Science in China Series F: Information Sciences

© 2008 SCIENCE IN CHINA PRESS

Springer

www.scichina.com
info.scichina.com
www.springerlink.com

Physically-based Fluid Animation: A Survey

TAN Jie YANG Xubo[†]

[1] Department of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, China;

In this paper, we give an up-to-date survey on physically-based fluid animation research. As one of the most popular approaches to simulate realistic fluid effects, physically-based fluid animation has spurred a large number of new results in recent years. We classify and discuss the existing methods within three categories: Lagrangian method, Eulerian method and Lattice-Boltzmann method. We then introduce techniques for seven different kinds of special fluid effects. Finally we review the latest hot research areas and point out some future research trends, including surface tracking, fluid control, hybrid method, model reduction, etc.

Physically-based animation, Navier-Stokes equations, finite difference method, smoothed particle hydrodynamics, Lattice-Boltzmann method, particle level set.

1 Introduction

Close-up scenes of fluid phenomena such as stormy oceans, curly rising smokes and droplet splashes are amongst the most spectacular visual effects both in the real life and in the special effects industry. Photographers, movie makers and game developers all try their best to catch these moments of beauty (e.g. the famous photograph “milk crown” and the movie “Poseidon”). It is obvious that the realistic fluid animation is getting more and more demanding as people have higher and higher requirements on the visual effects of movies and games. However, the extreme complexity of fluid dynamics renders it impossible for the artists to animate fluid effects frame by frame. Thus, physically-based methods are now becoming the widely used techniques for generating realistic fluid animations.

Physically-based methods model the dynamics of fluids by solving the governing equations. Although the Navier-Stokes equations are proposed hundreds years ago to depict the fluid phenomena, the general closed form solutions remain undiscovered. With the development of the computer technology, various numerical methods are applied in approximating the Navier-Stokes solutions; and a new subject, namely computational fluid dynamics (CFD), was established. Admittedly, a lot of ideas and algorithms in physically-based fluid animation are brought from CFD literature. However, the purposes of CFD and physically-based fluid animations differ. CFD aims at predicting the actual flow fields through simulations while computer graphics (CG) mainly focuses on generating plausible visual effects. Thus, compared with the methods in CFD, the algorithms in physically-based

Received May 30, 2008; accepted October 14, 2008
doi:

[†] Corresponding author (email: yangxubo@cs.sjtu.edu.cn)

This work is supported by 863 National High Technology R&D Program of China (No. 2006AA01Z307)

fluid animation usually trade accuracy for speed. Furthermore, the ability of the animation control, the ease of the implementation and the generality of the framework are other considerations of computer graphics. In other words, the researches on CFD form the foundations of the physically-based fluid animation. But still a lot of work should be done to fill the gap between the different purposes and applications in these two areas. Liu et al.[1] provided a detailed review of earlier works in the field of physically-based fluid animation.

In addition to modeling the dynamics of fluids, rendering is another important issue in fluid animation. Various fluid phenomena demonstrate disparate visual effects. Thus, different rendering methods are adopted according to the categories and representations of fluids. Liquids usually have clear interfaces at liquid-liquid or liquid-air boundaries. If the interfaces are represented by explicit triangle meshes, the traditional rendering pipeline with shaders to calculate the reflection and refraction fits well. If the interfaces are represented as implicit surfaces such as a signed distance function[2], they can be ray traced effectively. Smokes, which do not have a clear boundary interface, are usually defined as a density field. Ray marching algorithm is used to render the smoke by sampling the density field and accumulate the density values. There are quite a few other methods and accelerating algorithms in the realm of fluid rendering, which are beyond the scope of this paper. This paper focuses on how to model the dynamics of fluids in a physically-realistic way. Readers interested in photo realistic rendering may refer to Pharr and Humphreys's book[3].

2 Three methods in physically-based fluid animation

Physically-based fluid animations are based on three fundamental governing equations of fluid dynamics–the continuity, momentum, and energy equations. They are the mathematical statements of the fundamental physical principles governing the fluid motions:

1. Mass is conserved.
2. Newton's second law, $\mathbf{F} = m\mathbf{a}$
3. Energy is conserved.

Although the forms of the equations could vary according to the viewpoints of fluids (Eulerian or Lagrangian), it can be shown that they're equivalent through some simple mathematical manipulations.

2.1 Lagrangian method

The Lagrangian approach treats the continuum as a particle system. Each point in the fluid is labeled as a separate particle, with a position $\mathbf{x}$ and a velocity $\mathbf{u}$. With the Lagrangian viewpoint, the incompressible Navier-Stokes equations are derived as

$$\nabla \cdot \mathbf{u} = 0$$
$$\frac{D\mathbf{u}}{Dt} = \nu \nabla^2 \mathbf{u} - \frac{\nabla p}{\rho} + \mathbf{f}$$

where $\mathbf{u}$ is the velocity, ν is the viscosity, ρ is the density, p is the pressure and $\mathbf{f}$ is the body force. The left hand side of the momentum equation can be interpreted as the acceleration of a particle while the right hand side is the net force exerted. We note that the equations have been simplified assuming that the fluid is incompressible since most visually appealing fluid effects in real life have little compressibility[15].

Reeves[4] introduced the particle system which is then widely used to model the deformable bodies, clothes and other chaotic phenomena. The particle system is an irregular discretization of the continuum. In order to solve the Navier-Stokes equations, the gradient operator ∇ and laplacian operator ∇^2 should be well defined under such an irregular discretization. Monaghan[5] introduced the Smoothed Particle Hydrodynamics (SPH) method into the computer graphics community to address this issue. It defines a smoothing kernel to interpolate the physical properties (velocities, densities, etc.) at an arbitrary position from the neighboring particles. We briefly review this method here because SPH is now becoming a more and more popular technique in the field of fluid simulation [6−8].

The fluid is represented by a set of particles $i \in [1...N]$ with positions $\mathbf{x}_i$, masses m_i and additional attributes A_i (velocities, densities, etc.). SPH defines how to compute a smooth continuous field $A(\mathbf{x})$ from the discrete attribute values A_i sampled at particle locations $\mathbf{x}_i$ as

$$A(\mathbf{x}) = \sum_i m_i \frac{A_i}{\rho_i} W(\mathbf{x} - \mathbf{x}_i, h)$$

The kernel function $W(\mathbf{r}, h)$ is typically a smooth, radial symmetric, normalized function with finite support. For example, in Müller's work[6], the kernel was designed as

$$W(\mathbf{r}, h) = \frac{315}{64\pi h^9} \begin{cases} (h^2 - |\mathbf{r}|^2)^3 & 0 \leqslant |\mathbf{r}| \leqslant h \\ 0 & \text{otherwise} \end{cases}$$

The gradient and laplacian of the smoothed attribute function $A(\mathbf{x})$ are

$$\nabla A(\mathbf{x}) = \sum_i m_i \frac{A_i}{\rho_i} \nabla W(\mathbf{x} - \mathbf{x}_i, h)$$

$$\nabla^2 A(\mathbf{x}) = \sum_i m_i \frac{A_i}{\rho_i} \nabla^2 W(\mathbf{x} - \mathbf{x}_i, h)$$

Consequently, the right hand side of momentum equation can be easily discretized with the above definitions.

Another problem associated with the SPH is how to solve the pressure term or how to enforce the motion of particles to satisfy the incompressibility constraint. A bunch of works[6,9−11] employed the ideal gas equation to relate pressure and density. This results in high compressibility and oscillations which cause severe visual artifacts. Cummins and Rudman[12] presented a projection method that was also used in Eulerian approaches. Similarly, in Premoze et al.'s work[8], a velocity estimate was projected onto a divergence free subspace by solving a Poisson equation. Becker and Teschner[13] proposed to use the Tait equation with a high speed of sound, which resulted in a weakly compressible formulation with very low density fluctuations.

This meshless method of using Lagrangian particles can operate more easily with irregular boundaries, between multiple fluids interaction and generally requires less computational resources. Though Lagrangian approach has been widely used in many interactive applications, due to the difficulties in surface reconstruction and rendering, particle-based methods have not yet demonstrated the same level of realism as its grid-based counterparts.

2.2 Eulerian method

The Eulerian approach follows another strategy. Instead of treating the fluid as flowing particles and then tracking each particle, it looks at fixed points in space and sees how the fluid quantities (including densities, temperatures and velocities) measured at those points change with time. Thus, the whole fluid region is modeled as fields of fluid quantities. For a specific time and a given position, there exists a group of values to represent the fluid state. For instance, the vector field $\mathbf{v}(x, y, z, t)$ is to characterize the velocities and the scalar field $p(x, y, z, t)$ is to measure the pressure inside the fluid. The incompressible Navier-Stokes equations have the following form with Eulerian viewpoint:

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

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla^2 \mathbf{u} - \frac{\nabla p}{\rho} + \mathbf{f}$$

The detailed derivation of above equations can be found in any textbooks of fluid dynamics, such as Anderson's work[14].

