Adaptive Grid Refinement Using View-Dependent Octree for Grid-Based Smoke Simulation

Rinchai Bunlutangtum and Pizzanu Kanongchaiyos

Chulalongkorn University, Bangkok, Thailand
rinchai.b@student.chula.ac.th, pizzanu@cp.eng.chula.ac.th

Abstract. Computational cost is one of the major problems in animating smoke. Recently, adaptive grid refinement using octree structure has been proposed, which is a successful method for reducing the computational cost of a detail-preserving fluid simulation. Although octree grid is optimized for details, viewing is not addressed. Smoke distant from the viewing screen which usually has less visual attention and is unnecessary for high-detail simulation can be optimized for speed. However, applying such view-dependent optimization to the octree grid directly may cause animation artifacts and loss in natural fluid behaviours. This paper, we present a method for view-dependent adaptive grid refinement, extending the traditional octree grid by considering the viewing frustum, as well as variation in fluid quantities as criteria for grid refinement. In our method, refinement conditions with adaptive thresholds are proposed to optimize the grid for both view and details. Additionally, our method preserves visual details and fluid behaviours which allows high-detail smoke animations in relatively less amount of computational cost consumption, especially when applied for large scale simulations.

Keywords: Simulation of Natural Environments, Physics-based Animation, Fluid Dynamics, Adaptive Refinement.

1 Introduction

Physically-based fluid simulation is a widely used technique for animating smoke, fire and other fluid phenomena. It generates physically realistic results and stunning effects which are impossible for the artist to animate manually frame by frame. Unfortunately, physically-based simulation usually comes with high computational cost as a trade-off. For graphics and realism, the main focus is on generating plausible visual effects rather than accuracy. Since computational resources especially in terms of time and memory usage are limited, a challenging topic is how to minimize the computational cost, while still being able to obtain as highly detailed animations as possible.

Grid-based fluid simulation is a widespread used method for physically-based smoke animations. [9,17,7,18] use fixed uniform grid for smoke animations. Since smoke animations are becoming more and more demand in the special effects

J.M. Allbeck and P. Faloutsos (Eds.): MIG 2011, LNCS 7060, pp. 204–215, 2011.

industry, animating on fixed uniform grid in a larger domain, or refinement for higher detail is not scalable, because of its high computational cost consumption. To address this, several adaptive grid refinement techniques were introduced to optimize the simulation. [13] and [16] replace the traditional fixed uniform grid with an adaptive non-uniform grid, using an octree structure. Adaptive grid refinement using an octree structure has been successful in optimizing the simulation. The grid is subdivided only in some specific areas that require higher detail and are merged to save computational cost when details are no longer necessary. Later,[1] implements the octree grid on graphics hardware, which utilizes high performance parallel computing. These adaptive grid techniques allow the capturing of small visual details while takes lower computational cost compared to the earlier fixed uniform grid.

Small-scale details can be neglected in some regions such as hidden or distant smoke as they usually have less visual attention, [2] proposed a method that is optimized for view by using a view-dependent grid. Instead of being constructed on Cartesian coordinates as usual, the grids are constructed on the transformed polar coordinate that is most fit for view. With a view-dependent grid, fluid details gradually decrease proportionally as the distance from the camera increases, thus providing constant screen-space detail across the simulation domain. However, unlike octree grid, the view-dependent grid is subdivided uniformly on transformed coordinates, thus, grid size is fixed and not adaptive for detail optimization.

The octree grid is optimized for details whereas the view-dependent grid is optimized for view. Optimizing the grid for both viewing and details by applying view-dependent optimization to octree grid directly is straightforward but unfortunately, might cause undesired simulation artifacts or ineffective optimized grid. Thus a proper handling method along with user adjustable parameters is preferred. This paper, we present a method for view-dependent adaptive grid refinement on octree grid that minimizes unnecessary computational cost while still preserving any small visual details and their natural behaviours. In our method, we propose adaptive thresholds that are view-dependent and proportional to the distance ratio with respect to the viewing frustum. These thresholds weight the subdivision and merging conditions in order to preserve details that are usually lost when optimizing the grid. Our method optimizes the grid for both viewing and details. Furthermore, it preserves small-scale fluid details while consuming a lower amount of computational cost compared to the simulation using either a traditional fixed uniform grid or an octree grid without view-dependent optimization.

