INTERACTIVE SYSTEM FOR PULVERIZED COAL COMBUSTION VISUALIZATION WITH FLUID SIMULATOR

Marek Gayer[1], Pavel Slavík[2] and František Hrdlička[3]
Department of Computer Science and Engineering
Czech Technical University, Karlovo náměstí 13, 121 35 Prague 2
Czech Republic

ABSTRACT

Designing pulverized coal combustion simulation and visualization systems is still a complex task. The current solutions, such as the commercial CFD packages are focused on the precision of the computation. Our solution is suited to the education and computer graphics area, where the precision does not take the first, but also not the last place. The main factor in our solution utilizes the interactivity of the system and its ability to visualize the gained results real-time.

We briefly describe the key elements of our system. In the theoretical part of the paper, we introduce our original concept of the simple air fluid simulator, which can be used in general airflow computer graphics animations. Our method is based only on applying the two physics fundamentals – Newton's Second Law and The Continuity Equation. Furthermore, we briefly describe our simplified coal combustion engine, the heat transfers and virtual coal particle system, which are used for visualization of the coal flow.

Key Words: Visualization, Particle Systems, CFD, Combustion, Engineering Education

1. INTRODUCTION

In recent years world attention has focused increasingly on man's use of energy as well as the available technologies for energy production and consequences of their use. Today's coal power plants create a significant part of the overall pollution. The kernel of the coal power plant is a combusting boiler. The goal is to improve design of the boilers - to reduce pollution, find ways of preparing fuel, determine coal particle sizes and quantity, speed etc. Instead of constructing real boilers and investigating the boiler behavior in reality, computer aided simulation and visualization models are often used. The models that simulate combustion processes are of various types.

The most important and complex task for simulation and visualization of the combustion processes is the modeling of flow (flow of the air and of the combustibles during the combustion process). Here we can use some approaches, which use a mixture of experimental and theoretical approaches. For example, in our previous work we have used the airflow streams running from the jets, described as Abramovic's streams [1]. These methodologies can bring very fast and quite reliable results, however with the lost of generality. The current research tends to use full understanding and modeling of the physical and mathematical background of the problem. This area is subject to intensive research by the scientists and mathematicians from the Computational Fluid Dynamics (CFD) field.

There are numerous CFD software packages that are able to solve fluid motion and even combustion processes with very high precision, but at the cost of extreme demands on the computational speed. In general, we have to choose the proper method or software package based on a compromise between the required simulation precision and computation speed. In the computer graphics and education applications, the interactivity of the model and visualization speed suitable for real-time animations is of great importance, so various simplifications to the physical model are often accepted.
In recent years, most scientists use simplified Navier-Stokes equations as the base of the physical model. The fluid simulators and solvers based on the Navier-Stokes equations are used for the various practical applications such as the animation of liquids, gas and smoke [2], aerodynamics simulation and animation [3], animation of the water surface [4] and many others. Some of them are even used for animations and special effects where the resulting pictures play a decisive role for the applications such as movies [5]. However, current models, which deal with simulation and visualization of behavior of flames and are more concentrated on the visualization part of the combustion process (e.g. movies etc.), often use several other approaches like cellular automata [6] or diffusion processes [7].

There are two types of Navier-Stokes simulators. Most of them such as [8] use unstable, time-step dependent solutions of the Navier-Stokes equations. Others such as can be found in [9] and [10] use stable fluid models that are able to determine the flow progress independently of

---

[1,2] Department of Computer Science and Engineering
[3] Faculty of Mechanical Engineering, Department of Thermal and Nuclear Power Plants

the time steps, but at some cost to the computational speed of the single frames. In most cases, the solvers are suited for special applications, but in most cases they can be modified and reused in general applications.

Our interactive coal combustion simulation and visualization system is a voxel[4] application, the heart of which consists of the fluid simulator. It is based on a simple approach, which on the top level consists only of application of the two physics laws – Newton's Second Law and The Continuity Equation. This methodology could be applied to the general computer graphics visualizing and animations tasks involving the move of the air mass. The heat and temperature changes generated during the combustion processes are computed using a simplified coal pulverized combustion engine. Our fluid solver also allows distribution of the heat by the moving mass. We use a virtual coal particle system, which is used both for interaction (and combustion) with the hot air volumes and for the visualization of the coal flow.