In Eulerian methods, the above equations are discretized with the grids. The finite difference methods are used to solve the equations numerically. Recently, there are two ways to store the fluid quantities on the grid. The most popular way is to store the scalars, such as pressures, level set values and temperatures at the center of each grid and to store the vectors, such as velocities at the faces of each grid cell. This staggered configuration of MAC grid was first presented by Harlow and Welch[16] which benefited from its unbiased and second order accurate central difference scheme. Most of the state-of-art simulations adopted the staggered grid. Another way is to store all the quantities at the node of each grid cell, such as in Stam's paper[17]. The advantage is simplicity. There is no need to handle different variables differently. Interpolations are simplified significantly as well.

At the early stages of physically-based fluid animation, researches did not tackle the sophisticated Navier-Stokes equations directly. On the contrary, they made several assumptions and reduced the governing equations to the wave equation[18] or the shallow water equations[19]. The height field was used to represent the water surface. Although this algorithm was quite simple and efficient, a myriad of interesting fluid phenomena, such as overturning waves, sprays and splashes, could not be captured. Foster and Metaxas's work[20] was the first example that solved the full 3D Navier-Stokes equations to animate fluids. Stam[21] improved it, achieving the unconditionally numerical stability by introducing the semi-Lagrangian method for the convection term and implicit solver for the viscosity and pressure terms. It became the standard framework to implement fluid animation codes. The Eulerian method can be divided into four steps, from the initial velocity $\mathbf{u}_0(\mathbf{x})$ to the resultant velocity $\mathbf{u}_4(\mathbf{x})$ after one time step:

$$\mathbf{u}_0(\mathbf{x}) \overbrace{\longrightarrow}^{\text{add force}} \mathbf{u}_1(\mathbf{x}) \overbrace{\longrightarrow}^{\text{advect}} \mathbf{u}_2(\mathbf{x}) \overbrace{\longrightarrow}^{\text{diffuse}} \mathbf{u}_3(\mathbf{x}) \overbrace{\longrightarrow}^{\text{project}} \mathbf{u}_4(\mathbf{x})$$

The four sub-steps are:

$$\begin{array}{ll} \text{Add force:} & \mathbf{u}_1(\mathbf{x}) = \mathbf{u}_0(\mathbf{x}) + \Delta t\mathbf{f} \\ \text{Advect:} & \mathbf{u}_2(\mathbf{x}) = \mathbf{u}_1(\mathbf{p}(\mathbf{x}, -\Delta t)) \\ \text{Diffuse:} & (\mathbf{I} - \nu\Delta t\nabla^2)\mathbf{u}_3(\mathbf{x}) = \mathbf{u}_2(\mathbf{x}) \\ \text{Project:} & \nabla^2 p(\mathbf{x}) = \frac{\rho}{\Delta t}\nabla\cdot\mathbf{u}_3(\mathbf{x}) \\ & \mathbf{u}_4(\mathbf{x}) = \mathbf{u}_3(\mathbf{x}) - \frac{\Delta t}{\rho}\nabla p \end{array}$$

Combining the Eulerian method and level set based surface tracking algorithms[25,26] has produced stunning results, simulating various interesting fluid phenomena such as smoke, water, fire, droplets, non-Newtonian flow, bubbles, etc. In general, these techniques have progressed to the point where fluid phenomena can be modeled so realistically that a naïve viewer may have difficulties in telling reality from simulated footage.

2.3 Lattice Boltzmann method

Li et al.[34] introduced the Lattice Boltzmann method (LBM) into the computer graphics community. LBM is a relatively new approach to approximating the Navier-Stokes equations. Unlike traditional CFD methods, which solve the governing equations of macroscopic properties (mass, momentum and energy), the LBM is based on microscopic models and mesoscopic kinetic equations (the Lattice Boltzmann equation). The fundamental idea is to construct simplified kinetic models that incorporate the microscopic and mesoscopic physical processes so that the macroscopic averaged properties obey the desired macroscopic equations (the Lattice Boltzmann equation converges to the Navier-Stokes equation).

The Lattice Boltzmann equation is the governing equation of LBM:

$$f_i(\mathbf{x} + \mathbf{e}_i\Delta\mathbf{x}, t + \Delta t) = f_i(\mathbf{x}, t) + \Omega_i(f(\mathbf{x}, t))$$

where f_i is the velocity distribution function in the ith direction $\mathbf{e}_i$ and $\Omega_i(f(\mathbf{x}, t))$ is the collision operator representing the rate of change of f_i resulted from collision. It is non-trivial to prove that the Lattice Boltzmann equation is an approximation of Navier-Stokes equations when viewed macroscopically. Chen and Doolen[35] showed the detailed derivations. The whole solving process can be divided into two steps, the stream step and the collide step. In the first step, all velocity distribution functions are convected with their respective velocities. This propagation results in a movement of the real values to the neighboring cells. Formulated in terms of distribution functions, it can be written as

$$f_i^*(\mathbf{x}, t + \Delta t) = f_i(\mathbf{x} - \Delta t\mathbf{e}_i, t) \qquad (i = 0, 1, ..., M)$$

The second step performs the BGK collision operator[36] which calculates a linear combination of f_i^* and the local equilibrium distribution functions f_i^{eq}. Next, the new velocity distribution function at the end of current time step is achieved:

$$f_i(\mathbf{x}, t + \Delta t) = (1 - \omega)f_i^*(\mathbf{x}, t + \Delta t) + \omega f_i^{eq}$$

where the relaxation parameter ω is viscosity related and in the range of $(0, 2]$. The flow is very viscous when ω is close to zero, while ω near 2 results in more turbulent flows. However, in modeling the turbulent flows, traditional BGK model suffers from stability problems. The basic LBM algorithm is extended by applying the Smagorinsky sub-grid model[34,37] to improve the numerical stability. Finally, the velocity field can be recovered from the velocity distribution function

$$\mathbf{u}(\mathbf{x}) = \sum_i f_i(\mathbf{x})\mathbf{e}_i$$

Due to its particulate nature and local dynamics, the LBM has several advantages over other conventional physically-based fluid animation methods, especially in dealing with complex boundaries, incorporating of microscopic interactions, and parallelizing the algorithm. For example, multiphase flows[38] have always been a challenge to Eulerian and Lagrangian methods because of its moving and deformable interfaces. On the contrary, the LBM provides a relatively easy and consistent way to incorporate the underlying microscopic interactions by modifying the collision operator. Successful applications of multiphase LBM models can be found in various complex fluid systems, simulating interface instability, bubble/droplet dynamics, wetting on solid surfaces and so on.

2.4 Comparisons of three methods

Unlike CFD for engineering purpose, in which the computational result serves as a prediction of the experiments and a guideline of the designs, the numerical accuracy and the deviation from the experiments can be used as a benchmark; as for physically based fluid animations, realistic appearances and plausible visual effects are the most important criteria. It is difficult to rigorously compare which simulation approach dominates. There is no clear winner for the three approaches introduced above since every method has its advantages and disadvantages.

The Lagrangian method (SPH) uses the traditional particle systems to model the fluids. Both the concept and the implementation are straightforward. It is easy for SPH to demonstrate the turbulent splashing flows[39] and to catch small details of fluid phenomena such as bubbles[40,41] and foams[42]. Furthermore, the demands of computational resources of SPH with moderate number of particles are generally less than its Eulerian or LBM counterparts. So in games or other interactive system, a lot of fluid phenomena are simulated using the Lagrangian method. However, there are three major drawbacks that prevent this approach from being superior to others. First, the smoothing kernel should be designed carefully because the stability, accuracy and speed of the SPH method largely depend on the choice of the smoothing kernels. It is often desired that more than one kernel be provided to interpolate different fluid attributes[6,43]. Second, incompressibility cannot be strictly guaranteed by just relating the pressure and density with the ideal gas equation. Although Becker and Teschner[13] provided an alternative approach, it required much more stringent time steps than the Eulerian approach. The last drawback of the Lagrangian method is its difficulty in constructing a smooth surface for rendering. Many research works have presented ad hoc solutions, but up till now, the quality of liquid surfaces constructed from the whole bunch of particles is not as competitive as its Eulerian counterpart.

The major advantages of the Eulerian methods (combined with level set based surface tracking methods) are the smooth liquid surfaces and large time steps. But it suffers from lengthy computational time, aliasing boundary discretization and poor scalability. Eulerian methods solve the Poisson equation in the projection step to enforce the fluid incompressibility. The linear system could contain millions of unknowns, depending on the grid resolution. Even though the linear system is sparse and symmetrical positive definite which enables fast solver such as MIC(0) preconditioned conjugate gradient method, the solution time is still very long and amounts for the largest potion of the total simulation time. Besides, most of the research works used uniform MAC grid to discretize the Navier-Stokes equations. But the axis-aligned cubic cells cannot fit non-axis-aligned physical boundaries well. Stair-step artifacts appear in the voxelized solution even though the real boundary is smooth. An unstructured tetrahedral mesh[44,45] helps, but the computational overhead is considerable. Batty et al.[46] introduced a fast variational framework in the light of regular grids for accurate solid-fluid coupling. But coupling fluid with thin shell-like boundaries (such as cloth, paper, etc.) still imposes challenges and difficulties to the researches. Another severe problem of Eulerian method is its poor scalability. Doubling the grid resolution means octuple memory consumption and even more than eight times of the simulation time. This means it is infeasible to tackle large bodies of water using the uniform MAC discretization. Thus, some adaptive approaches[47] and hybrid methods[48] are introduced to alleviate this problem.

