Real-time Simulation and Visualization using Pre-calculated Fluid Simulator States

Marek Gayer[1], Pavel Slavík[2] and František Hrdlička[3]

[1,2] *Department of Computer Science and Engineering, Faculty of Electrical Engineering*
Czech Technical University in Prague
Karlovo náměstí 13, 121 35 Prague 2
Czech Republic
xgayer@fel.cvut.cz , slavik@cslab.felk.cvut.cz

[3] *Department of Thermal and Nuclear Power Plants, Faculty of Mechanical Engineering*
Czech Technical University in Prague
Technická 4, 166 07 Prague 6
Czech Republic
hrdlicka@fsid.cvut.cz

Abstract

We present our extension for structured fluid simulators and solvers widely used in computer graphics and simulation applications. It is based on storing pre-calculated fluid simulator states (FSS). The simulation using our extension is based on partial computation with synchronous utilization of pre-calculated fluid simulator states stored on disk device.

We have incorporated this concept to real-time simulation and visualization system of combustion processes in pulverized coal boilers. The system is based on a simple fluid simulator and a coal particle system.

We compared features between classical approach of storing data sets and saving the output of visualization to common movie files formats. Furthermore, we have performed measurements of overall acceleration and disk space requirements. The disk space requirements are in orders less demanding than the ones needed for saving corresponding data sets. This allows better scalability and storing and replaying results of complex tasks with large grids and/or ten thousands of particles.

1 Introduction and related work

Nowadays, simulation and visualization of various physical and nature phenomena using fluid simulators and solvers based on the Navier-Stokes equations has major theoretical and practical importance in simulation and especially computer graphics field. These simulators and solvers are widely used for various research projects and practical applications such as animations of liquids and water [1], fire [2], gas and smoke [3], and many others. Some of them are used for special effects such as melting [4] and animations in movies [5]. In most cases, the fluid simulators are suited for specialized applications, but in most cases they can be with certain effort modified and utilized in general applications. For example, fluid simulator originally developed for animation of liquids could be modified for air flow modeling.

We use fast and simple fluid simulator for real-time simulation and visualization of combustion processes in pulverized coal boilers used in power engineering [6]. We have extended the fluid simulator with particle system engine for both simulation and visualization of combustion processes.

1.1 Real time fluid simulator conditions

Although many of the current fluid simulators offer real-time simulation and visualization, they are always limited by solution conditions under they achieve interactive frame rates. In most cases it is determined by choosing low resolution or even 2D grid for computations. Interactive frame rate of the fluid simulators can depend on the type of simulated problem. For instance, required precision or expected increase of velocities and mass in the time moment, that could cause occurrence of possible instabilities of computation in the fluid simulators must be kept in mind.

Typical example of these limitations is simulation of combustion processes, where due to sudden increase of temperature the pressure raises quickly and thus could lead the fluid simulator to instabilities. This could lead to necessity for dynamic or permanent increasing number of time-steps in computation with the consequence of slowing down the simulation.

On one hand, drawback of slowing down the simulation due to increased time step can be in some cases avoided by choosing stable fluid simulators, such as [7], [8]. These are more independent on time-steps. On the other hand, stable fluid simulators are in general not only harder to implement, but can have serious problems when

using fixed obstacles (such as walls) in the computed scene.

For example, we have tried to adopt our system to stable fluid simulator scheme based on [7]. Although this system is without doubt excellent for modeling nature phenomena such as water and clouds, we found it unsuitable for coal combustion processes due to problems with static obstacles.

Moreover, when using fluid simulators for real-time simulation and visualization in practice, heavy optimizations of the fluid simulator code to maximize the performance of the fluid simulator should be done as well.

True real-time feature of the fluid simulators is especially important in those applications, when we need to present our results or CFD applications e.g. in education. In such case, common known pitfalls of fluid simulators such as slow computation, disabling real-time visualization at all, blow-ups (serious and unrecoverable error in computation of fluid simulator), and local freezes caused by dynamic increase of simulation time-steps of fluid simulator should be avoided.

