Hong Kong Polytechnic University, and a Ph.D. degree from City University of Hong Kong. He is currently an Assistant Professor at Durham University. Frederick Li served as a guest Editor for some special issues of World Wide Web Journal, Journal of Multimedia and JDET. He has served as a Program Co-Chair of ICWL 2007-2008, 2013, 2015 and IDET 2008-2009. His research interests include distributed virtual environments, computer graphics and e-Learning systems.



Xiaohui Liang is currently a Professor, working at the school of computer science and engineering at Beihang University, P.R. China. He received his master and Ph.D. degrees in computer science and engineering from Shanxi University and Beihang University respectively. His main research interests include computer graphics and animation, visualization and virtual reality.