The LBM is becoming more and more popular in the field of physically-based fluid animations in recent years. Its basic algorithm is simple to understand and implement. Since the algorithm is parallel, it is suitable to be implemented on the graphics hardware. Because the LBM models the microscopic and mesoscopic behavior of fluids, some fluid phenomena that are difficult for the classical Eulerian or Lagrangian method are handled naturally with LBM. The main drawbacks of LBM are the poor scalability and small time steps. In LBM, the whole computational domain is divided into cubic grids to store the velocity distribution functions. So it shares the same scalability problem with the Eulerian method. On the other hand, the time step must be kept small enough to ensure the stability of the simulation. But it should be noted that a single LB step is usually significantly faster than a single update step of the Eulerian Navier-Stokes solver.

Both the advantages and disadvantages of each method are briefly tabulated and the representative result of each method is shown in Figure 1.

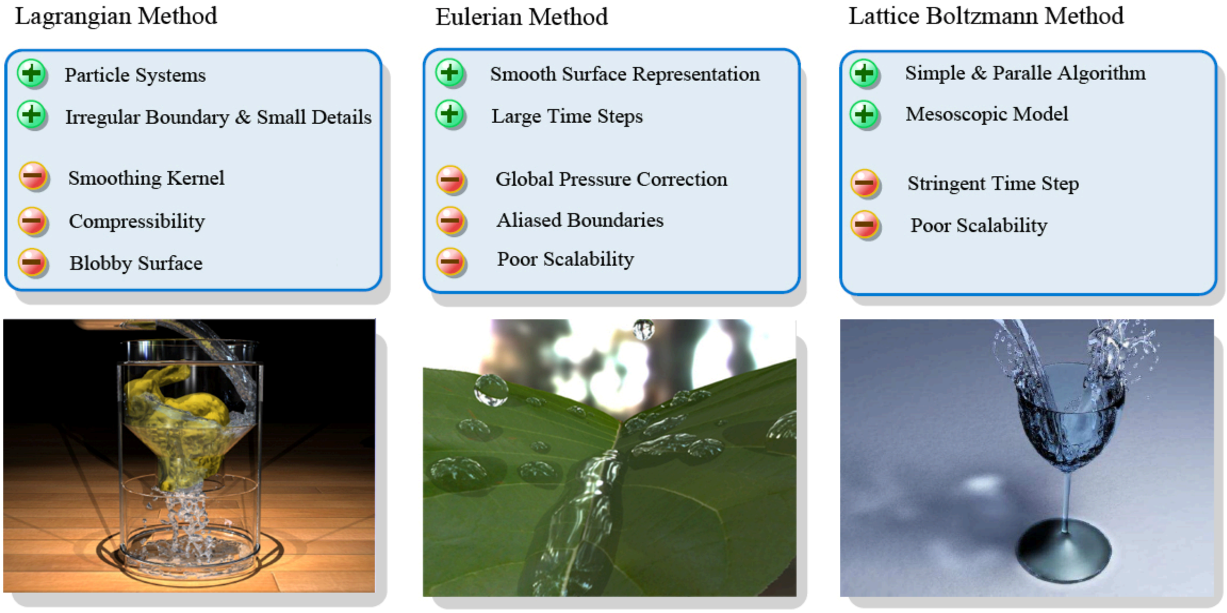

Fig. 1. Comparisons of three fluid animation methods. Left-bottom: porous flow (Image courtesy of Toon Lenaerts, Bart Adams and Philip Dutré[83]), middle-bottom: water drop (Image courtesy of Huamin Wang, Peter J. Mucha and Greg Turk[32]) and right-bottom: splashes in a cup (Image courtesy of Nils Thürey and Ulrich Rüde[120]).

3 Different Types of Fluids

A lot of natural phenomena are categorized as fluid motions and their dynamics are described by Navier-Stokes equations, including cloud, smoke, fire, explosion, sand, droplet, bubble, foam, viscoelastic/viscoplastic flow, fluid-object interaction and so on. Exploring the fluid-like natural phenomena and simulating them with the state-of-art fluid animation methods is an exciting and promising research direction. Due to the different visual effects and behaviors of these phenomena, both the mathematical equations and the numerical methods are modified to model their disparate innate physical properties.

3.1 Clouds and Smokes

Clouds and smokes are the most commonly seen phenomena in our daily life. They are relatively easy to model because both of them have small viscosity and do not possess a clear interface. Harris and Baxter[49,50] introduced the physical principles behind the formation of clouds, considering the humidity in the atmosphere. They also proposed a GPU accelerated Navier-Stokes solver for interactive cloud animations. Dobashi et al.[51] used lattice gas solvers based on cellular automata. Miyazaki et al.[52] used an approach similar to Fedkiw et al.[22] which included vorticity confinement forces.

The hot and turbulent gas motion can be broken down into two components: 1) convection due to Newtons laws of motion and 2) rotation and swirling due to drag and thermal buoyancy[27]. Because the viscosity of smoke is often negligible, Fedkiw et al. [22] dropped the parabolic term in Naver-Stokes equations, added a vorticity confinement force[53] and adopted a higher order interpolation scheme to compensate the damping of the vorticity due to the numerical dissipation.

3.2 Fire and Explosion

Similar to Fedkiw's method[22], Nguyen et al.[28] used the inviscid Euler equations and the vorticity confinement method, achieving turbulent flame effects. The visible flame consists of three distinct effects: the blue core, the hot gaseous products and the smokes. They proposed a multiphase physically-based model. In addition to the Euler equations, another set of boundary conditions describing the mass conservation during the phase change are added to close the equation. It is worthy of mentioning that a very practical procedural model for fire was proposed by Lamorlette and Foster[54].

Explosion results from combustion of extremely fast speed. It produces supersonic blast waves and a variety of visual effects. An initial chemical or nuclear reaction often causes a blinding ash of light and dust clouds racing across the ground. Massive objects are moved, deformed, or fractured. Neff and Fiume[56] modeled and visualized the blast wave based on an empirical blast curve. Yngve et al.[55] simulated the propagation of an explosion through the surrounding air using the compressible Navier-Stokes equations. Compressible Navier-Stokes equations are composed of three equations, with an additional energy equation. Their system implemented solid-fluid two-way interaction and integrated the brittle fracture model of Obrien and Hodgins[57]. While the compressible flow equations are useful for modeling shock waves, they introduced a very strict time step restriction. Rasmussen et al.[58] combined two-dimensional high resolution physically based simulation with a moderate sized three-dimensional Kolmogorov velocity field for an efficient simulation of large scale explosions.

3.3 Viscous, elastic, plastic flows

Many materials, including wax, glass, cement and so on, exhibit variable viscosity. They melt, flow and harden according to the changing temperature. In early graphics literature, Desbrun and Cani[59] simulated the softening and melting behavior with particle methods, which avoided to maintain connectivity information for solid phase. They also simplified the topological change and the transition between different material behaviors. Carlson et al.[29] added several capabilities to the classical Eulerian method in order to animate such phenomena. First, they changed the viscosity term from $\nu\nabla^2\mathbf{u}$ to $\nabla \cdot (\nu\nabla\mathbf{u})$ to enable the viscosity of the animated material to vary in space. Second, they tied temperature to the viscosity to allow for melting and hardening. Later, Müller et al.[7] proposed a mesh-free and continuum-mechanics-based model with a dynamically adapted, point-sampled surface for the animation of elastic, plastic and melting materials.

Another kind of materials shows a combination of fluid and solid characteristics. These materials elastically resist deformation up to a certain threshold after which they begin to flow. A large variety of materials exhibit this type of behaviors and a few common examples include: mucus, egg white, dough, gelatin, unset cement, liquid acrylic, toothpaste, gels, clay, and liquid soap. Clavet et al.[60] simulated elastic and plastic behaviors with a particle system. The elastic and plastic effects are obtained by adding springs with various rest length between the particles. Goktekin et al.[30] added an elastic/plastic term $\frac{\mu_e}{\rho}\nabla^T\epsilon$ to the Navier-Stokes equations to simulate viscoelastic fluids. In Bargteil et al's work[61], the linear basis functions of the Lagrangian Finite Element Methods were updated each simulation step and simulation domain was remeshed when they became ill-conditioned, allowing for large plastic deformations for the viscoplatic flow. Wojtan and Turk[62] coupled a high resolution surface with low resolution physics simulation which led to fast and detailed animations of complex elastic and plastic behavior.

3.4 Sand

Early in graphics community, the granular materials such as sand were directly simulated by simple particle systems[63,64]. Zhu and Bridson[65] animated sand as fluids. They introduced a simplified frictional plasticity model and adopted the Particle-In-Cell (PIC)[66] and the Fluid-Implicit-Particle (FLIP) method[67] which used particles to advect the velocity field instead of Semi-Lagrangian method. The PIC suffered from excessive numerical dissipation due to the back-and-forth interpolations, which was cured by the FLIP method. A weighted average of the two was used to achieve the desired viscosity.