2. THE FLUID SIMULATOR PHYSICAL BACKGROUND

Rather than using complex and stable (but computationally more expensive) methods such is in [10], [11] or finite differences methods such as in [8] for solving differential equations we use only the principle of local simulation.

In our approach, the air in the boiler could be considered as in viscous and a stationary flow. We assume Newton's Second Law and The Continuity Equation (mass should be conserved). See Eq. 1, 2.

$$\frac{dv}{dt} = \frac{F}{m} \qquad [m^2 s^{-1}] \tag{1}$$

$$\frac{\partial m}{\partial t} = \oiint v \rho dS \qquad [kgs^{-1}] \tag{2}$$

Newton's Second Law says that change of the velocity per specified time *dt* of an object is dependent on the forces acting on it. In our case, the force can be expressed as the differences of pressures (*dp*) between the neighboring voxels (Eq. 3).

$$F = dp\ S \qquad [N] \tag{3}$$

The *S* is the face surface between the neighboring voxels. For 2D space, we assume the depth of the voxels to be equal to 1. This is needed for the equations of the face surface. Thus for the face surface between the neighboring voxels in the x and y direction in 2D space we get (Eq. 4,5):

$$S_x = h\ d = h \qquad [m^2] \tag{4}$$

$$S_y = w\ d = w \qquad [m^2] \tag{5}$$

The *h*, *w* and *d* are the height, width, and depth of the voxel. The difference of the pressure *dp* between the two voxels can be calculated from the total air mass *m* in the voxel using the state equation of the perfect gas (Eq. 6,7).

$$p = k\ m \qquad [Pa] \tag{6}$$

$$dp = k\ (m_2 - m_1) \qquad [Pa] \tag{7}$$

The constant is *k*, which can be derived from the perfect gas equation (see Eq. 8).

$$k = \frac{RT}{VM} \tag{8}$$

Using Newton's Second Law and Eq. 3 for a selected voxel we get the increase of the selected voxel velocity *dv* per time step *dt* as Eq. 9:

$$dv = \frac{dpSdt}{m} \qquad [m^2] \tag{9}$$

We calculate this equation for all the components of the current voxel velocity (dv_x, dv_y for 2D space and dv_x, dv_y and dv_z for 3D space).

Next, for the second fluid simulator step, we must calculate the change of the mass in the current voxel during the time step *dt*. By using the continuity equation we get (Eq. 10):

$$dm = dv\ \rho\ dS\ dt \qquad [kg] \tag{10}$$

The ρ is the density of the voxel and could be easily calculated from the mass in the voxel and the volume of the voxels. The *dm* is the mass increase in the selected direction. We calculate this equation for all the directions (*X,Y* for the 2D case, *X, Y, Z* for the 3D case). We also update the temperature inside the voxel by the weighted average of the temperatures of the mass which is present in the voxel and the mass flowing from neighbor voxel toward the current voxel.

The viscosity is added to the solver using a simple equation. The friction tension between the next voxels is determined by:

$$\tau = \mu \frac{dv}{dS} \qquad [Pas] \tag{11}$$

The τ is fiction pension (pressure), μ is dynamic viscosity, *dv* is difference of the velocities of the neighboring voxels and *dS* is the touching space face.

We set the proper boundary conditions corresponding to the surrounding environment. For the voxel, which collides with the boiler walls we assume that no air can pass through/from it and thus the normal component of

[4] Currently, our system is implemented only in 2D space. Although in this case of describing the separate cells in the boiler space we should speak about slice of the voxel, we keep for simplicity the word voxel.

the air velocity is zero. Behind the outlet voxels of the boiler we assume atmospheric pressure $p=10^5$ Pa.

In each step we calculate the new values of the velocity and mass for all the voxels in the boiler. We periodically repeat the computation until the required time step of the boiler simulation is reached. We have implemented this methodology into working code which simulates the movement of the air inside the streams in real-time.

The main advantage of this method is very fast convergence of the results. No calculations are performed over the global voxel array. The results of this method for the specified time could be immediately used at the runtime. Our method, as well as other numeric methods, is principally independent of the boundary conditions, which can moreover be changed anytime during the simulation.