1.2 Storing the simulation output to movies or data sets

One common possible solution of maintaining real-time presentation of the fluid simulator results is to render the visualization output of the fluid simulator to common movie files (e.g. AVI or MPG). However, in this case we get several significant disadvantages. The first is total loose of interactivity. For example it is virtually impossible to even zoom to some area of the simulated problem without losing the visualization quality. Even change of any basic visualization options such as type and size of used graphics primitives is not available. Many applications moreover visualize not only one parameter, but several ones such as velocity, mass, temperature, pressure, heat fluxes, and many others with availability of immediate choose of one or more at the same time. The fluid simulators applications can also offer statistics features, which could summarize global and local parameters of the solved task (such are coal particle diameters) at selected time step upon request. Again, quick change to one of these parameters immediately is virtually impossible in convention image based animation formats.

The second common way is to store data sets of the whole simulation to the disk. But the general problem here is large requirements on the storage capacity and data transfer speed. This is namely a problem when storing characteristics of large grids and simulated objects (such as thousands of particles for maintaining simulation and visualization of combustion processes).

Under all these circumstances we have decided to design, investigate and test a new architecture of applications based on fluid simulators with partial support by pre-computed states of fluid simulator stored on external storage.

In this architecture we benefit from today's both high capacity and performance hard drives in today's commodity PCs. This allows storing the data for later replaying and processing of the results. Nowadays, high-capacity storage devices are often used in maintaining real-time visualization. One of the most often usage is for volume rendering [10], large data sets and out-of-core processing [11] such as constructing of streamlines and particle traces [12].

2 Simulation of combustion processes

We can divide fluid simulators to two types. First are complex and stable (but computationally more expensive) methods such is [7], [8] that are able to determine the flow progress independently of the time steps, but at some cost to the computational speed of the single frames. The second are unstable, where acceptable time step is dependent of the type of the task we solve. Our simulator, as well as others [1], is unstable. It is based on the principle of local simulation and uses 2D structured grid [13]. The simulated area is divided to grid cells. In each step we calculate the new characteristics (e.g. velocities, masses) for all grid cells. All calculations are reduced on nearest neighbors of the calculated cells, see Fig. 1. We periodically repeat these computations in each time step of the simulation. The stability of unstable solvers depends on proper selection of dimensions of selected grid cell, time step in regards of speed of value changes (such are velocities and mass) in cells during simulation.

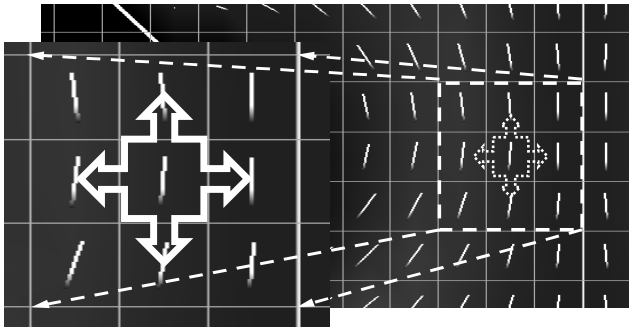

Figure 1: Division of the boiler area to 2D grid cells. The cell values in the next time step are computed from nearest neighbors only.

2.1 Fluid simulator state

Our fluid simulator state consists of the following arrays, which contains values of all grid cells:

- The velocity array [m/s]

- The mass of air array (can be easily recomputed to the array of pressures) [kg]
- The temperature array [K]
- The O_2 concentration array [%]

The values in these arrays represent overall characteristics of the cell. When computing values for the next step, the values from current state are being used. New values are computed from previous solved state using internal fluid simulator code.

2.2 Coal particle system

We use a coal particle system, enabling easy and fast computation of the combustion processes. In our system, the particle system allows us both the computation and visualization of coal mass elements in the boiler. The particles displayed and calculated do not correspond to the real coal particles in the boiler. Instead, they represent corresponding mass of coal in the cell under investigation. The quality and speed of both simulation and visualization can be altered by increasing or decreasing the amount of particles.