2 Related Work

Grid-based fluid simulation was first introduced in computer graphics by [9] but because their model uses an explicit integration scheme, their simulations are only stable if the time step is chosen small enough, therefore, simulation is relatively slow. [17] proposed an unconditionally stable simulation by using

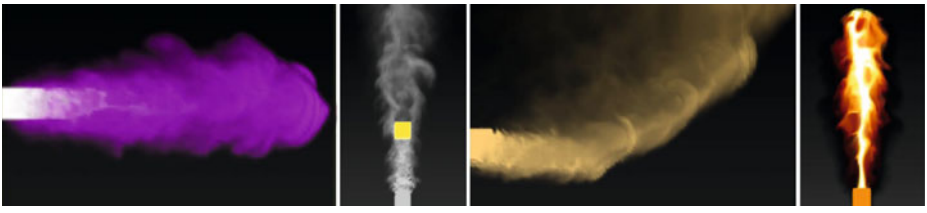

Fig. 1. Four different scenes of smoke simulation using octree grid with our view-dependent adaptive grid refinement. With our method, not only details are preserved but simulation also consumes lower computational cost with less of artefacts.

semi-Lagrangian advection scheme with implicit solvers. However, numerical dissipation was severe in this method. [7] introduced a vorticity confinement term to model the small scale rolling features characteristic of smoke to compensate the numerical dissipation caused by the implicit model. The methods have been extended to other fluid phenomena such as fire [14], explosion [8], viscoelastic materials [10] and bubbles [19]. These previous works can be categorized as fixed uniform grid as the grid are subdivided uniformly and their position are fixed entire simulation domain.

To animate fluid in a larger domain or to obtain higher details, [4,3] introduced adaptive mesh refinement (AMR) for compressible flows while [15] presented an adaptive mesh method using an octree. In computer graphics, the octree data structure has been proposed for adaptive grid refinement by [13] and asymmetric octree by [16], which results in detail optimization while [2] proposed a method that is optimized for view by using view-dependent grid, decreasing fluid details as distance from the camera increases, thus providing constant screen-space detail across the simulation. However, contrary to the adaptive grid using octree, view-dependent grid is subdivided uniformly, thus, lacking detail optimization.

Another recent grid refinement technique is Adaptive grid using tetrahedral meshes [12,5,6]. Tetrahedral meshes are easily to conform to complex boundaries and their size can be adjusted to optimize the simulation. However, simulation is more complicated for free surfaces and moving obstacles. Simulation also needs a specific scheme to prevent volume loss or artificial damping.

One of the main problems with octree grids is their dynamic and irregular structure, which is contrary to the design of graphics hardware. [1] present a problem decomposition for parallel processing that takes advantage of the graphics hardware while reducing expensive hierarchy traversals on non-uniform and adaptive octree grids.

3 Proposed Method

The idea to optimize the grid for both viewing and details is that cells should be divided for higher resolution if they are too close to the camera or their neighbouring cells are relatively different to each others. On the other hand,

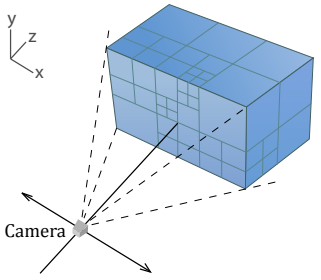

Fig. 2. Illustration of view-dependent octree grid

cells should be merged for lower resolution if their distances to the camera are too far for any fine details to be visualized or the variation of fluid quantities among neighbouring cell are no longer significant. However, not only does the distance from camera affect the grid refinement, but the dimensions of viewing frustum and its perspective are also important parameters that should be properly weighted with the refinement conditions in order to achieve an effective detail-preserved optimization.

