PRE-CALCULATED FLUID SIMULATOR STATES TREE

Marek Gayer and Pavel Slavík
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

Abstract

We present our original methodology for acceleration of fluids simulation computed by structured fluid simulators and solvers. The methodology is based on storing Pre-Calculated Fluid Simulator States (FSS) and organizing them in hierarchical tree structures allowing incremental solving, interactive replaying and modifications of simulated tasks. Thus, the parameters and boundary conditions of the simulation can be real-time modified during replaying. The simulation using our methodology is based on only partial computation with synchronous utilization of pre-calculated fluid simulator states stored on hard disk storage.
We have incorporated this concept into 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 have made comparison of FSS features with classical approach of storing data sets and saving the visualization output to common movie formats. Furthermore, we have performed measurements of overall acceleration of simulation and discussed disk space demands. The disk space requirements are in orders less demanding than the ones needed for storing corresponding data sets. This allows better scalability and storing and interactive replaying simulation results of complex tasks with large grids and/or ten thousands of particles.

Key words
CFD, Real-time and Interactive Simulation, Fluid Simulators and Solvers, Data sets, Coal Combustion.

1. Introduction

Providing real-time simulation and visualization of various physical phenomena and processes is a common goal of many research projects and applications. Real-time simulation and visualization offers many significant benefits to users of those applications:

- Obtaining the results quickly
- The results can be obtained for any selected time moment of simulation
- Possibility to get good a overview of the dynamics of the simulated process
- Offer easy manipulation of the simulated model
- User can made interactive changes to the simulated process with immediate visualization response
- Real-time visualization offers the results in readable, easily understandable and attractive form This is very important in education and demonstration of these processes in general

We could find without any doubt many other benefits of real-time simulation and visualization. Unfortunately, in many complex tasks, the full and namely precise real-time simulation cannot always be fully, easily and simply achieved, even on high performance computer systems. The typical example is simulation and modeling of various natural phenomena and manufacturing processes involving behavior of fluids using Computation Fluid Dynamics (CFD) simulation techniques, e.g. for combustion modeling [1]. In most cases, this is caused mainly due to a complex mathematical and physical description of the process and resulting time-demanding simulation calculations. In those cases, researchers and developers of simulation applications are trying to find various concepts, algorithms and tricks to achieve all or at least some of the above described real-time simulation and visualization benefits.

1.2 Speed up and optimization of simulation

The most common, general way is to try to simplify the mathematical model or to choose a model which can be effectively solved on existing computer architectures and can be easily and quickly implemented. For example, for fluid simulations, various simplified forms of Navier-Stokes equations are being used in Fluid simulators and solvers. This topic is covered by hundreds of research papers in simulation and namely computer graphics fields. The practical demand of such projects is very large and interesting for many researchers. Currently, animations of liquids [2], water surfaces and waves [3], fire [4], air and gas [5], smoke [6], fluid flows on smooth surfaces [7] and many others are being investigated. Results of some projects are used for special effects such as melting [8]

and animations in movies [9]. Often, namely in commercial products, simulation code optimization (choosing proper data structures and effective and fast algorithms and other implementation issues) are very important. The performance difference between code not-optimized and optimized code can be measured in tens or even hundreds of percents.
Although the effort described so far often leads to the possibility of real-time simulation and visualization in many applications, several possible drawbacks still remain. At first, simplification of the mathematical model leads more or less to reduction of accuracy of results. Often, various simplifications such as low resolution of grid cells for calculations and a reduced number of simulated objects (such as particles) are used. Some applications run in real-time, but with necessity of lowering the time steps of computation of results. Other applications give visually very acceptable results, but with considerably decreased accuracy of simulation. But in many applications, this approach (including described simulation code optimization) is still not sufficient to allow real-time simulation at all.

1.3 Replaying animations and data-sets

Therefore, another common way for providing real-time presentation of results is dividing the simulation and visualization into two separate steps. In the first step, the data are calculated and stored on a storage device (e.g. hard disk). In the second step, the data are replayed at interactive frame rates.
The simulated data are either stored to the standard animation formats (such as AVI and MPEG) or to data sets. In case of animations, the interactive and other advantages of above described real-time simulation are completely lost. There is no possibility for further change in the visualization method. Only one of the selected characteristics can be visualized at a time. Also post-processing (e.g. presenting simulation results statistics or data-analysis in form of various graphs) is not available at all. Also, the quality of image is reduced (especially when a zoom into specified area in the animation file is required) and the disk space requirements are relatively large for longer simulations in this case.
The data-sets allow storing of one or more computed characteristics. The interactivity is still limited; the additional changes to the already computed simulation setup and simulation configuration are of course not available. Moreover, the data-sets, if they contain several stored characteristics, have considerable requirements on disk space. As well as in the previous case, the simulation is limited to the strict time portion based on the data set saved on the disk. On the other hand, data-sets offer full interaction in the visualization part of the simulation (e.g. changing of visualized characteristic, zooming to any simulated area, offering various visualization methods, out-of-core visualization, post-processing and others). Nowadays, high-capacity storage devices are often used for providing real-time visualization. One of the most often used is for volume rendering [10], large data sets and out-of-core processing [11] such as construction of streamlines and particle traces [12].