3.5 Small-scale Flow

Droplets belong to the small-scale phenomena of liquids in which the surface tension plays an important role. Wang et al.[32] presented a physically-based method to enforce contact angles at the intersection of fluid free surfaces and solid objects. The heart of the technique was the virtual surface method, which modified the level set distance field in order to maintain an appropriate contact angle. The surface tension calculated on the contact line captured all interfacial tensions, including liquid-solid, liquid-air and solid-air tensions. With the virtual surface method, it was straightforward to simulate the capillary effects and droplet spreading when impacting on a solid surface. Another research to model the small-scale details of fluid motion was proposed by Hong and Kim[31]. They focused on the discontinuities of the pressure and viscosity across the interface of multiphase flow. To obtain the derivatives at discontinuous regions with sub-grid accuracy, the variables are extrapolated across interfaces like the ghost fluid method[68].

3.6 Bubbles and Foams

In real fluids, we often observe bubbles rising and floating on the surface. The lively but chaotic motion of bubbles has enchanted and challenged many scientists and researchers. Hong and Kim[69] modeled the bubble in the water as the multiphase flow and combined the volume-of-fluid[70] method and the front-tracking[71] method to track the evolving interfaces. Greenwood and House[72] proposed a simpler method in which they generated passive air-particles and advected them using the Eulerian velocity field. Müller et al.[73] adopted the SPH to handle the full two-way coupling of water and air. Thuerey et al.[40] implemented an interactive bubble system by coupling SPH bubbles with the shallow water simulation using locally defined vortices on particles. Kim and Carlson[74] used the modular design that decoupled bubble dynamics from water surface dynamics, both visually and computationally. Hong et al.[41] proposed a new bubble model based on incorporating SPH into an Eulerian grid-based simulation. The SPH modeled small-scale bubble motions while the grid simulation handled background flows of large bodies of water and air. This hybrid model overcome the difficulty in simulating small bubbles in the context of the multiphase flows on a coarse gird, achieving sub-grid visual details. The bubble simulation using LBM was studied in Thürey and Rüde's work and Pohl et al.'s work[75,76].

If the bubbles float on the water surface and do not burst, they will stack, forming the wet foam. The water between those stacked bubbles will drain, leaving a micrometer-thin film of liquid between bubbles, forming the dry foam. Simulations of wet or dry foams are very challenging because the level set surface tracking method suffers from a small but steady volume loss that leads to obvious artifacts. Kim et al.[33] proposed the volume control method to address this problem. They tracked the volume change of each connected region and applied a carefully computed divergence that compensated the undesired volume loss.

3.7 Fluid-Object Interaction

Fluids interact with other objects every day: athletes diving into the swimming pool, flags waved in the wind, etc. Takahashi et al.[77] presented a simple two-way coupling between fluids and solid objects. The velocities of the solid objects served as the boundary conditions for fluid motion while the pressure field solved from the Navier-Stokes equations was integrated at the solid surface to provide a net force and a net torque exerted on the solid objects. However, the alternative nature of the coupling was inconsistent between the motions of fluids and solid objects. Génevaux et al.[78] represented the solids by mass-spring models and fluids by marker particles. The interactions were calculated through the mutual forces between the marker particles and mass nodes at the interface. However, it is awkward to find a mass-spring representation for an arbitrary solid object. Carlson et al.[79] proposed the rigid fluid method that treated solids as fluids at first and then projected the velocity field in the solid region onto a subspace satisfying the rigid constraints. But the method cannot handle light solids stably. Guendelman et al.[80] returned to the alternating approach, generalized it to include octree, thin solids and arbitrary solid dynamics. They solved the pressure field for a second time by adding solid masses to the fluid grid density similar to the Immersed Boundary Method[81], which improved the noisy pressure. Klingner et al.[45] used the tetrahedral mesh for accurate boundary discretization and extended the mass conservation (projection) step to include the dynamics of rigid body. Thus the interaction between the fluids and solids can be solved simultaneously rather than in an alternating manner. It was extended to model the interaction between fluids and deformable bodies[82]. Batty et al.[46] derived a fast variational approach that allowed accurate boundary conditions and two-way coupling in the light of regular grids. While all the work mentioned above deal with interaction between fluids and impenetrable objects, recently, Lenaerts et al.[83] simulated the full two-way coupling between fluids and porous deformable material such as sponge with the SPH framework. The porous objects were sampled by particles which represented local porosity and permeability distributions at a macroscopic scale. The number of computational elements were kept low while a realistic simulation was achieved.

4 Current Research Areas and Directions

Although most of the researches in physically-based fluid animations use either one single approach or a combination of several that introduced above, none of them is perfect. Researchers still face a variety of challenges and difficulties in simulating various fluid phenomena. In this section, we'll briefly review some active research areas and directions in recent years.

4.1 Surface tracking and representation

Surface tracking and representation is an interesting and hot research area in liquid animation. Unlike some volumetric effects such as smoke or clouds, liquids have clear interfaces. Due to the complexity of fluid motions, it is non-trivial to model the liquid surface by explicit triangle meshes because the liquid is likely to merge or split, which will invalidate the topologies of original mesh during the simulation.

Classical metaballs[84] is commonly used in particle-based method but produces very blobby appearance. Distance based surface tracking was proposed by Adams et al.[85] to generate smoother water surface. Müller et al.[86] presented a powerful approach for the generation of surfaces defined by the boundary of a three-dimensional point cloud. A depth map with internal and external silhouettes was first generated in screen space. Then it was used to construct a 2D screen space triangle mesh with a technique derived from Marching Squares. The algorithm only generated surface where visible, such that view-dependent level of detail came for free and interesting visual effects were possible by filtering in screen space.

The front-tracking[71] method used particles to explicitly discretize the free surface and stored a connectivity list between these particles[87]. It is difficult to maintain the connectivity list when the free surface undergoes large deformations or topological changes. To avoid this difficulty, the point-set method was introduced by Torres and BrackBill[88]. Although this approach unchains the front tracking method from its dependence on point connectivity, the point regeneration algorithm is complex and computationally expensive.

The VOF[70] method can handle topological changes naturally with the marching cubes algorithm[91]. Basically, it only uses one scalar value - the volume of fluid - for one cell, by which we can estimate the exact position of the liquids and calculate the total volume of fluid inside the simulation domain. The level contour reconstruction method[89] is similar to the combination of the VOF method and the marching cubes algorithm used in Kunimatsu et al.'s work[90], which possesses the inherent capability of dealing with topological changes.

Osher and Fedkiw[2] introduced the level set method and dynamic implicit surface which were suitable to represent liquid surfaces. This approach represents the liquid surface implicitly as a signed distance field $\phi(\mathbf{x})$. $\phi(\mathbf{x})$ is defined as the shortest distance from position $\mathbf{x}$ to the liquid surface. Thus, the liquid surface is characterized by the zero contour of the signed distance field. Positive $\phi(\mathbf{x})$ means $\mathbf{x}$ is outside the fluid region while negative $\phi(\mathbf{x})$ means inside. Given a velocity field $\mathbf{u}(\mathbf{x})$ solved from the Navier-Stokes equations, the implicit liquid surface is evolved by solving the following level set equation using the upwind schemes:

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0$$

In addition, the signed distance function has other good properties. For example, the surface normal and curvature can be calculated easily:

$$\mathbf{N} = \nabla \phi$$

$$\kappa = \nabla^2 \phi$$

Foster and Fedkiw[25] used dynamic level set to track the evolution of the surface. Even through the level set equation can be solved using a highly accurate fifth order WENO scheme, it still suffers from high volume loss and smoothing artifacts. They alleviated this problem by coupling particles with level sets. Enright et al.[26] further improved the level set based method by seeding particles at both sides of the interface. The particles revised the level set values each frame to preserve the detailed surface features as well as the total volume of fluid. Due to the presence of the auxiliary particles, this approach was named particle level set (PLS). Mihalef et al.[92] presented a marker level set method (MLS) to track the dynamic liquid surface. MLS only seeds the marker particles at the interface, yielding more efficient and accurate results than the popular PLS. Furthermore, the surface markers allow the MLS to handle non-diffusively surface texture advection. Bargteil et al.[93] proposed the Semi-Lagrangian Contouring (SLC) method. Different from the PLS, the SLC updates its signed distance field using the Semi-Lagrangian method and the triangle meshes are extracted by Marching Cube algorithm every time step. Besides the merits of implicit surface, it benefits from explicit polygon meshes, volume conservation, adaptive resolution and easy surface property convection.

We recommend interested reader to refer to Osher and Fedkiw's book[2] for more detailed discussion about free-surface tracking.

4.2 Fluid Control

Physically-based and control is a contradiction in fluid animations. Physically-based means the motion is governed by physical principles while control means the animation is based on artists' intentions. While realism is one important aspect, the use of fluid animation is also greatly determined by the ability to efficiently control the behavior of the fluid. In many cases, the accurate physical behavior is not even desired. For example, animate a fluid character as in the movie "Terminator" or create a huge wave from scratch in a specific position in "Poseidon". Unfortunately, controlling complex PDEs is very difficult. Although the parameters in the Navier-Stokes equations such as viscosity and body forces give the animators some degrees of control, higher-level control is also desired in a production environment, where animators are mostly interested in modifying the large-scale motion of the fluid, while the physically-based simulation should take care of the fine-scale details such as small vortices and splashes.