Basically, refinement conditions are used to determine whether to perform subdivision or merging by comparing variation in fluid quantities with arbitrary constants, called refinement thresholds. To further optimize the grid for both viewing and detail, we propose a *view-dependent weighting factor* as a factor to weight these refinement thresholds. Thus, in this manner, the thresholds are adaptive and directly proportional to the viewing frustum. Thresholds with view-dependent weighting are called *adaptive thresholds* and the refinement with conditions compose of these thresholds is called *view-dependent adaptive grid refinement*. Detail of our method is described in the following subsections.

3.1 Octree Structure

We have constructed an adaptive grid with octree structure as illustrated in figure 2. A hierarchical grid with subdivision of 2^3 is employed to ensure transversal smoothness, whereas higher hierarchical subdivision e.g. 4^3 may cause visual artifact due to rapid changing in grid size during subdivision and merging. The velocity vectors are stored at the center of each cell as stated in [17], differing from the traditional MAC grid [11] which stores velocity at the cell faces, because memory usage for storing velocity at cell nodes is less than that for storing at cell faces, and it is also straightforward to implement. Pressure, density, temperature and other fluid quantities are stored at cell nodes as well (Figure 3).

3.2 Measuring Fluid Variation

[16] determine fluid variation by measuring variation in density. However, we found that measuring variation in velocity also yields a relevant result which can

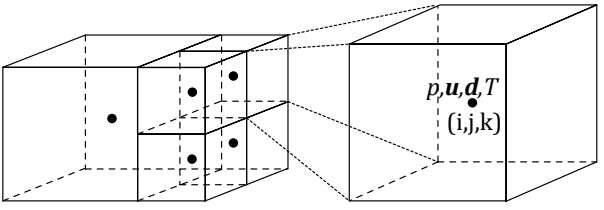

Fig. 3. Structure of octree grid where fluid quantities are stored at cell nodes

be used for inferring variation of other quantities as well. This is because density is always advected (carried) by velocity fields; thus, measuring velocity itself is reasonable. Moreover, by measuring variation of velocity, Laplacian terms shown in Equation 1 do not need to be computed, as they can be directly obtained from the diffusion step during solving Navier-Stokes equations (Section 4). Our equation for measuring variation in velocity is as follows:

$$C(x, y, z) = max(|\nabla_x^2 u|, |\nabla_y^2 v|, |\nabla_z^2 w|) \tag{1}$$

Equation 1 measures the variation of velocity, where u, v, w denote scalar components of velocity vector: $\mathbf{u} = (u, v, w)$. ∇_x^2 is a Laplacian on x direction: $\nabla_x^2 u = \partial^2 u / \partial x^2$. ∇_y^2 and ∇_z^2 are defined likewise.

3.3 View-Dependent Weighting Factor

Any viewing parameters that affect the grid refinement are gathered and rewritten as a mathematical relation called *view-dependent weighting factor*, as a factor to weight the refinement thresholds.

Suppose that we are observing an arbitrary object within a rectilinear perspective view. If we move the camera constant speed away from the object then the observed size of that object is decreased hyperbolically. In addition, if we increase object's actual size and move the camera away accordingly then the observed size should be preserved as constant. We have inherited this relation for constructing a view-dependent weighting factor as the grid's resolution over distance should be coarsened away in a proportion that the observed grid resolution still remains constant.

Let Ψ represent a view-dependent weighting factor and r and R represent the Euclidean distances measured from camera to arbitrary cell's node and from camera to front clipping plane respectively (see Figure 4). Ψ should then be proportional to r. Moreover, Ψ should be affected by dimensions of the viewing frustum and its perspective. For instance, Ψ should be greater if r is large with respect to R or the viewing frustum has high perspective. The view-dependent weighting factor is:

$$\Psi = \alpha \left(\frac{r - R}{R} \right) \tag{2}$$

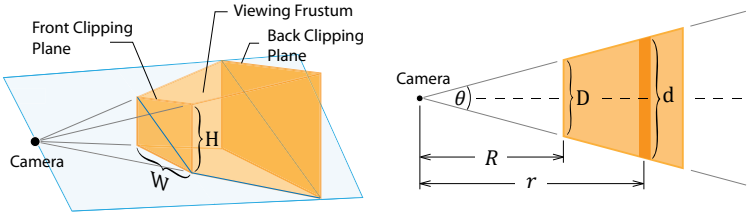

Fig. 4. Viewing frustum shown in three-dimensions (left) and its diagonal cross-section (right), where shading areas are the visible volume