Instead of simulation of these processes using classical complex differential equations approach [6], we use a simple, statistical view of the combustion process [14]. The combustion and heat transfers and fluxes are being computed separately for single grid cells and corresponding particles inside them.

2.3 Simulation system architecture

For schematic architecture of our interactive simulation and visualization system combustion processes see Fig. 2. The particular parts of our system would be more described in the following text. The original approach is divided into the following steps, see Scheme 1.

```
While(Simulating)
  FluidSimulatorUpdate();
  CombustionEngine();
  Interaction();
  Visualization();
Endwhile
```

Scheme 1: Original approach progress

The `FluidSimulatorUpdate` computes new velocities masses, oxygen concentrations and temperatures for the individual cells in the current time step.

The application control and computed values are then passed to the `CombustionEngine` routine to the coal combustion system simulator. This part is independent on the fluid simulator code. It handles various actions, such as burning and moving the coal particles within the air present in the grid cell. `Interaction` routine reacts on user input, sets or modifies various solved task input parameters and boundary conditions, parameters and values. `Visualization` routine maintains visualization of computed cell values and particle characteristics. After that, time is increased by a selected time step and the computation repeats.

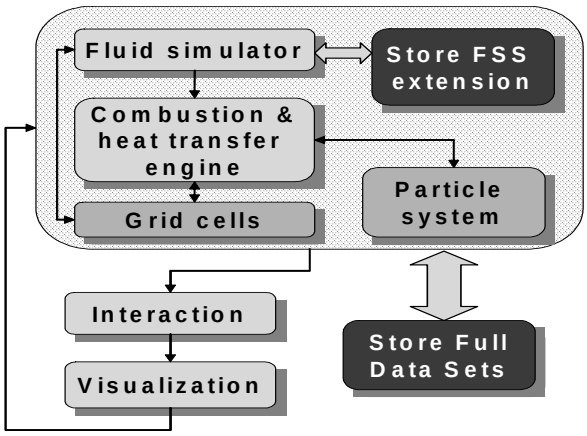

Figure 2: Schematic architecture of our interactive simulation and visualization system

```
while(Simulating)
   FluidSimulatorUpdate();
   CombustionEngine();
   if (Required)
      SaveFullDataSet();
      FullDataSetsSaved = true;
   else
      SaveFluidSimulatorState();
      FullDataSetsSaved = false;
   endif
   Interaction();
   Visualization();
Endwhile
```

Scheme 2: Storing phase

3 Extending fluid simulator with pre-calculate states feature

Each state of the fluid simulator corresponds to values computed in selected time step. In general the state consists of variables (e.g. stored in arrays such described in 2.1). We extend fluid simulator by dividing the original progress from Scheme 1 to storing and replaying phase.

In the storing phase we add the routine Save`FluidSimulatorState`, which stores the computed

variables and state of the fluid simulator, see Scheme 2. After storing the pre-computed fluid simulator states, we can replay results in replaying phase. For that purpose, we instead of computing next fluid simulator state via `FluidSimulatorUpdate` routine, restore the state from storage device through the use of routine `LoadFluidSimulatorState`, see Scheme 3. Values from pre-computed state are then passed to `CombustionEngine`, which maintains the rest of simulation, namely combustion and movement of coal particles. Thus, with support of pre-computed values, only partial computations are performed.

```
while(Simulating)
   if (FullDataSetsSaved)
      LoadFullDataSet();
   else
      LoadFluidSimulatorState();
      CombustionEngine();
   endif
   Interaction();
   Visualization();
endwhile
```

Scheme 3: Replaying phase

3.1 Storing full data sets

Instead of storing pre-calculated fluid simulator states we can store all computed data to data sets (e.g. we store characteristics computed by fluid simulator such are velocities, concentrations, masses, temperatures together with characteristics computed using `CombustionEngine` - burned heat, heat transfers in cells and burnout ratio, diameters, masses etc. of particles located in cells). This is realized by calling routines `SaveDataSets` and `LoadFullDataSet`.

This could be useful for simulation, where on top fluid simulator are further time-consuming computations. It is also suitable for further processing (e.g. out-of-core visualization), using either own code or an external real-time visualization system such as AVS or IBM Data Explorer or VTK [15].