Foster and Metaxas[94] were the first that dealt with fluid control. They embeded pressure and velocity controllers to direct the fluid motion. While Feldman et al.[95] demonstrated the capabilities of particle based fluid control for explosion simulation, Rasmussen et al.[96] introduced more types of control particles, including viscosity, velocity divergence and level set particles, for simulating melting, expansion and contraction of the liquid. Treuille et al.[97] controlled smoke simulation through user-defined keyframes of smoke density and velocity field. They formulated an optimization framework and used the derivatives to solve for the force parameters to minimize the objective function. They also presented a novel multiple-shooting approach to improve the scalability of their method. The efficiency was greatly improved by adopting the adjoint method to compute the gradient, which enabled full 3D control of smoke and liquid animations[98]. Fattal and Lischinski[100] avoided the optimization problem and proposed the idea of driving smoke towards target smoke density states by finding a closed-form solution of the Euler Equations with a gathering term. This simple technique is much faster than solving an optimization problem, but still achieving comparable results. Pighin et al.[99] controlled the flow simulation using radial basis functions. Hong and Kim[101] derived potential fields from the initial smoke distribution to its target distribution and used the gradient of this potential field as the control force. Shi and Yu controlled smoke[102] by matching the implicit surface of the fluid with static or moving target shapes. Velocity constraints derived from a shape matching functional are imposed at the boundary, driving the smoke towards the desired shape. They[103] controlled liquids through two external forces: a feedback force which compensates for discrepancies in both shape and velocity and a negative gradient field of the geometric potential. Thürey et al.[104] decomposed the velocity field according to scales and only applied the control forces to the coarse-scale components of the flow. Therefore, small-scale details which are often smoothed out by force-based control methods are preserved.

4.3 Hybrid method

As mentioned in section 2.4, every approach has its advantages and disadvantages, which inspires the researchers to integrate several methods into one to combine their merits and avoid their respective demerits.

In Eulerian approach, there are many types of computational grids, such as 2D height fields[18], 3D MAC grids, tetrahedral meshes[44,45] and so on. Feldman et al.[105] coupled the tetrahedral mesh with regular MAC grid to discretize the Navier-Stokes equations. They used the tetrahedral mesh near the solid boundaries because the triangles fitted arbitrary boundaries well while using the regular grid in the large open space because MAC grid was more accurate and easier to implement. But the presence of the transition cell linking the tetrahedron cell and MAC cell limited its usage to static scenes only. Irving et al.[48] coupled the height field with the MAC grid to tackle large bodies of water with the assumption that the pressure profile is linear vertically in a gravity dominated flow. Tan et al.[106] unified above methods into a general multi-layer framework. The whole computational domain was divided into several nesting layers of grids of different discretizations. The Navier-Stokes equations were solved on the multiple layers at successive passes and the solutions on different layers were synchronized through prolongations and restrictions.

Thürey et al.[107] combined the 2D shallow water model with the 3D LBM to animate the open water phenomena, which sped up the simulation dramatically. Thuerey et al.[40] integrated a particle model for bubbles and foams into the shallow water framework that enabled real-time bubble simulations. Kang et al.[108] presented a hybrid approach for the animation of chemically reactive fluids, in which the Lagrangian methods were to bring about a chemical reaction that affected the overall flow of the fluid and the Eulerian methods were to model the fluid to be visualized in the rendering stage. Selle et al.[109] seeded Lagrangian vortex particles into the grid-based fluid simulator to create turbulence in smoke, water and explosions.

Grid-based methods have difficulties in resolving features below the scale of the underlying grid. Although adaptive methods (e.g. RLE, octrees) can alleviate this to some degree, separate techniques are still required for simulating small-scale phenomena such as spray and foam. Losasso et al.[110] proposed a two-way coupled simulation framework that used the Eulerian and particle level set method to efficiently model dense liquid volumes and a SPH method to simulate the diffuse regions such as sprays. Kim et al.[39] also coupled a particle system with Eulerian level set methods to create turbulent splashing water. The volume loss of particle level set method was estimated to guide the number of splashing particles generated. Hong et al.[41] used a hybrid approach coupling SPH with grid-based method to simulate the bubbly water. While the Eulerian approach on coarse grids was suitable for modeling large bodies of water and air, the particle-based bubble model added sub-grid visual details of small-scale bubbles.

4.4 Model reduction

Model reduction is an increasingly important technique in computer graphics. Although it has been used to reduce a wide range of problems, ranging from global illumination to elastostatics and dynamics, it is far from well-known in physically based fluid animations. The idea of model reduction is to project a high dimensional problem (even with infinite dimensions) onto a lower dimensional subspace and then solve the problem in the reduced subspace, which can save the computational resources and simulation time dramatically.

Treuille et al.[111] were the first that introduced the concepts of model reduction into the fluid animation world. They used an accurate offline solver to produce a set of high-resolution fluid simulations. These velocity fields were distilled into a small basis of size proportional to the system's principle modes of variation. Then the Galerkin projection was computed to reduce the Navier-Stokes equation onto the low dimensional subspace spanned by this set of bases. After these pre-computations, the equations could be solved very quickly in the subspace. The simulation runtime costs were proportional to the number of basis rather than the number of simulation voxels. They reported real-time animation with the model reduction techniques. However, there are some limitations. It is difficult to determine how many bases and which set of bases should be chosen. If the number of bases is too small or the user presents runtime inputs on which the system has not been trained, the simulation result is not satisfactory.

More recently, an interesting work[112] presented a unified framework of modeling and rendering the participating media on an analytic reduced space. The Navier-Stokes equations were projected onto subspace spanned by the Legendre polynomial basis. They derived analytic expressions for the derivative and integral operators in the Legendre coefficient space, as well as the triple product integrals of Legendre polynomials. The computational speed-up was up to three orders of magnitude which came at the cost of its limited ability to handle high frequency fluid phenomena.

4.5 GPGPU

The physically-based fluid animations involve intensive numerical calculation. Fortunately, many computations are parallel and can be implemented on the GPU efficiently. With the development of graphics hardware, especially after the release of DirectX 10 API and G80 series graphics cards, it is possible to port a large portion or even the whole simulation onto the GPU, achieving real-time animations.

Harris and Baxter[49,50] simulated and rendered the clouds on graphics hardware in real-time. Boltz et al.[113] mapped two sparse linear solvers: the conjugate gradients and multigrid onto the GPU. Both solvers play an important role in solving the Navier-Stokes equations. Wu et al.[115] accelerated the whole computation by packing the scalar and vector variables into four channels of texels and tested their results in 2D scenes. Liu et al.[114] extended their work to 3D scenes with complex obstacles. LBM were implemented on the older Geforce Ti 4600 GPU using the shading language[34] and the latest GPU Geforce 8800 Ultra with Compute Unified Device Architecture (CUDA)[116]. Speed-up factors of approximately 50 and 20 were reported respectively. With the support of DirectX 10 and the latest graphics card, Crane et al.[117] simulated and rendered various fluid phenomena including smoke, water and fire in realtime and provided a number of optimization options. However, in order to guarantee the interactive frame rate, they limited the iterative numbers in solving the Poisson equation. As a result, the total volume of the fluid was not strictly conserved, which was repaired by taking the weighted sum between the simulated result and a static equilibrium. The famous Nvidia's demo "Cascades" [118] used a particle system to simulate a real-time waterfall entirely on GPU. Geiss[119] presented a GPU-based Marching Cube algorithm using the geometry shader that could be used to implement the VOF or SLC surface tracking methods.

5 Research Trends and Potential Research Areas

Although it has been more than two decades since the start of researches on physically-based fluid animation, there still exist a large number of challenges and difficulties to overcome. For example, few of the realistic fluid effects can now be simulated in real-time and can be applied into interactive applications. The papers of SIGGRAPH and SIGGRAPH/Eurographics symposium on computer animation this year also provide us with some insights of future research trends. In this section, we point out the future research trends and some potential research areas of physically-based fluid animations.

5.1 High spatial frequency details

Most small-scale details, such as turbulent splashes, thin water sheets and small vortices, are the high spatial frequency motions of fluids. Although they are the most visually attractive, it is hard to capture them with traditional physically-based methods. According to Nyquist sampling theory, the sampling rate determines the highest frequency that can be reconstructed. In other words, more samples or finer grids are necessary in order to capture the high frequency details of fluid motions. However, large amounts of sampling particles or refined grids result in a linear increase in memory use and a greater than linear increase in the computational time, which renders the brute force method infeasible.

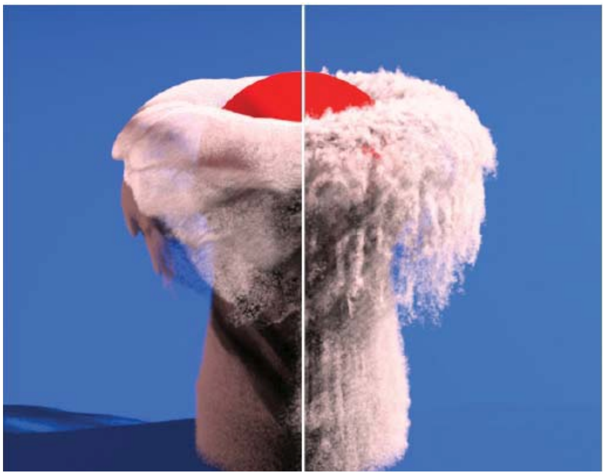

Fig. 2. Comparisons between the coarse simulation without and with the high frequency details (Image courtesy of Theodore Kim, Nils Thürey, Doug James et al.[121]).