On the other hand, like [8] this method is conditionally stable. The stability depends on the size of the voxel, time step and maximum velocity and mass values in the voxels during solution. This is caused by only reducing all the calculations on the nearest neighbors of the calculated voxels. Thus, if the velocity in the voxel exceeds the maximum speed, it allows the longer distance than the length of the voxel to pass as mass for the time *dt*. Other instabilities can occur in situations, when from some voxel calculations, the result is that more air mass should flow out than the total present mass in the voxel. Thus, we must carefully set or compute the *dt* parameter regarding the voxel size to avoid such cases.

3. IMPLEMENTATION OF THE FLUID SIMULATOR

Our fluid simulator algorithm has been implemented in 2D voxel space. However, it can be easily extended to 3D space.

The fluid simulator consists of the following arrays:

1) The velocity array [m/s]
2) The air mass flow array which can be easily recomputed to the array of the air flow pressures [kg]
3) The temperature array [K]
4) The O_2 concentration array [values from 0.0 to 1.0]

The chamber of the boiler is divided to the voxel area of the dimensions of the `XRES` * `YRES` arrays. For each of the arrays we keep two instances in the memory. These are accessed by the two pointers that define which of this array is the source (holds the state at the given time *t*) and destination (holds the state at the time *t* + *dt*). At the end of the simulation step, we swap the source and destination pointers. Regarding the equations above, we choose the following representations of the arrays:

The mass flow (M0, M1), temperature (T0, T1) and concentration of O_2 (C0, C1) arrays:

```
M0,T0,C0[XRES][YRES], M1,T1,C1[XRES][YRES]
```

The values in these arrays represent overall characteristics of the voxel and are related to the middle of the voxel.

The velocity arrays:

```
VX0[XRES+1][YRES], VX1[XRES+1][YRES]
VY0[XRES][YRES+1], VY1[XRES][YRES+1]
```

The values of the velocity arrays are related to the center of the walls of the margins of each voxel.

For example, to the voxel with the mass `M[0][0]` correspond the velocities `VX[0][0]`, `VX[1][0]`, `VY[0][0]` and `VY[0][1]`. This representation allows fast computation of the mass, which flows between the neighbors voxels. It is also used for example in [8]. The situation is shown in the following Figure 1

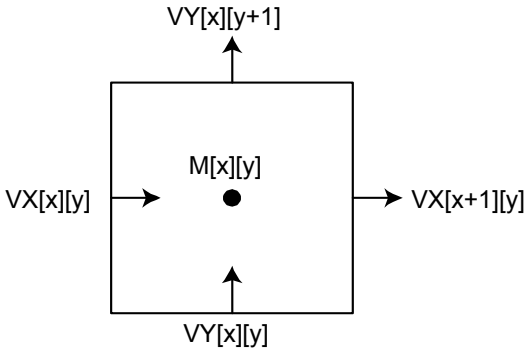

Figure 1: Representations of velocity and mass arrays

The air flow simulation is divided to the following steps:

```
Initialize;
while( simulating ) {
     VelocityStep(VX0, VY0, M0, VX1, VY1,
     dt);
     MassAndTemperatureStep(VX0, VY0, M0,
     M1, T0, T1, C0, C1, dt);
     SwapAllThePointers;
     CombustionSystemSimulatorCode(dt);
}
```

The `Initialize` routine sets all the initial conditions such as the pressure, velocities and obstacles at places where the boiler walls are detected. The `VelocityStep` routine counts new velocities for all the individual voxels. Depending on these velocities the `MassAndTemperatureStep` routine calculates all the mass changes from the nearest neighbor voxels for the corresponding voxels. After that, the `SwapAllPointers` routine swaps the pointers of the source and destination arrays. The program control is than passed through the `CombustionSystemSimulatorCode` routine to the coal combustion system simulator. This part is independent of the fluid simulator code. It handles various actions, such as interacting and moving of the coal particles with the air in the voxel, changing the boundary conditions accordingly to the user interaction, visualization of the results, etc. These parts are briefly

described in the following text. After the code of the simulator is executed, the fluid simulator code continues running.

4. THE CONCEPT OF THE VIRTUAL COAL PARTICLES

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 the corresponding mass of coal in the voxel under investigation. Therefore, we call them virtual coal particles. Thus, one virtual coal particle carries many real coal particles. The quality and speed of simulation and visualization could be altered by increasing or decreasing the amount of these virtual particles.