Basically, Ψ is factored by distance ratio: r/R. However, we want the Ψ to be affected only further than the front clipping plane, not the origin; thus, we offset the ratio by R, which yields a distance ratio: $(r - R)/R$. α is a camera's perspective ratio defined as follow.

$$\alpha = \frac{D}{2R} = \tan(\frac{\theta}{2}) \tag{3}$$

Equation 3 refers to the viewing frustum (figure 4). D is a diagonal length of front clipping plane: $D = \sqrt{H^2 + W^2}$, where H and W are height and weight of front clipping plane respectively. R is a Euclidean distance from camera to the center of front clipping plane, θ is the camera's field of view (abbreviated as FOV, also called angle of view) measured diagonally.

3.4 Adaptive Thresholds

Grid refinement is performed by means of refinement conditions, i.e., merging condition and subdivision condition. In general, refinement conditions comparing fluid variation with constant thresholds are used. To further perform the view-dependent adaptive grid refinement, we propose refinement conditions incorporated with *adaptive thresholds* as stated below.

$$T^* = T(\tau\Psi + 1)\phi = T\left(\tau\alpha\left(\frac{r - R}{R}\right) + 1\right)\left(\frac{res_{grid}}{res_{image}}\right) \tag{4}$$

T^* is an adaptive threshold for arbitrary viewing frustum at distance r and FOV θ, where Ψ is the view-dependent weighting factor and τ is a view-dependent coefficient, specifying the weight that the view should affect the grid refinement. T is a refinement threshold specified on a front clipping plane of a viewing frustum at arbitrary FOV θ, obtained by $T = \alpha T_{90^o}$, where T_{90^o} is a refinement threshold at $FOV = 90^o$ and α is a camera's perspective ratio.

Altering the FOV might affect the grid resolution. For instance, narrow FOV (zooming) makes small details get larger hence, grid resolution must be coarsened to allow finer details. To preserve overall details whether FOV is changed, we factor a refinement threshold with a view-dependent weighting term: $(\tau\Psi + 1)$.

Image resolution, which is simply a pixel count per unit length of screen, is another factor that should be considered. If we lower the image resolution but keep everything else fixed, the grid can be allowed to be coarser (since details are negligible in a low-resolution image). To address this, we define a *resolution ratio* (ϕ) as a ratio of grid resolution over image resolution: $\phi = res_{grid}/res_{image}$. For example, a resolution ratio of two means the grid resolution is twice finer than the image resolution. Since grid resolution is no need to be greater than image resolution thus T^* should be higher to allow coarser grid.

3.5 Refinement Conditions

We have constructed refinement conditions for view-dependent adaptive grid refinement, defined as follows.

$$C(x, y, z) > T_s^* = T_s(\tau\Psi + 1)\phi \tag{5}$$

$$C(x, y, z) < T_m^* = T_m(\tau\Psi + 1)\phi \tag{6}$$

Equation 5 and 6 are refinement conditions for subdivision and merging respectively. $C(x, y, z)$ is the measuring of fluid variation while on the right hand side of these equations are adaptive thresholds inherited from equation 4 where T_s^* and T_m^* are for subdivision and merging respectively.

Refinement conditions categorize each cell in to one of these stages i.e., subdivision, merging and idle stage. If equation 5 is satisfied then the subdivision stage is assigned. If equation 6 is satisfied then the merging stage is assigned. Otherwise, if neither equation 5 nor equation 6 is satisfied then the idle stage is assigned therefore, no refinement is performed.