Many researchers addressed this issue. The most representative method was to combine a coarse Eulerian Navier-Stokes solver with a high frequency procedural flow. Both Kim et al.[121] and Schechter et al.[122] determined the energy spectrum using the Kolmogorovs theory. While Schechter et al.[122] used the curl of the Perlin noise to generate a divergence free procedural velocity field of each frequency band, the former researchers adopted the wavelet noise instead. And they adopted different ways to preserve the temporal coherence of the procedural turbulence. Although the visual effects of such methods are quite convincing (please refer to Figure 2), there are several limitations. Since the high frequency details are synthesized according to the statistical model, it cannot reproduce the results of direct high resolution simulations. And the quality of obstacle interaction depends on the quality of coarse Eulerian simulation.

High spatial frequency details of the fluid motions are indispensable to generate realistic visual effects. From the number of the papers addressing this topic in past few years, we note more and more researchers started to focus on how to capture these visually attractive high frequency features. But up till now, it is a still an open question and there is no perfect solution yet.

5.2 Numerical dissipation

Usually, fluid animations emphasize more on performance than accuracy and more on stability than numerical dissipation. Thus, large time steps, schemes of low accuracy and implicit methods are employed. Unfortunately, all of them introduce huge amounts of numerical dissipation. The numerical dissipation is also called the numerical viscosity, which gradually dissipates the energy of the simulated system. The resultant simulation appears much more viscous than it should be and many visually attractive features are smoothed out.

Although using the semi-Lagrangian method for advection term is unconditionally stable with arbitrary large time steps, it is a first-order accurate discretization scheme both in time and space. When simulated with large time steps or large grid spacing, the fluid animation suffers from high numerical dissipation and results in sticky motions. Several methods were proposed to combat the numerical dissipation and to sharpen the semi-Lagrangian method. Fedkiw et al.[22] adopted the limited Catmull-Rom scheme for higher order interpolation. Kim et al.[23] introduced the Back and Force Error Compensation and Correction (BFECC) method that estimated the error and subtracted it. It improved the accuracy to second order both in space and time. Other higher order schemes, such as QUICK[123], WENO[2] and CIP[24], were adopted to suppress the numerical errors in discretizing the advection term.

Discretization of the advection term is not the only source of dissipation: the first order time splitting used in graphics, where velocities are separately advected and then projected to be incompressible, also introduces large errors[122]. Even though this splitting error can cause obvious artifacts, there is few research in this field. Consider a simple example: if a rigidly rotating fluid is advected by 90 degrees in one time step, the angular velocity would be entirely transferred into a divergent field, which pressure projection would subsequently zero out. Consequently, the angular momentum is not conserved and is damped by the splitting error. Although the vorticity confinement[22] and spin particles[109] can recover some of the lost angular momentum, neither of the techniques is applicable to above simple cases. Later, Schechter and Bridson[122] proposed a new multistep predictor to alleviate the nonphysical dissipation of angular momentum.

From a historical perspective, we note that in the early stage of physically-based fluid animation, researchers mainly focused on fast and stable fluid solvers. But with the growing demands for more realistic animations and the increasing computational power, high accuracy and low numerical dissipation of the methods are getting more emphasis nowadays. Although the way to reduce the numerical viscosity for the advection term has been researched for more than one decade in computer graphics, the time splitting error is a brand new research area in fluid animations. Since the rotating motion of fluids is the source of a variety of interesting phenomena, we believe that reducing the dissipation of angular momentum caused by the splitting error will be an active and promising research direction.

5.3 Grid generation methods

Grid generation is a potential research area in physically-based fluid animation. The quality of the computational grid is a key factor for the Navier-Stoke solver to get an accurate solution. A good grid helps to enforce the correct boundary conditions and capture the small-scale details. As far as we know, in physically-based fluid animation, only two kinds of grids and their combinations[106] are adopted to discretize the Navier-Stokes equations: Cartesian grids and tetrahedral meshes. Both of the two have their shortcomings: Cartesian grids cannot fit arbitrary boundaries, introducing aliasing artifacts. Tetrahedral meshes involve large amounts of computational overhead. It is time consuming to create the mesh and interpolate values from the mesh. Although combining the Cartesian grids with the tetrahedral meshes in a multiple layer formation alleviates their respective problems, the local error in one layer can propagate throughout the whole computational domain.

The area of numerical grid generation is relatively young in practice, although its roots in mathematics are old. In CFD, grid generation is such an active research area that it has become an entity by itself; it is the subject of numerous special conference, as well as several books[124,125]. Since the grid generation in fluid animation is far from perfect right now, it is worthwhile to take a look at the development of grid generation in CFD and borrow some ideas. In CFD literature, a boundary fitted grid is constructed in the physical space which could be a non-uniform curvilinear grid or other non-uniform structured grids. Then a transformation is found to transform the physical space to a computational space. The non-uniform structured grid in physical space corresponds to a uniform rectangular grid in the computational space. Similarly, the governing partial differential equations are also transformed into the computational space using the chain rules of derivatives. Solve the transformed equations on the uniform rectangular grid. Then transform the solution back to the physical space. For simple cases such as stretched (compressed) grids, the transformation can be easily found by a few lines of mathematical derivations[14]. But if the boundary is quite complex (e.g. an airplane), other more sophisticated techniques: elliptic grid generation or zonal grids are suitable.

6 Conclusion

In this survey, we have introduced the basic concepts of physically-based fluid animation, the commonly used methods, different types of fluid phenomena and recent active research areas and directions. The purpose of this article is to provide readers with a rapid reference of the topic so that experienced users might easily identify the best or the most commonly used methods for a particular task and point out their strengths and deficiencies. Similarly, the beginners will hopefully find a gentle and up-to-date review of this field, get acquaint to the concepts of physically-based fluid animation without wasting time and effort in looking for the original references.

7 Acknowledgement

We thank Theodore Kim, Huamin Wang and Nils Thürey for granting us the permissions of using their beautiful images. We would like to thank everyone who spent time reading early versions of this paper, including the reviewers, for their comments and criticism. This work was supported by 863 National High Technology R&D Program of China (No. 2006AA01Z307).

1 Liu Y Q, Liu X H, Zhu H B, et al. Physically based fluid simulation in computer animation. Journal of Computer-Aided Design & Computer Graphics, 2005, 12.

2 Osher S, and Fedkiw R. Level-set methods and dynamic implicit surfaces. Springer-Verlag New York, Inc., 2003.

3 Pharr M, and Humphreys G. 2004. Physically based rendering: from theory to implementation. Morgan Kaufmann Pub-lishers Inc., San Francisco, CA, USA.

4 Reeves W T. Particle Systems - A technique for modeling a class of fuzzy objects[J]. Computer Graphics, 1983, 17(3): 359 376.

5 Monaghan J J. 1992. Smoothed particle hydrodynamics. Annual review of Astronomy and Astrophysics 30, 543C574.

6 Müller M, Charypar D, and Gross M. 2003. Particle-based fluid simulation for interactive applications. In SCA 03: Pro-ceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, Air-ela-Ville, Switzerland, Switzerland, 154C159.

7 Müller M, Keiser R, Nealen A, et al. 2004. Point based animation of elastic, plastic and melting objects. In SCA 04: Pro-ceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 141C151.

8 Premoze, S, Tasdizen T, Bigler J, et al. 2003. Particle-based simulation of fluids. Comput. Graph. Forum 22, 3, 401C410.

9 Stam J, and Fiume E. Depicting fire and other gaseous phenomena using diffusion processes. In SIGGRAPH 95: Proceed-ings of the 22nd annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1995), ACM Press, pp. 129C136.

10 Takeshita D, Ota S, Tamura m, et al. Particle-based visual simulation of explosive flames. Computer Graphics and Appli-cations, 2003. Proceedings. 11th Pacific Conference on (2003), 482C486.

11 Keiser R, Adams B, Gasser D, et al. A unified lagrangian approach to solid-fluid animation. Eurographics Symposium on Point-Based Graphics (2005), 125C133.

12 Cummins S, and Rudman M. An sph projection method. Journal of Computational Physics 152, 2 (1999), 584C607.

13 Becker M, and Teschner M. 2007. Weakly compressible sph for free surface flows. In Symposium on Computer Animation, Eurographics Association, M. Gleicher and D. Thalmann, Eds., 209C217.

14 Anderson J D, Jr. 'Computational Fluid Dynamics: The Basics with Applications, McGraw-Hill Inc, 1995.

15 Bridson R, and Muller-Fischer M. 2007. Fluid simulation: Siggraph 2007 course notes. In SIGGRAPH 07: ACM SIGGRAPH 2007 courses, ACM, New York, NY, USA, 1C81.