The selection of optimal variant depends on the type of task we solve, requirements on the storage devices, the performance of the device and the time/space requirements ratio between the fluid-simulator code and post-processing code (e.g. combustion engine). In most cases, the fluid simulator part would be much more time consuming than the post-processing part or there would be no post-processing at-all. In such cases, it would be necessary to store only pre-computed fluid simulator states. But in this case, at the first frame, the full data set must be also stored to allow restoring the complete state before replaying.

3.2 An demonstration example of our results

The following figures display example simulation results. The Fig. 3 shows captured frame of real-time visualization of cell temperatures inside boiler chamber area. The Fig. 4 is a similar sample visualization of simulated coal particles (utilizing above described particle system). The results were simulated using our pulverized coal combustion application based on a simple fluid simulator. The exactly same results can be gained using described pre-computed fluid simulator states extension. In such case, system runs up to 6x faster. At the same time it consumes only a fraction of disk space, which would be required for storing corresponding full data sets (only 1.6GB of disk space for fluid simulator states per 13 minutes animation). The detailed configuration setup on which this demonstration example has been performed can be found in Table 1 and Table 2 (FSS 50 * 100 case).

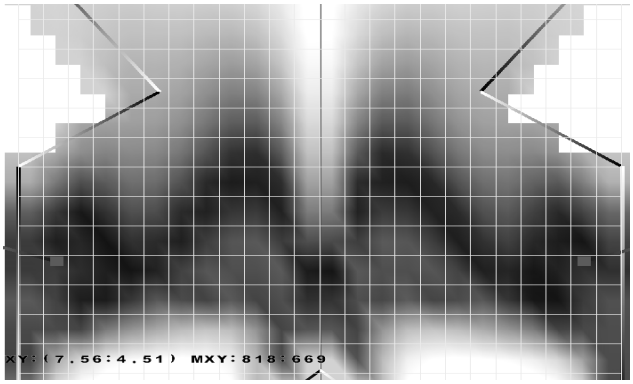

Figure 3: Sample visualization of cell characteristics. The darker values correspond to greater temperatures.

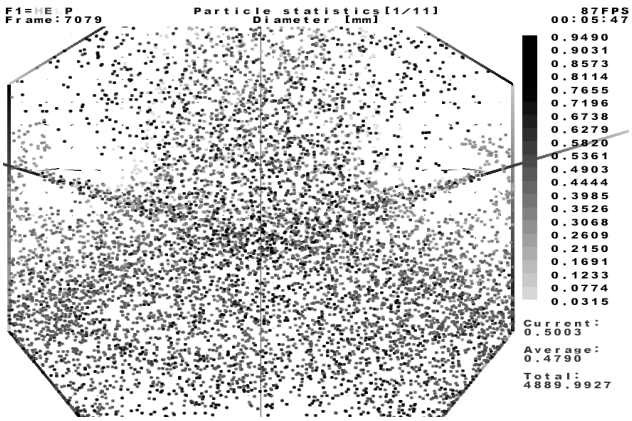

Figure 4: Visualization of simulated particle characteristics (e.g. particle diameters).

4 The results

We have successfully implemented our concept of pre-calculated fluid simulator states to a structured grid fluid simulator for simulation and visualization of combustion processes. We have simulated starting of pulverized coal combustion in a 2D simplification of power plant boiler (width 6.4 meters, height 13.9 meters). We have measured 2 cases. In the first case, the grid was set to 20 x 40 cells. In the second case, the grid was set to 50 x 100 cells, and the computation code was more precise due to decreased time step. Further, we have compared the features, performance and requirements of the two following versions: pre-computing only the fluid-simulator states (FSS) and storing full data sets (FULL).

As a storage drive for pre-computed states and full data sets we used 120GB Seagate Barracuda Ultra Ata V. Binary uncompressed data files were used for both storing FSS and FULL versions. All results and measurements were performed on a commodity AMD Athlon 1333Mhz system equipped with 512MB SDRAM PC 133 memory. The input parameters and results are summarized in the following Table 1 and Table 2.