1.4 Our effort

Our research effort is primarily focused on simulation and visualization of combustion processes. In our previous effort, we had developed a simple fluid simulator with combination of coal particle systems for modeling and visualization of combustion process dynamics in the combustion boiler chamber [13]. This way, we were able to interactively and in real-time, simulate and visualize tens of various particle and cell characteristics regarding the combustion process inside the boiler chamber (such as temperatures, velocities, pressures, mass fluxes, heat radiation and many others).
Currently, our application is able to run in real time at interactive frame rates, but with certain simplifications (namely setting low resolution 2D computation grid and using only about 10,000 particles). Setting a higher resolution computation grid and computational time steps leads to a considerable slowing down of the simulation.
Therefore, in our current research, we have decided to design and implement a unique architecture, which would combine the benefits of interactive, real-time simulation, allowing for testing various boiler modifications, combined with utilization of speeding up the visualization using pre-computed results stored in data sets.
Instead of saving simulated results to data sets, we only save partial computations of the most time demanding part of the simulation. Thus, we save only the states of the fluid simulator data, and not any other results such as particle data. This results in a considerable speed-up of simulation and consequent visualization. Thus, our approach is similar to using data sets, but with drastically reduced disk space requirements. Next, namely low disk space requirements allow us to construct hierarchical structures forming those pre-calculated results. It allows for investigating various configurations and modifications of boilers with accelerated speed. Our concept can be easily reusable in other applications based on fluid simulators and solvers.

2. Simulation of combustion processes using fluid simulator and particle systems

The fluid simulator or solver is typically a core element, on which simulation of the modeled task is being solved. The architecture and concept of our fluid simulator has been described in detail in a previous paper (see [13]). In general, the fluid simulator core computation is the most time consuming part of fluid applications.
Our simulator is based on the principle of local simulation and uses a 2D structured grid. The simulated area is

divided into 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 those computations in each time step of the simulation.
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 quality and speed of both simulation and visualization can be altered by increasing or decreasing the amount of particles. 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.

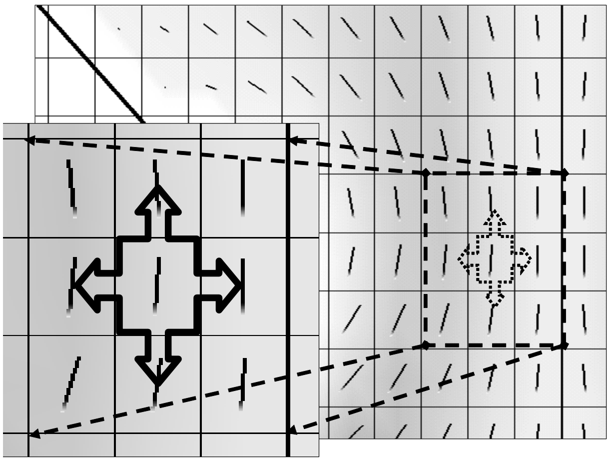

Figure 1: Division of the boiler area part to 2D grid cells.

3. Fluid simulator states (FSS)

In general, the results of the fluid simulator computation consist of computed values, which describe the physical situation for each of the grid cells in the simulated area (see Fig. 1). These values can be furthermore used in subsequent simulation and/or visualization. When computing values for the next step of the fluid simulator, the values from the current state are being used. New values are computed from the previous solved state using internal fluid simulator code. In general, the state consists of arrays of variables. Our fluid simulator state description consists of the following arrays, which contain values of all grid cells:

- The velocity array (X and Y component part) [m/s]
- The mass of air array [kg]
- The temperature array [K]
- The O_2 concentration array [%]

Because the generation of values for the next time step is the most time-consuming part of the simulation, we decided to build an extension, which stores these states to the disk storage. The extension stores and reads the pre-calculated states. The simulation is divided into storing and replaying phases. With support of pre-calculated values, only partial computations are being performed in the replaying phase. In opposition to storing full data sets, storing fluid simulator states stores only those part of the characteristics which are computed using fluid simulator code and are difficult to compute. The rest of the characteristics are computed using our combustion & heat transfer engine. Similarly, the particle system characteristics are not being saved at all. See Fig. 2 for an illustrative incorporation of the Fluid Simulator State (FSS) extension into our system.