16 Harlow F H, and Welch J E. 1965. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. Phys. Fluids 8, 12, 2182C2189.

17 Stam J. Real-Time Fluid Dynamics for Games[A]. In: Proceedings of the Game Developer Conference[C], 2003.

18 Kass M, and Miller G. Rapid, Stable Fluid Dynamics for Computer Graphics[J]. Computer Graphics, 1990, 24(4): 49 57.

19 Bridson R. Shallow water discretization, Lecture notes Animation Physics, 2005.

20 Foster N, and Metaxas D. 1996. Realistic animation of liquids. Graph.

Models Image Process. 58, 5, 471C483.

21 Stam J. 1999. Stable fluids. In SIGGRAPH 99: Proceedings of the 26th annual conference on Computer graphics and in-teractive techniques, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 121C128.

22 Fedkiw R, Stam J, and Jensen H W. 2001. Visual simulation of smoke. In SIGGRAPH 01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM, New York, NY, USA, 15C22.

23 Kim B, Liu Y, Llamas I, et al. 2005. FlowFixer: Using BFECC for fluid simulation. In Eurographics Workshop on Natural Phenomena.

24 Song O-Y, Shin H, and Ko H-S. 2005. Stable but nondissipative water. ACM Trans. Graph. 24, 1, 81C97.

25 Foster N, and Fedkiw R. 2001. Practical animation of liquids. In SIGGRAPH 01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM, New York, NY, USA, 23C30.

26 Enright D, Marschner S, and Fedkiw R. 2002. Animation and rendering of complex water surfaces. In SIGGRAPH 02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, ACM, New York, NY, USA, 736C744.

27 Foster N, and Metaxas D. 1997. Modeling the motion of a hot, turbulent gas. In SIGGRAPH 97: Pro-ceedings of the 24th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Pub-lishing Co., New York, NY, USA, 181C188.

28 Nguyen D Q, Fedkiw R, and Jensen H W. 2002. Physically based modeling and animation of fire. ACM Trans. Graph. 21, 3, 721C728.

29 Carlson M, Mucha P J, Van Horn R, et al. 2002. Melting and flowing. In SCA 02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation, ACM, New York, NY, USA, 167C174.

30 Goktekin T G, Bargteil A W, and Obrien J F. 2004. A method for animating viscoelastic fluids. In SIGGRAPH 04: ACM SIGGRAPH 2004 Papers, ACM, New York, NY, USA, 463C468.

31 Hong J -M, and Kim C -H. 2005. Discontinuous fluids. ACM Trans. Graph. 24, 3, 915C920.

32 Wang H, Mucha P J, and Turk G. 2005. Water drops on surfaces. In SIGGRAPH 05: ACM SIGGRAPH 2005 Papers, ACM, New York, NY, USA, 921C929.

33 Kim B, Liu Y, Llamas I, et al. 2007. Simulation of bubbles in foam with the volume control method. ACM Trans. Graph. 26, 3, 98.

34 Li W, Wei X M, and Kaufman A. Implementing lattice boltzmann computation on graphics hardware[J]. The Visual Com-puter, 2003, 19: 444 456.

35 Chen S Y, and Doolen G D, Lattice boltzmann method for fluid flows. Annual Review of Fluid Mechanics, Vol. 30, No. 1. (1998), pp. 329-364.

36 Bhatnagar P L, Gross E P, and Krook M. A model for collision processes in gases. Phys. Rev., 94:511C525, 1954.

37 Wei X M, Zhao Y, Fan Z, et al. Natural phenomena: blowing in the wind. Proc. of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer animation, pages 75C85, July 2003.

38 Losasso F, Shinar T, Selle A, et al. 2006. Multiple interacting liquids. ACM Trans. Graph. (SIGGRAPH Proc.) 25, 3, 812C819.

39 Kim J, Cha D, Chang B, et al. 2006. Practical animation of turbulent splashing water. In Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim., 335C344.

40 Thuerey N, Sadlo F, Schirm S, et al. 2007. Real-time simulations of bubbles and foam within a shallow water framework. In Proc. of the 2007 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim., 191C198.

41 Hong J M, Lee H Y, Yoon J C, et al. Bubble Alive. To appear in the ACM SIGGRAPH conference proceedings, 2008.

42 Takahashi T, Fujii H, Kunimatsu A, et al. Realistic animation of fluid with splash and foam[J], Computer Graphics Forum, 2003, 22(3): 391 400.

43 Desbrun M, and Cani M P. Smoothed particles: A new paradigm for animating highly deformable bodies. In Computer Animation and Simulation 96 (Proceedings of EG Workshop on Animation and Simulation), pages 61C76. Springer-Verlag, Aug 1996.

44 Chentanez N, Feldman B E, Labelle F, et al. 2007. Liquid simulation on lattice-based tetrahedral meshes. In SCA 07: Pro-ceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurgraphics Association, Aire-la-Ville, Switzerland, Switzerland, 219C228.

45 Klingner B M, Feldman B E, Chentanez N, et al. 2006. Fluid animation with dynamic meshes. ACM Trans. Graph. 25, 3, 820C825.

46 Batty C, Bertails F, and Bridson R. 2007. A fast variational framework for accurate solid-fluid coupling. In SIGGRAPH 07: ACM SIGGRAPH 2007 papers, ACM, New York, NY, USA, 100.

47 Losasso F, Gibou F, and Fedkiw R. 2004. Simulating water and smoke with an octree data structure. In SIGGRAPH 04: ACM SIGGRAPH 2004 Papers, ACM, New York, NY, USA, 457C462.

48 Irving G, Guendelman E, Losasso F, et al. 2006. Efficient simulation of large bodies of water by coupling two and three dimensional techniques. ACM Trans. Graph. 25, 3, 805C811.

49 Harris M J, Baxter W V III, Scheuermann T, et al. Simulation of cloud dynamics on graphics hardware[A]. In: Proceedings of Graphics Hardware, San Diego,. 2003.92 101.

50 Harris M J. Real-time cloud Simulation and rendering[D]. PhD thesis of The University of North Carolina at Chapel Hill, 2003.

51 Dobashi Y, Kaneda K, Okita T, et al, efficient method for realistic animation of clouds. In SIGGRAPH 2000 Conference Proceedings, Annual Conference Series, pages 19C28, July 2000.

52 Miyazaki R, Dobashi Y, and Nishita T. 2002. Simulation of cumuliform clouds based on computational fluid dynamics. Proc. EUROGRAPHICS 2002 Short Presentation, 405C410.

53 Steinhoff J, and Underhill D. Modification of the euler equations for vorticity confinement: Application to the computation of interacting vortex rings. Physics of Fluids, 6(8):2738C2744, 1994.

54 Lamorlette A, and Foster N. Structural modeling of flames for a production environment [J]. ACM Transactions on Graphics, 2002, 21(3): 729 735.

55 Yngve G D, Obrien J F, and Hodgins J K. 2000. Animating explosions. In Proceedings of SIGGRAPH 2000, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 29C36.

56 Neff M, and Fiume E. 1999. A visual model for blast waves and fracture. In Proceedings of Graphics Interface 1999, 193C202.

57 Obrien J F, and Hodgins J K. 1999. Graphical modeling and animation of brittle fracture. In Proceedings of SIGGRAPH 1999, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 137C146.

58 Rasmussen N, Nguyen D Q, Geiger W, et al. Smoke simulation for large scale phenomena [J]. ACM Trans-actions on Graphics, 2003, 22 (3): 703 707.

59 Desbrun M, and Cani M -P. Animating soft substances with implicit surfaces. In Computer Graphics Proceedings (1995), ACM SIGGRAPH, pp. 287C290.

60 Clavet S, Beaudoin P, and Poulin P. 2005. Particle based viscoelastic fluid simulation. In Proc. Sym-posium on Computer Animation, 219C228.

61 Bargteil A W, Wojtan C, Hodgins J K, et al. 2007. A finite element method for animating large viscoplastic flow. ACM Trans. Graph. 26, 3, 16:1C16:8.

62 Wojtan C, and Turk, G. Fast viscoelastic behavior with thin features. To appear in the ACM SIGGRAPH conference pro-ceedings, 2008.

63 Miller G, and Pearce A. 1989. Globular dynamics: a connected particle system for animating viscous fluids. In Comput. & Graphics, vol. 13, 305C309.

64 Luciani A, Habibi A, and Manzotti E. 1995. A multiscale physical model of granular materials. In Graphics Interface, 136C146.

65 Zhu Y, and Bridson R. 2005. Animating sand as a fluid. ACM Trans. Graph. 24, 3, 965C972.

66 Harlow F H. 1963. The particle-in-cell method for numerical solution of problems in fluid dynamics. In Ex-perimental arithmetic, high-speed computations and mathematics.

67 Brackbill J U, and Ruppel H M. 1986. FLIP: a method for adaptively zoned, particle-in-cell calculuations of fluid flows in two dimensions. J. Comp. Phys. 65, 314C343.

68 Fedkiw R, Aslam T, Merriman B, et al. 1999. A non-oscillatory eulerian approach to inter-faces in multimaterial flows (the ghost fluid method). Journal of Computational Physics 152, 457C492.

69 Hong J -M, and Kim C -H. 2003. Animation of bubbles in liquid. Comput. Graph. Forum (Eurographics Proc.) 22, 3, 253C262.