In Figure 5, we have illustrated the relation between variation $C(x, y, z)$ and distance r with several parameter adjustments. (a) and (b) show the comparison between using cameras with different FOVs. The perspective of camera, which is specified by FOV, corresponds to the steepness of slope which indicates the tendency of merging and division. A result of using a wider FOV is a greater view-dependent level of detail. As a consequence, grid resolution tends to be coarsened faster as the distance from the camera increases. In addition to a wider FOV, T_s^* and T_m^* are shifted upward to allow coarser grid resolution as details usually get smaller in wide-angle view. The result of using narrow FOV is inversion. Furthermore, if an infinitesimal value of FOV ($\alpha \approx 0$) is applied then the refinement results as no view-dependent optimization since the view-dependent terms are neglected and left only the constant terms T_s and T_m. The view-dependent terms are also neglected when the camera is placed at infinity ($R \approx \infty$), or the view-dependent coefficient is set to zero ($\tau = 0$).

The sensitivity of stage transitions (between subdivision, merging and idle) correspond to the differential value of the adaptive threshold T_s^* and T_m^*, which can be adjusted by altering T_s and T_m. In figure 5, (c) T_s^* is close to T_m^* which results in a narrow idle stage area; hence, it is sensitive to stage transition, whereas (d) T_s^* is much different to T_m^* therefore, it is tolerant to stage transition. As stage transitions consume computational cost, assigning inappropriate thresholds may

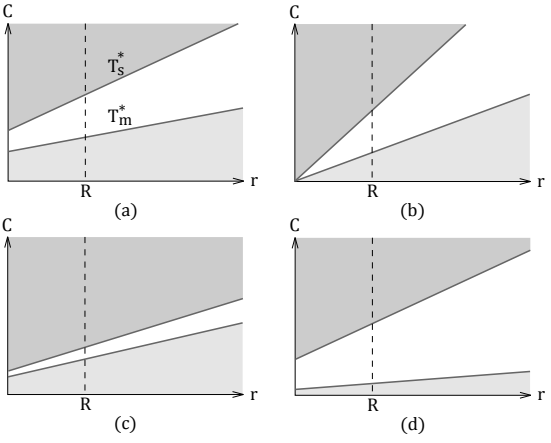

Fig. 5. Graphical plots of each stages i.e., subdivision (dark grey area), merging (light grey area), idle (white area) which are separated by T_s^* (upper lines) and T_m^* (lower lines) over variation $C(x, y, z)$ and distance r using narrow FOV camera (a), wide FOV camera (b), small (c) and large (d) of difference between T_s and T_m.

result in an excessive overhead due to frequent merging and subdivision or otherwise result in an inefficient optimization; thus, thresholds must be carefully selected. Moreover, to prevent excessive refinement overhead, grid size should be gradually adapted at each time step, which can be done by performing only one operation (i.e., subdivision, merging, or idle) per cell each iteration. This reduces recursive subdivision and merging overhead and also prevents rapid changing in cell size that might cause animation artifacts.

4 Smoke Simulation

After applying the refinement method, the grid is well optimized and ready for the simulation step. We solve the Navier Stokes equations that describe fluid flow and its behaviours with a stable scheme proposed by [17] and [7], then we follow a discretization scheme proposed by [13] to solve the Poisson equations on the octree grid, which is adaptive and has non-uniform structure.

5 Results and Discussion

In order to evaluate our refinement method, we have constructed and compared results of three different simulations i.e., an octree grid with view-dependent optimization, an octree grid without view-dependent optimization, and a fixed uniform grid. Results are compared by measuring both computational cost and animation quality. Computational cost is compared by measuring number of