Table 1 – Pulverized coal combustion simulation start setup

Grid size	20*40	50*100
Interactivity	Full	Full
Seek & change simulation speed ability	No	No
Simulation time step set	0.005s	0.0009s
AVG Particles/Frame	10671	9173
Total frames	11999	13329
Simulation time	1212s	5090s
AVG frame rate [Fps]	**9.9**	**2.6**

Table 2 - Results gained using pre-calculated fluid simulator state engine and full data sets

Store method / Grid size	FSS / 20*40	FULL/ 20*40	FSS / 50*100	FULL / 50*100
Interactivity	Partial	Partial	Partial	Partial
Seek & change speed ability	No	**Yes**	No	**Yes**
Generation time	1214s	1230s	5128s	5133s
Write [MB/s]	0.16	8.0	0.3	3.7
Replay time	627s	603s	816s	864s
Read [MB/s]	0.31	14.6	1.9	**21.95**
AVG Fps	**19.1**	**19.9**	**16.3**	**15.4**
Disk space GB	**0.2**	**9.4**	**1.6**	**19.1**
Total Acceleration	**x 1.9**	**x 2.0**	**x 6.2**	**x 5.9**

4.1 The results discussion

In both cases, we have gained multiple acceleration of the fluid simulator based animation. Storing the data to disk drive consumed virtually none additional time cost. This is caused due to Ultra DMA 5 transfer mode of disk data into the PC memory with minimal assistance of system processor. Without pre-calculated states and data sets, full interaction can be made to the boiler task (e.g. changing of visualization parameters, moving the coal and air jets, modification of various characteristics and simulation parameters such is fuel value of coal).

When replaying the stored results, on one hand the interactivity is limited to visualization part only. We can change all the visualization parameters and methods, choose selected simulated area, immediate switch to any visualized characteristics of particles and cells and more. On the other hand, in certain cases it is fully sufficient e.g. when presenting the simulation results in education.

When full data sets are stored, random skip to any moment of simulation time is available. Moreover, visualization play speed can be arbitrarily and easily changed (by choosing skipping of selected frames) or could be even back-reversed. On the other hand, storing full data sets is more limited to specific used storage drive.

When running demanding tasks with large cell grids, this method can suffer from both space and performance limitations of the drive. In case FULL/50*100, the disk read transfer reached 21.95MB/s. This is nearly approaching the limit of the current commodity disk drives read speed. Then due to speed limit of the drive could appear performance bottleneck when replaying the simulation. This could only be fixed by storing only selected frames (e.g. only each 1 from 10 or even 100).

When storing only the pre-computed states of the fluid simulator, the disk space requirements are considerably less then with full data sets. It seems this option is nevertheless more generally applicable in our application. Due to low storage space requirements, it is also suitable for large, even 3D cell grids with particle system with tenth thousands of particles. In this case storing full data sets would lead to very demanding requirements of both disk capacity and speed.

5 Future work

Implementing existing compression method such as GZIP or BZIP2 to FSS files could decrease both the disk space and traffic requirements for complex tasks.

Further, we plan to build real-time simulation and visualization system of pulverized coal combustion processes based on stored pre-calculated fluid simulator states organized in tree structure. It would allow fast simulation and visualization of multiple pre-computed configurations of boiler.

Lastly, we plan testing our system on very large or even datasets and we plan make another measurements of performance increase disk space save in such cases.

6 Conclusion

The simulation with pre-calculated fluid simulator extension states is based on partial computation with synchronous utilization of pre-calculated fluid simulator states stored on disk device. This concept can drastically improve the simulation and subsequent visualization speed of wide computer graphics applications based on fluid simulator while keeping the preciseness of computation unchanged. Pre-calculating states of the fluid simulator rather than storing full data sets of simulation results in much less disk space requirements, with virtually unchanged acceleration. In visualization part, all interactive actions and features such as changing the visualization parameters and visualization of arbitrary characteristics are kept. Even simple and not performance optimized applications based on 2D or 3D cell grid fluid simulators (even non-real-time) can benefit from our concept. In other words, pre-calculated fluid simulators extension can help overcome performance bottleneck of time-consuming fluid simulator codes, namely when using high-resolution grids or more precise, complex computation methods.