70 Hirt C W, and Nichols B D, Volume of Fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys., 39, 201C255 (1981).

71 Tryggvason G, Bunner B, Esmaeeli A, et al. A front tracking method for the Computations of multiphase flow, J. Comput. Phys., 169, 708-759 (2001).

72 Greenwood S T, and House D H. 2004. Better with bubbles: enhancing the visual realism of simulated fluid. In Proc. of the 2004 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim., 287C296.

73 Müller M, Solenthaler B, Keiser R, et al. 2005. Particle-based fluid-fluid interaction. In Proc. of the 2005 ACM SIGGRAPH/Eurographics Symp. on Comput. Anim., 237C244.

74 Kim T, and Carlson M. 2007. A simple boiling module. In SCA 07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, Aire-la-Ville, Switzerland, Swit-zerland, 27C34.

75 Thürey N, Rüde U. 2004. Free surface lattice-boltzmann fluid simulations with and without level sets. In Workshop on Vision, Modeling, and Visualization.

76 Pohl T, Deserno F, Thürey N, et al. 2004. Performance evaluation of parallel large-scale lattice boltzmann applications on three supercomputing architectures. In Super-computing, 2004. Proceedings of the ACM/IEEE SC2004 Conference.

77 Takahashi T, Heihachi U, and Kunimatsu A. 2002. The simulation of fluid-rigid body interaction. In Proc. SIGGRAPH Sketches & applications.

78 Génevaux O, Habibi A, and Dischler J-M. 2003. Simulating fluid-solid interaction. In Graphics Interface, 31C38.

79 Carlson M, Mucha P J, and Turk G. 2004. Rigid fluid: animating the interplay between rigid bodies and fluid. ACM Trans. Graph. (Proc. SIGGRAPH) 23, 377C384.

80 Guendelman E, Selle A, Losasso F, et al. 2005. Coupling water and smoke to thin deformable and rigid shells. ACM Trans. Graph. (Proc. SIGGRAPH) 24, 3, 973C981.

81 Peskin C S. 2002. The immersed boundary method. Acta Numerica 11, 479C517.

82 Chentanez N, Goktekin T G, Feldman B E, et al. 2006. Simultaneous coupling of fluids and deformable bodies. In ACM-EG Proc. Symposium on Computer Animation, 83C89.

83 Lenaerts T, Adams B, Dutré P. Porous flow in particle-based fluid simulations. To appear in the ACM SIGGRAPH con-ference proceedings, 2008.

84 Blinn J F. 1982. A generalization of algebraic surface drawing. ACM Trans. Graph. 1, 3, 235C256.

85 Adams B, Pauly M, Keiser R, et al. 2007. Adaptively sampled particle fluids. In SIGGRAPH 07: ACM SIGGRAPH 2007 papers, ACM, New York, NY, USA, 48.

86 Müller M, Schirm S, and Duthaler S. 2007. Screen space meshes. In SCA 07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, Aire-la-Ville, Switzer-land, Switzerland, 9C15.

87 Unverdi S, and Tryggvason G, A front tracking method for viscous, incompressible, multifluid flows, J. Comput. Phys., 100 25 (1992).

88 Torres D J, and Brackbill J U, The point-set method: front-tracking without connectivity, J. Comput. Phys., 165, 620-644 (2000).

89 Shin S, and Juric D, Modeling three-dimensional multiphase flow using a level contour reconstruction method for front tracking without connectivity, J. Comput. Phys., 180, 427C470 (2002).

90 Kunimatsu A, Watanabe Y, Fujii H, et al, Fast simulation and rendering tech-niques for fluid objects, Computer Graphics Forum (Eurographics 2001 Proc.), 20(3), 357C367 (2001).

91 Lorensen W E, and Cline H E. 1987. Marching cubes: A high resolution 3d surface construction algorithm. SIGGRAPH Comput. Graph. 21, 4, 163C169.

92 Mihalef V, Metaxas D, and Sussman M, Textured liquids based on the marker level set. Eurographics 2007, Volume 26(2007), Number 3.

93 Bargteil A W, Goktekin T G, Obrien J F, and Strain J A. 2006. A semi-Lagrangian contouring method for fluid simulation. ACM Trans. Graph. 25, 1, 19C38.

94 Foster N, and Metaxas D. Controlling fluid animation. In Proc. of CGI (1997).

95 Feldman B E, Obrien J F, and Arikan O. Animating suspended particle explosions. In Proceedings of ACM SIGGRAPH 2003 (Aug. 2003), pp. 708C715.

96 Rasmussen N, Enright D, Nguyen D, et al. Directable photorealistic liquids. In Proc. of Symposium on computer Animation (2004).

97 Treuille A, Mcnamara A, Popovic Z, et al. Keyframe control of smoke simulations. ACM Trans. Graph. 22, 3 (2003), 716C723.

98 Mcnamara A, Treuille A, Popovic Z, et al. Fluid control using the adjoint method. ACM Trans. Graph. 23, 3 (2004), 449C456.

99 Pighin F, Cohen J M, and Shah M. Modeling and editing flows using advected radial basis functions. In SCA 04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation (2004), ACM Press, pp. 223C232.

100 Fattal R, and Lischinski D. Target-driven smoke animation. ACM Trans. Graph. 23, 3 (2004), 441C448.

101 Hong J -M, and Kim C.-H. Controlling fluid animation with geometric potential: Research articles. Comput. Animat. Virtual Worlds 15, 3-4 (2004), 147C157.

102 Shi L, and Yu Y. Controllable smoke animation with guiding objects. ACM Trans. Graph. 24, 1 (2005).

103 Shi L, and Yu Y. Taming liquids for rapidly changing targets. In Proc. of Symposium on Computer Animation (2005).

104 Thürey N, Keiser R, Pauly M, et al. 2006. Detail-preserving fluid control. In SCA 06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 7C12.

105 Feldman B E, Obrien J F, and Klingner B M. 2005. Animating gases with hybrid meshes. In SIGGRAPH 05: ACM SIGGRAPH 2005 Papers, ACM, New York, NY, USA, 904C909.

106 Tan J, Yang X B, Zhao X, et al. Fluid animation with multi-layer grids. To appear in Proc. Symposium on computer Ani-mation 2008.

107 Thürey N, Rüde U, and Stamminger M. 2006. Animation of open wa-

ter phenomena with coupled shallow water and free surface simulations. In SCA 06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer ani-mation, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 157C164.

108 Kang B, Jang Y, and Ihm I. 2007. Animation of chemically reactive fluids using a hybrid simulation method. In SCA 07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, Euro-graphics Association, Aire-la-Ville, Switzerland, Switzerland, 199C208.

109 Selle A, Rasmussen N, and Fedkiw R. 2005. A vortex particle method for smoke, water and explosions. ACM Trans. Graph. 24, 3, 910C914.

110 Losasso F, Talton J, Kwatra N, et al. 2008. Two-way coupled SPH and particle level set fluid simulation. IEEE Trans. on Vis. and Comput. Graph. (In Press).

111 Treuille A, Lewis A, and Popovic Z. Model reduction for real-time fluids. ACM Trans. Graph. 25, 3 (2006).

112 Gupta M, and Narasimhan S G. 2007. Legendre fluids: A unified framework for analytic reduced space modeling and ren-dering of participating media. In Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2007).

113 Boltz J, Farmer I, Grinspun E, and Schr Ooder P. 2003. Sparse matrix solvers on the gpu: conjugate gradients and multigrid. ACM Trans. Graph. 22, 3, 917C924.

114 Liu Y Q, Liu X H, and Wu E H. Real-time 3d fluid simulation on gpu with complex obstacles[A]. In: Proceedings of Pacific Graphics 2004, Seoul, 2004.247 256.

115 Wu E H, Liu Y Q, and Liu X H. An improved study of real-time fluid simulation on gpu[J]. Journal of Computer Animation and Virtual World (CASA2004), 2004, 15(3-4): 139 146.

116 Tölke J. Implementation of a lattice boltzmann kernel using the compute unified device architecture developed by nVIDIA. To appear in Computing and Viualization in Science

117 Crane K, Llamas I, and Tariq S. Real-time simulation and rendering of 3D fluids. GPU Gem 3, Chaper 30, Nvidia 2007.

118 Lin N. Special effect with Geforce 8 series hardware. Game Developer Conference Shanghai, 2007.

119 Geiss R. Generating complex procedural terrains using the gpu. GPU Gem 3, Chaper 1, Nvidia 2007.

120 Thürey N, Rüde U. Stable free surface flows with the lattice Boltzmann method on adaptively coarsened grids. To appear in Computing and Visualization in Science, 2008

121 Kim T, Thürey N, James D, et al. Wavelet turbulence for fluid simulation, To appear in SIGGRAPH 2008

122 Schechter H. and Bridson R. Evolving sub-grid turbulence for smoke animation, Symposium on Computer Animation 2008.

123 Molemaker J, Cohen M J, Patel S, et al. Low viscosity flow simulations for animation, Symposium on Computer Animation 2008.

124 Thompson and Joe F. (ed.): Numerical grid generation, North-Holland, New York, 1982

125 Thompson, Joe F, Z V A Warsi, et al. Numerical grid generation: foundations and applications, North-Holland, New York, 1985