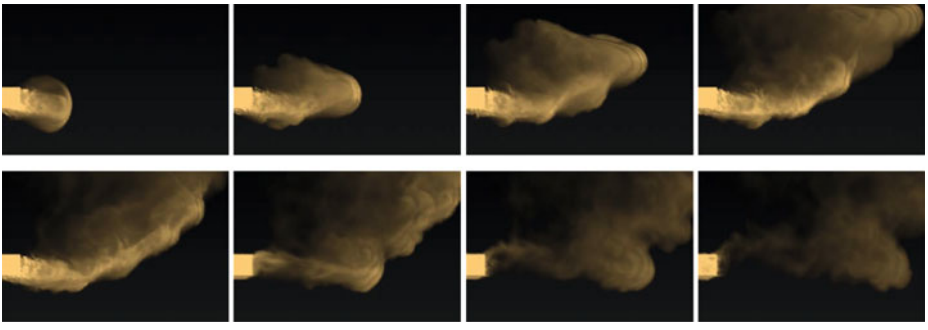

Fig. 6. Smoke animation using octree grid with view-dependent adaptive grid refinement. Simulated with view-dependent coefficient $\tau = 1$, initial grid resolution of $32 \times 20 \times 12$ and maximum grid resolution of $128 \times 80 \times 48$.

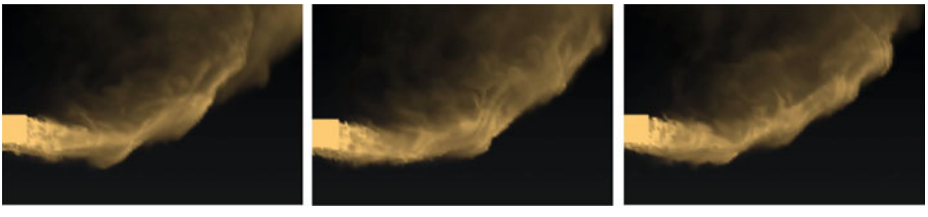

Fig. 7. Snapshots comparing visual results of using fixed uniform grid (left), octree grid without view-dependent refinement (center) and octree grid with view-dependent refinement (right). Detail loss by our method is unnoticeable at $\tau = 1$.

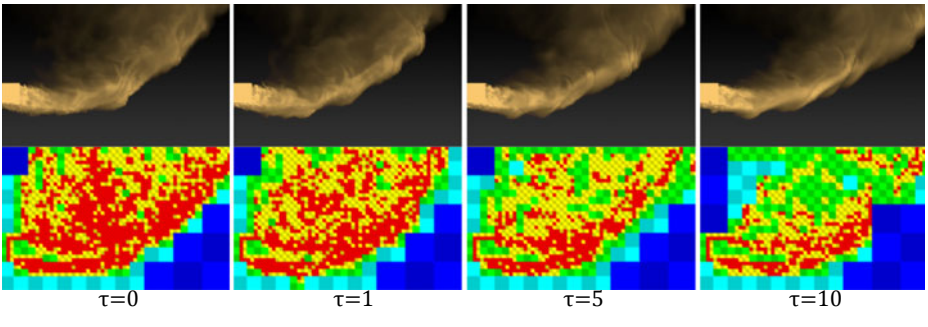

Fig. 8. Snapshots comparing the effect of view-dependent refinement by varying view-dependent coefficient τ. Visual results (above) showing preserved details and their corresponding cut away view (below) showing current grid size in range of color i.e., red is fine grid, blue is coarse grid. In this simulation, $\tau = 1$ is an optimal weighting value between speed and detail, $\tau = 5$ is a maximum practical value that detail loss still unnoticeable and with $\tau = 10$, simulation suffers from visible detail loss i.e., smooth flow and no small swirl effect.

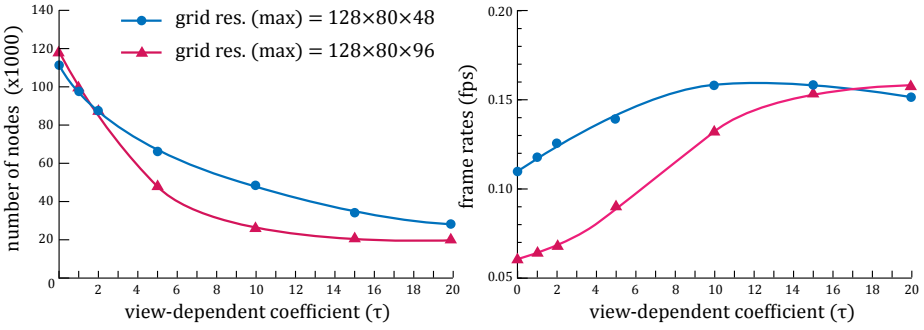

Fig. 9. Right: Number of cell nodes on various view-dependent coefficient. Left: The relation between frame rates and view-dependent coefficients. Incrementing the view-dependent coefficient decreases the number of nodes, which speed up the simulation.