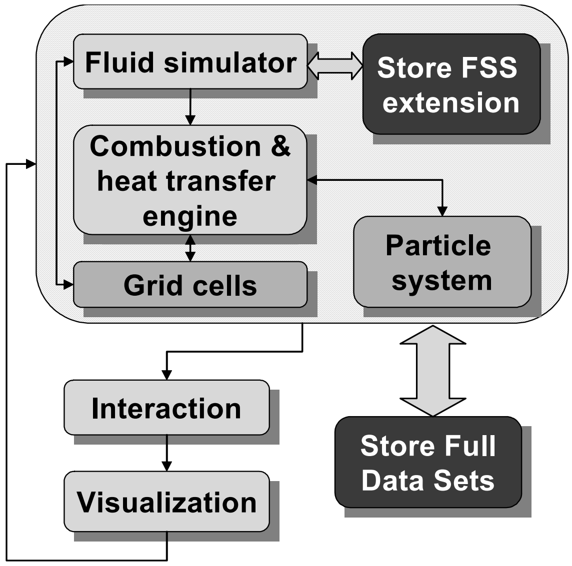

Figure 2: Incorporation of storing fluid simulator states (FSS) extension into our fluid simulator application

4. Forming FSS to tree cluster structures

The advantages, specifically low disk space requirements, allow constructing and storing fluid simulator states for even large simulations.
The same reasoning also allows for creation and organization of these states in hierarchical tree structures. In such trees, a node corresponds to one pre-calculated state. Each of the nodes can be easily and interactively extended by simply storing another pre-calculated FSS. The new pre-calculated state uses as boundary conditions the results of the state from which it was derived (corresponds to the ancestor node). Each of the nodes has its unique ID of the tree where it belongs and unique number of nodes in the tree. This way, the sequence of connected simulated task modifications are being defined in every node.
Replaying the path in the tree from top to bottom results in sequential replaying of connected saved states of the boiler with selected modifications of configuration in each of the nodes. After finishing running a part of the simulation, which utilizes pre-calculated FSS, the

simulation can easily continue without any pre-calculated data (the last values computed using FSS are used as boundary conditions for further simulation, which runs at the original, not accelerated speed).

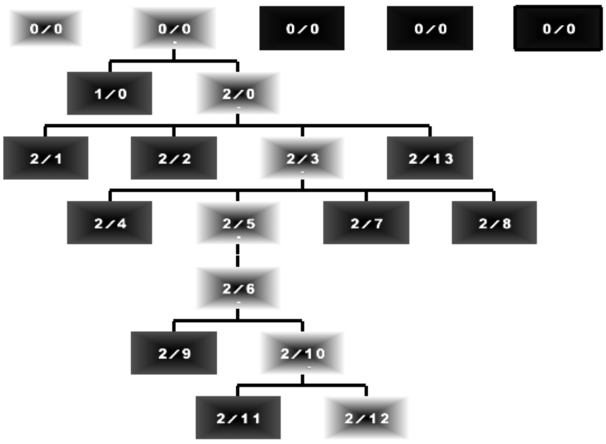

Figure 3: Hierarchical tree of pre-calculated fluid simulator states. Each node represents one file with saved FSS. The current selected path of simulation with configurations modifications is being highlighted.

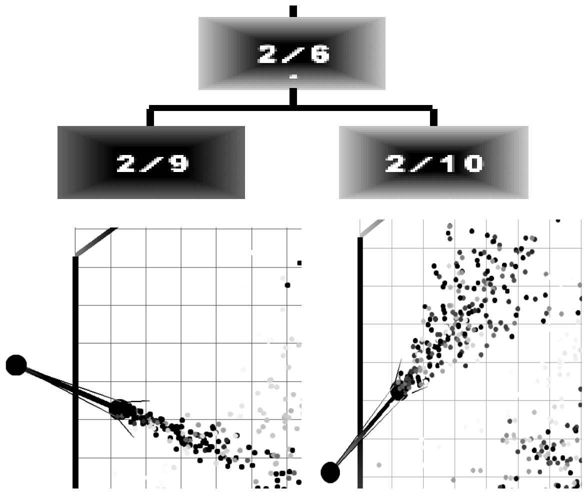

Figure 4: Illustration of interactive change of parameters of simulation in different FSS sub-trees. In the left sub-tree (2/9) – the inlet is kept in original position; in the right sub-tree (2/10) – the position and spread angle of the inlet has been changed. The FSS accelerated simulation continues with changed configuration.