The movement of the virtual coal particles through the voxels is determined by the gravity force F_g and aerodynamic resistance F_0. These forces are computed from the following equations:

$$F_g = m\,g \qquad [N] \tag{11}$$

$$\vec{F}_0 = \frac{1}{2} c\,\rho\,S\,dv^2 \qquad [N] \tag{12}$$

The m is the mass of the real coal particle, c is the coefficient of the air resistance, ρ is the density of the air in the present voxel, S is the surface of the particle cross section and dv is the difference of the velocities of the particle and air velocity in the voxel, where the particle is located (the air flow around the particle).
From Eq. 11 and Eq. 12 we get for the particle acceleration (Eq. 13):

$$\vec{a} = \frac{c\,\rho\,S\,dv^2}{2\,m} - \vec{g} \qquad [m^2 s^{-1}] \tag{13}$$

The velocity of the virtual coal particle is then modified using Eq. 14:

$$\vec{v}_p = \vec{v}_p + \vec{a} \cdot dt \qquad [ms^{-1}] \tag{14}$$

Before moving a particle to the predicted destination, we must check for possible collisions with the wall. If this is the case, the particle track is mirrored and bounced from the wall.

5. COMBUSTION AND HEAT TRANSFER

The combustion process of the pulverized coal and resulting heat convection is a quite complex problem [12], [13]. Again we are exploring some simplifications due to the need for the fast computation. First, we are using a simplified combustion process. Instead of simulation of these processes using complex differential equations, we use a statistical view of the combustion process. We consider the fuel value of the coal and molar capacities of the combustibles. The combustion is being computed separately for all the single voxels.

To start combusting coal, two conditions must be satisfied: in the computed voxel, there must be at least some minimal combustion temperature (which is defined in our case to be 573 K), and a proper mass of coal and air that is to be burned.

Depending on the current temperature, weight and proportion of the coal, the virtual coal particles are being burned. For air mass, we just decrease the corresponding 0_2 concentration. For coal particles, we decrease the amount of the combustible part of the particle.. If the mass of the combustible part reaches some minimal value, we assume that the coal particle is burned out and we change it to the burnt gas particle. See Figure 2.

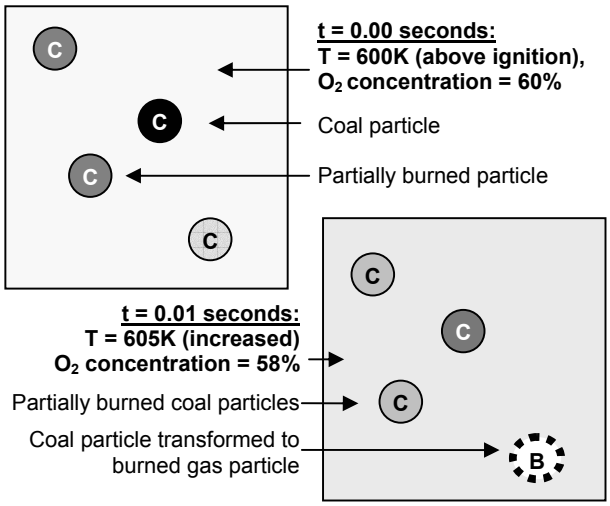

Figure 2: Example interaction of coal particles during the combustion process for the time dt in a selected voxel

Between these processes, depending on reaction heat transfer, the released heat is transferred to the air mass, which is present in the current voxel. Therefore, the overall temperature of the voxel increases.

Because of the dynamic processes in the boiler, the heat is distributed by the air mass to the other voxels, thus increasing the temperature and making it possible to start other combustion reactions. We also count the heat radiation between the walls and the mass in the voxels. The heat transferred from the given surface of the voxel F during the time dt is comparable to differences of the temperatures of the wall and the voxel (power of four) [12]. We also need to determine the coefficient of the radiation C_{12} regarding the present mass in the current voxel. These ideas are summarized in Eq. 15.

$$Q = F \cdot C_{1.2} \cdot \left\{ \left(\frac{T_1}{100} \right)^4 - \left(\frac{T_2}{100} \right)^4 \right\} \cdot dt \qquad [J] \tag{15}$$

We assume, for the sake of simplicity, that the temperature of the walls is constant (typically below the minimal combustion temperature).