frames per second while animation quality is compared using visual results. All experiments reported in this paper were performed on a machine with dual core CPU 2.40 GHz and 2 GB of RAM.

We have constructed a simulation domain as illustrated in figure 2 where the camera is placed on the z-axis facing toward the domain, and its visual angle is fixed at 90 degrees. We have applied two sets of grid resolution in our experiment. In the first set, an initial resolution of $32 \times 20 \times 12$ is applied to the octree grid with a specified maximum resolution of 128 x 80 x 48 while for the fixed uniform grid, the resolution is always fixed at 128 x 80 x 48. In the second set, we double only the domain depth in order to evaluate the efficiency of view-dependent optimization on different domain sizes. According to the results shown in Figure 9, simulation gains speed by applying our method. Moreover, the method efficiency is largely depends on fluid-to-camera distant (r), larger domains which usually contains a lot of distant smoke are more effective for optimization; hence, speed-up is significant in larger domains.

We have demonstrated the effect of the view-dependent adaptive grid refinement by varying only the view-dependent coefficient τ. Visual results are shown in Figure 8 and their corresponding computational cost are shown in Figure 9 and Table 1. The results show that applying a greater view-dependent coefficient reduces the number of cells in the domain which result in lower computational cost consumption. The simulation achieves a higher frame rate but detail loss is relatively greater. On the other hand, if a lower view-dependent coefficient is applied, more details are preserved but with higher computation cost as a trade-off. Furthermore, if the view-dependent coefficient is zero then view-dependent optimization is discarded. Grid refinement with a view-dependent coefficient of 1 is an optimal weighting between details and computational cost, since grid resolution is coarsened with an exact proportion to the frustum perspective. However, in our experiment, a view-dependent coefficient can be assigned up to 5 before detail loss becomes noticeable. This is because foreground smoke which usually has higher detail occludes others with lower detail behind.

Table 1. Result comparison of multiple view-dependent coefficients (τ) on two sets of different grid sizes. Speed-up is compared relative to records noted with asterisk(*) each set.

Grid type	Grid res. (max)	τ	fps	speed-up
Fixed uniform	$128 \times 80 \times 48$	-	.111	-
Adaptive octree*	$128 \times 80 \times 48$	0	.110	-
Adaptive octree	$128 \times 80 \times 48$	1	.118	7.28 %
Adaptive octree	$128 \times 80 \times 48$	2	.125	14.10 %
Adaptive octree	$128 \times 80 \times 48$	5	.139	26.57 %
Adaptive octree	$128 \times 80 \times 48$	10	.158	44.13 %
Fixed uniform	$128 \times 80 \times 96$	-	.052	-
Adaptive octree*	$128 \times 80 \times 96$	0	.061	-
Adaptive octree	$128 \times 80 \times 96$	1	.064	6.03%
Adaptive octree	$128 \times 80 \times 96$	2	.068	11.77%
Adaptive octree	$128 \times 80 \times 96$	5	.091	49.14%
Adaptive octree	$128 \times 80 \times 96$	10	.133	118.67%

6 Conclusion

We have presented a view-dependent adaptive grid refinement for simulating smoke on an octree structure. The refinement conditions with adaptive thresholds are constructed incorporating viewing information. Our proposed method has successfully optimized the grid for both viewing and details. Computational cost is minimized while visual details and fluid behaviours are still preserved. The method also yields a significant difference speed-up when applied to larger domains or those with greater perspective. Additionally, our method is adjustable to flexibly match user requirements i.e., higher view-dependent coefficient to gain speed, or lower view-dependent coefficient to preserve details.