The whole tree can be visualized and implemented to accept interactive user actions and modifications – e.g. extending the tree with nodes for another part of simulation, deleting parts of the partial simulation solution etc. An example of an FSS tree is shown in Fig. 3 and Fig. 4. This concept brings the following advantages over classical data-sets:

- Considerable speedup of replaying results, while requesting only a fraction of disk space requirements that would be required for corresponding data sets.
- Incremental and step-by-step solution and creating of re-playable results of the simulated problem. This is advantageous for solution complex tasks.
- Availability of interactive addition and deleting other parts of the solution and modifications of the task.
- Hierarchical storage of states allows utilizing of previous values without needing to restart the simulation. Utilizing pre-calculated states in ancestral sub-trees leads to another saving of the disk space.
- Possibility to interactively change the simulation state and boundary conditions of the simulation in each of the nodes with immediate reflection (e.g. in each node we can change the coal inlet parameters, combustible properties, change the amount of oxygen and many more). This is shown in Fig. 4. In the first sub-tree of a FSS node, the simulation continues with the original inlet position. In the second sub-tree, we have changed the direction and spread angle of the inlet and the simulation continues with changed parameters with simulation acceleration being kept. The user decides which part of the tree (and corresponding configuration) would be replayed.
- Possibility of selecting and constructing various paths through the tree to interactively replay parts of the various simulation configurations. These paths can be replayed separately and each of them represents another interactive simulation with modifications and other actions of the simulation user reflected.
- It enables students and users of simulation applications to extend the prepared and pre-calculated simulation solutions with their own modifications while keeping a high, accelerated speed of the replaying.
- Selected nodes can be made read-only, to disable accidental modification and deleting of the base FSS tree structure, which is intended to be used by all students and users.

5. Storage implementation of FSS tree

The fluid simulator states for corresponding FSS tree nodes are being stored in binary files. Every node of the FSS tree contains pre-calculated values of part simulation. Each node is also described with text files containing simulated case information and modifications, such as combustible and inlet properties. It also includes unique numbers of corresponding nodes and the node from which the node was derived from, to allow later reconstruction of the tree from the already stored files on the disk.

6. Comparison of FSS and Full Data Sets

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 the starting of pulverized coal combustion in a 2D approximation of a 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 a decreased time step. Furthermore, 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 (FDS), containing both cells and particle characteristics.
As a storage drive for pre-computed states and full data sets we used a 120GB Seagate Barracuda Ultra Ata V. All results and measurements were performed on a commodity AMD Athlon 1333Mhz system equipped with 512MB SDRAM PC 133 memory.
Each of the cases contained about 12500 animation frames and also average about 10000 particles per animation frame. The grid size has been set to 20 * 40 and 50 * 100 cells. The results are summarized in the following Table 1. In each of the cases, we got acceleration from 1.9 to 5.9 of the total amount of frames per second (FPS) compared to direct simulation and immediate visualization. Storing the data to disk drive consumed virtually no additional time cost compared to the version without storing the data (with fast Ultra DMA 5 data transfer mode used).

Store method / Grid size	FSS / 20*40	FDS / 20*40	FSS / 50*100	FDS / 50*100
Simulation 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**

Table 1 - Results gained using pre-calculated fluid simulator state engine and full data sets

Storing full data sets is more limited to the specific storage drive used. When running demanding tasks with large cell grids, this method would suffer from both space and performance limitations of the drive. Then due to the speed limit of the drive, a performance bottleneck could appear when replaying the simulation. This could only be fixed by storing only selected frames (e.g. only 1 from 10 or even only 1 from 100), but with losing the precision and resolution of the data stored. It is also not suitable to form the full data sets for interactive, hierarchical structures, because the disk space requirements would be too demanding.
When storing only the pre-computed states of the fluid simulator, the disk space requirements are in orders less than would be needed for storing corresponding data sets with simulation results. Due to low storage space requirements, it is also suitable for large, even 3D grids with a particle system with ten thousands of particles.