6. VISUALIZATION OF THE RESULTS

Our system uses the industry standard OpenGL platform for reliable and fast visualization. This means that our system could be used on a standard low-cost graphics accelerator. There is no lack of speed in particle visualization even when using coal particle streams consisting of ten-thousands of particles. The selected local characteristics in the voxel, such as the total temperature, mass storage, the wattage, and heat flux state and/or changes can be visualized in real-time. We use OpenGL linear interpolated quads with the support of the graphics hardware acceleration for visualization of the voxel characteristics (see Figures 3 and 5). Utilizing the advantage of the particle system concept, we can easily construct the particle traces. We produce this effect by saving the previous particle positions and characteristics.

7. THE COMBUSTION SYSTEM IMPLEMENTATION

All the parts of our system have been implemented in the standard, ANSI C computer language. In the current implementation, we can use some interactive actions even during the simulation, in real-time, without restarting the whole simulation. Thus, we can move jets and disable/enable/delete/modify parameters of air or coal jets. Other features include tracking of the selected particles. It allows us to monitor its position and characteristics at any time until it leaves the boiler space. Visualization is based on the OpenGL graphics interface. A windowing interface is maintained by the GLUT library. This means that our system is easily and fully portable to other systems.

8. DISCUSSION OF OUR RESULTS

The current research brings promising results. On a real test boiler (dimensions 6.4 m x 13.7 m) we have simulated and visualized combustion processes. To compare our results with current CFD methodology, we have used the professional CFD software package FLUENT 5.5. We have discussed the results obtained with the experts from the Faculty of Mechanical Engineering of CTU with positive response.

In Table 1 we summarize the global parameter results of our current implementation in the comparison with FLUENT. It clearly shows a well overall design and implementation of our ideas. Among the global parameters that indicate comparable results with the professional CFD package FLUENT, we have compared the contours of the temperature and velocity maps. They are visually similar - see Figures 3, 4, 5 and 6. Moreover, we are developing numeric comparison software that is able to statistically compare the results in each of the voxels between our system and Fluent and quantify corresponding errors and differences. Preliminary results from the active part of the boiler show that the share of voxels with deflection up to 20% from values obtained from FLUENT varies between 60% and 80% for temperature, flow directions and other values.

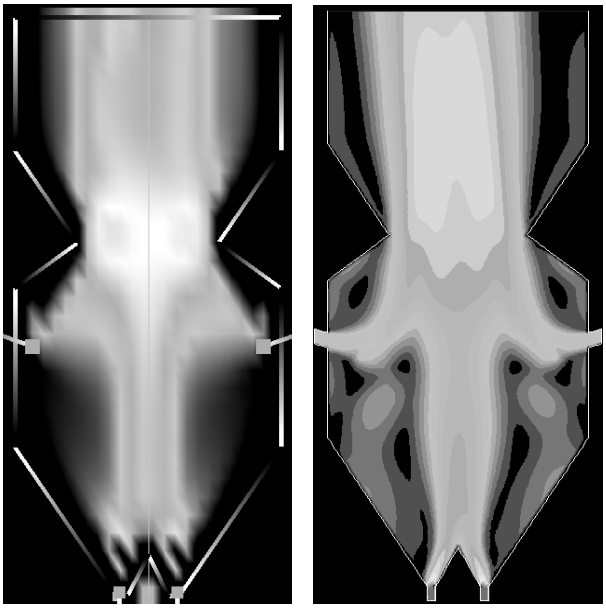

Figure 3, 4: The contours of velocity in the test boiler
Left: Our system, Right the CFD solver FLUENT 5.5

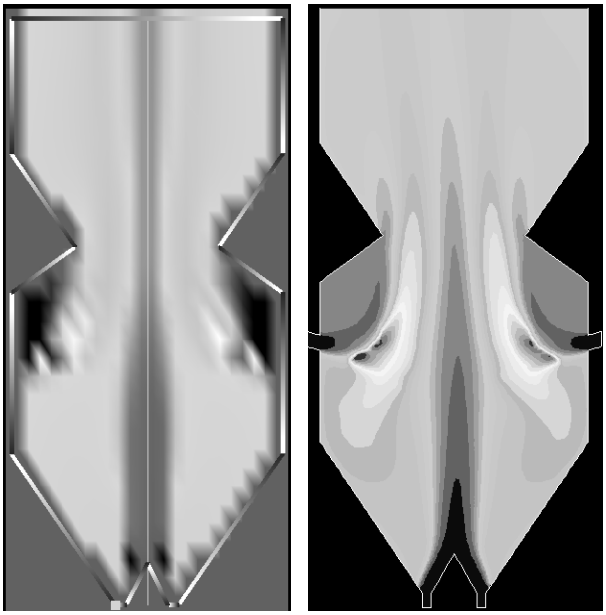

Figure 5,6: The contours of temperature in the test boiler
Left: Our system, Right the CFD solver FLUENT 5.5