We have performed tests of the architecture on our coal combustion simulation and visualization system based on fluid simulator and particle system. We have demonstrated that with this concept multiple acceleration of simulation and sub sequential visualization can be gained, while requesting only small fraction of disk space requirements that would be needed for storing whole frames either as movie files (with total loose of interactivity) or full data sets.

7 Acknowledgement

This project has been partially supported by the Ministry of Education, Youth and Sports of the Czech Republic under research program No. Y04/98: 212300014 (Research in the area of information technologies and communications).

8 References

[1] N. Foster and D. Metaxas, "Realistic Animation of Liquids ", *Graphical Models and Image Processing: GMIP*, 58(5), 1996, pp. 471–483.

[2] D. Nguyen, R. Fedkiw and H. Wann Jensen, "Physically based modeling and animation of fire. *Proceedings of Computer graphics and interactive techniques*, ACM Press, 2002, pp. 721-728.

[3] R. Fedkiw, J. Stam, and H. W. Jensen, "Visual Simulation of Smoke". *Proceedings of Computer graphics and interactive techniques*, ACM Press, 2001, pp. 15–22.

[4] M. Carlson, P. Mucha and R. Brooks Van Horn, III G. Turk, "Melting and flowing". *Proceedings of the SIGGRAPH symposium on Computer animation.* ACM Press, 2002, pp. *167-174.*

[5] P. Witting, "Computational Fluid Dynamics in a Traditional Animation Environment", *Proceedings of Computer graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., 1999, pp. 129-136.

[6] J. Warnatz and U. Maas and R.W. Dibble, *Combustion Physical and Chemical Fundamentals, Modeling and Simulation, Experiments*, Springer-Verlag, Berlin, 1995.

[7] J. Stam, "Stable Fluids". *In Computer Graphics Proceedings Annual Conference Series* (Los Angeles, USA), 1999, pp. 121–127.

[8] M. Kass and G. Miller, "Rapid, Stable Fluid Dynamics for Computer Graphics". *ACM Computer Graphics (SIGGRAPH '90)*, 24(4), 1990, pp. 49-57.

[9] J. X. Chen, N. da Vittoria Lobo, C. E. Hughes, and J. M. Moshell, "Real-Time Fluid Simulation in a Dynamic Virtual Environment". *IEEE Computer Graphics and Applications*, 17(3), 1997, pp. 52–61.

[10] Eric B. Lum, Kwan Liu Ma and John Clyne. "Texture hardware assisted rendering of time-varying volume data". *Proceedings of the conference on Visualization*, IEEE Press, 2001, pp. 263-270.

[11] S. Bryson, D. Kenwright and M. Cox and D. Ellsworth and R. Haimes. "Visually exploring gigabyte data sets in real time". *Communications of the ACM.* 42(8), ACM Press, 1999, pp. 82-90.

[12] R. Bruckschen, F. Kuester, B. Hamann, and K. I. Joy, "Real-time out-of-core visualization of particle traces". *In Proceedings of the IEEE 2001 symposium on parallel and large-data visualization and graphics*, IEEE Press, 2001, pp. 45–50.

[13] M. Gayer, P. Slavík, F. Hrdlička, "Interactive System for Pulverized Coal Combustion Visualization with Fluid Simulator", *Proceedings of the Visualization, Imaging, and Image Processing.* Acta Press, Anaheim, 2002, pp. 288-293.

[14] M. Gayer, F. Hrdlička, P. Slavík, "Dynamic Visualisation of the Combustion Processes in Boilers", *Journal of WSCG*, 2002, 10(3), pp. 25-32.

[15] W. Schroeder and K. Martin W. Lorensen, "*The Design and Implementation of an Object-Oriented Toolkit for 3D Graphics and Visualization*". *IEEE Visualization*, 1996, pp. 93-100.