Nevertheless, there are additional issues that are not yet addressed in this current implementation. First, we have not addressed invisible areas such as smoke behind obstacles or dense smoke. The invisible areas do not have to contain fine details; therefore, can reduce unnecessary processing time. Changing the camera viewpoint is another issue. As refinement is associated with the viewing position and the frustum, rapidly changing the view e.g., translation, rotation or even zooming might cause discontinuity artifacts and detail loss; therefore, view can be changed with a limited speed relative to the simulation frame rate.

Acknowledgements. This research was partially funded by the TRF-Master Research Grants (MAG) MRG-WI535E005, Yodia Multimedia Co.,Ltd and was in part supported by the CUCP Academic Excellence Scholarship from Department of Computer Engineering, Chulalongkorn University.

References

1. Ament, M., Straßer, W.: Dynamic Grid Refinement for Fluid Simulations on Parallel Graphics Architectures. In: Proc. Eurographics Symposium on Parallel Graphics and Visualization, pp. 9–15 (2009)
2. Barran, B.: View dependent fluid dynamics. M.S. thesis, Texas A&M University (2006)
3. Berger, M., Colella, P.: Local adaptive mesh refinement for shock hydrodynamics. Journal of Computational Physics, 64–84 (1989)
4. Berger, M., Oliger, J.: Adaptive mesh refinement for hyperbolic partial differential equations. Journal of Computational Physics, 484–512 (1984)
5. Chentanez, N., Feldman, B.E., Labelle, F., Brien, J.F.O., Shewchuk, J.R.: Liquid Simulation on Lattice-Based Tetrahedral Meshes Eurographics Association, pp. 219–228 (2007)
6. Elcott, S., Tong, Y., Kanso, E., Schröder, P., Desbrun, M.: Stable, circulation-preserving, simplicial fluids. In: ACM SIGGRAPH ASIA 2008, pp. 1–11 (2008)
7. Fedkiw, R., Stam, J., Jensen, H.: Visual simulation of smoke. In: Proc. Computer Graphics and Interactive Techniques, pp. 15–22 (2001)
8. Feldman, B.E., O'Brien, J.F., Arikan, O.: Animating suspended particle explosions. ACM Transactions on Graphics 22(3), 708 (2003)
9. Foster, N.: Realistic Animation of Liquids. Graphical Models and Image Processing 58(5), 471–483 (1996)
10. Goktekin, T.G., Bargteil, A.W., O'Brien, J.F.: A method for animating viscoelastic fluids. ACM Transactions on Graphics 23(3), 463 (2004)
11. Harlow, F.H., Welch, J.E.: Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. Physics of Fluids 8(12), 2182 (1965)
12. Klingner, B.M., Feldman, B.E., Chentanez, N., O'Brien, J.F.: Fluid animation with dynamic meshes. In: ACM SIGGRAPH 2006 Papers, pp. 820 (2006)
13. Losasso, F., Gibou, F., Fedkiw, R.: Simulating water and smoke with an octree data structure. ACM Transactions on Graphics 23(3), 457 (2004)
14. Nealen, A.: Physically Based Simulation and Animation of Gaseous Phenomena in a Periodic Domain. I Can (2001)
15. Popinet, S.: Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. Journal of Computational Physics 190(2), 572–600 (2003)
16. Shi, L., Yu, Y.: Visual smoke simulation with adaptive octree refinement. In: Proc. Computer Graphics and Imaging, 1319 (2004); Stam, J.: Stable fluids. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques. ACM Press/Addison-Wesley Publishing Co., 121128 (1999)
17. Stam, J.: Stable fluids. In: Proc. Computer Graphics and Interactive Techniques, pp. 121–128. ACM Press/Addison-Wesley Publishing Co. (1999)
18. Stam, J.: Real-time fluid dynamics for games. In: Proc. Game Developer Conference, vol. 18 (2003)
19. Zheng, W., Yong, J.-H., Paul, J.-C.: Simulation of bubbles. Graphical Models 71(6), 229–239 (2009)