Parameter	Our system	FLUENT 5.5
Average temperature	890 ^{o}C	1002 ^{o}C
Outlet temperature	814 ^{o}C	1068 ^{o}C
Maximum temperature	2546 ^{o}C	2488 ^{o}C
Average stream velocity	14 m/s	11 m/s
Average outlet velocity	25 m/s	21 m/s
Maximum velocity	56 m/s	48 m/s
Wattage	187 w/m^3	232 w/m^3
Mass total	21.1 kg	21.3 kg
Time needed to converge solution	**20 seconds**	**7 hours**

Table 1 - Global parameters results in the test boiler

9. CONCLUSION

Our coal combustion system is based on the fluid simulator. The fluid simulator allows real-time computation of the airflow. It utilizes Newton's Second Law and The Continuity Equation. The simulator is very easy to implement and thus can be reusable in various computer graphics tasks relating to air flow simulation and visualization.

The high speed of the fluid simulator and combustion system allows real-time visualization of the results (using OpenGL graphics interface). The system has been implemented in 2D voxel space, and regarding the methodology used can be easily extended to 3D space.

We have tested our system by modeling of a boiler with real dimensions, characteristics and parameters. The behavior and results gained from our system were comparable with a situation in a real boiler and with the results gained from the professional CFD software package FLUENT 5.5. Thus, the current implementation is correct, although, there are still many things to improve.

This results in the possibility to get a very good preview of the dynamics of combustion processes in a boiler. The students and developers of the combustion boilers could now test many configurations and modifications of the pulverized coal boilers interactively with an immediate response. Thus, our system could also be used for experimentations and education in the field of combustion processes. Currently, the system would be used in the educational process in the Faculty of Mechanical Engineering at the CTU Prague.

On the other hand, our fluid solver is conditionally stable as are the many other existing methods. Moreover, when requesting precise results, the professional CFD packages and software as well as some already existing fluid solvers based on the Navier-Stokes Equations would do a better job, but at the cost of the computation speed.

REFERENCES

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

[2] R. Fedkiw, J. Stam, and H. W. Jensen. Visual Simulation of Smoke, *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 2001, 15–22.

[3] J. Wejchert and D. Haumann. Animation Aerodynamics, *Proceedings of the 18th annual conference on Computer graphics and interactive techniques,* ACM press, 1991, 19-22.

[4] 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), 52–61, 1997.

[5] P. Witting. Computational Fluid Dynamics in a Traditional Animation Environment, *Proceedings of the 26th annual conference on Computer graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., 1999, 129-136.

[6] Y. Takai, K. Ecchu, N. Takai, A Cellular Automaton Model of Particle Motions and its Applications, *The Visual Computer, 11(5),* 1995, 240-252.

[7] J. Stam, L. Fiume., Depicting Fire and Other Gaseous Phenomena Using Diffusion Processes, *Proceedings of SIGGRAPH 95, Computer Graphics Proceedings, Annual Conference Series,* USA, 1995, 129-136.

[8] N. Foster and D. Metaxas, Realistic Animation of Liquids, *Graphical Models and Image Processing: GMIP*, 58(5), 471–483, 1997.

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

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

[11] J. Stam, Interacting with Smoke and Fire in Real Time, *Communications of the ACM*, 43(7), 76-83, 2000

[12] J. Warnatz and U.Maas and R.W. Dibble, *Combustion Physical and Chemical Fundamentals, Modeling and Simulation, Experiments, Pollutant Formation*, (Berlin:Springer-Verlag), 1995.

[13] Didier Arquès, Eric Felgines, Sylvain Michelin, Karine Zampieri – Thermal Convection in Turbulent Flow, Image Synthesis and Heat Animation, *Proceedings of WSCG*, Univ.of West Bohemia Press, Czech Republic, Plzen, 1999, 337-345.