7. A demonstration example of our results

The following figures display example simulation results of our application based on FSS. Fig. 5 shows a captured frame of real-time visualization of cell temperatures together with floating virtual coal particles inside the boiler chamber area. Our application can arbitrarily select and switch among a total of about 40 characteristics in the visualization phase. The results were simulated using our pulverized coal combustion application based on a simple fluid simulator. The exact same results (including switching to all other characteristics) can be gained using the pre-computed fluid simulator states extension described. In such a case, the system runs up to 6x faster. At the same time it consumes only a fraction of the disk space, which would be required for storing corresponding full data sets. In our test it consumed only 1.6GB of disk space for fluid simulator states per 13 minutes animation, while for storing corresponding data sets, 19.1GB would be needed. The detailed configuration setup on which this demonstration example has been performed can be found in Table 1.

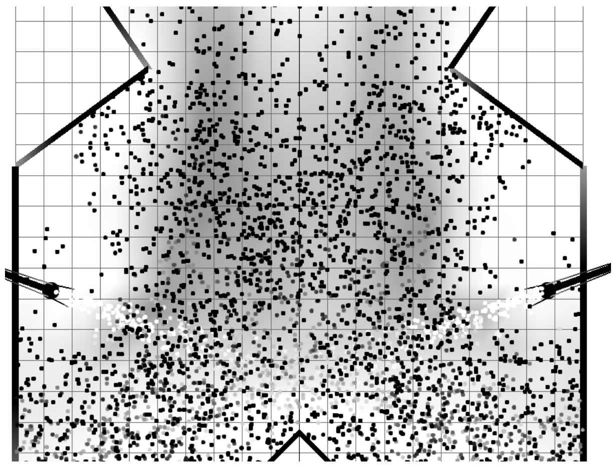

Figure 5: Sample visualization of selected cell characteristics together with floating virtual coal particles.

8. Conclusion

The simulation with a pre-calculated fluid simulator states (FSS) is based on partial computation with synchronous

utilization of pre-calculated fluid simulator states stored on a 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. 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 the performance bottleneck of time-consuming fluid simulator codes, namely when using high-resolution grids or more precise, complex computation methods.
We have designed and implemented hierarchical tree structures built from pre-calculated fluid simulator states, which allow incremental, progressive and easy construction of various configurations of the boiler with high speed, interactive visualization and replaying of results. The modifications of simulation boundary conditions are available in every node of the FSS tree. Thus, every interactively selected path in the FSS tree corresponds to one modified simulation solution.
We have performed tests of the architecture on our coal combustion simulation and visualization system based on the 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 a 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, while keeping the acceleration virtually either unchanged or better.
The original concept of a pre-calculated fluid simulator states tree can be easily utilized in various applications based on fluid simulators and solvers as well.
Currently, we use our results for an education application allowing the students to interactively test and preview-design many configurations of pulverized coal boilers.

9. 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).

References

[1] J. Gillespie, The use of CFD in design of combustion equipment, *Progress in Computational Fluid Dynamics, 1*(1/2/3), 2001, 80-90.

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

[3] D. Enright, S. Marschner and R. Fedkiw, Animation and rendering of complex water surfaces, *Proc. Computer graphics and interactive techniques,* ACM Press, 2002, 736-744.

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

[5] J. Stam, Stable Fluids, *Proc. Computer Graphics Proceedings and interactive techniques,* ACM Press, 1999, 121-127.

[6] R. Fedkiw, J. Stam, and H. W. Jensen, Visual Simulation of Smoke. *Proc. Computer graphics and interactive techniques*, ACM Press, 2001, 15-22.

[7] J. Stam, Flows on Surfaces of Arbitrary Topology, *Proc. Computer graphics and interactive techniques (ACM SIGGRAPH 2003),* ACM Press, 2003. http://www.dgp.toronto.edu/people/stam/reality/Research/pub.html

[8] M. Carlson and P. Mucha and R. Brooks Van Horn, III G. Turk. Melting and flowing, *Proc. SIGGRAPH symposium on Computer animation.* ACM Press, 2002, 167-174.

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

[10] E. B. Lum and K. L. Ma and J. Clyne, Texture hardware assisted rendering of time-varying volume data. *Proc. Conference on Visualization*, IEEE Press, 2001, 263-270.

[11] R. Bruckschen, F. Kuester, B. Hamann, and K. I. Joy, Real-time out-of-core visualization of particle traces. *Proc. IEEE 2001 symposium on parallel and large-data visualization and graphics,* IEEE Press, 2001, 45-50.

[12] S. Bryson and 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, 82-90.

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

[14] J. Tomeczek. *Coal Combustion* (Krieger Publishing Company, 1994).