FINAL REPORT

CONTRACT N° : IST-2000-29266

PROJECT N° : 29266

ACRONYM : VIRTUALFIRES

TITLE : Virtual Real Time Fire Emergency Simulator

PROJECT CO-ORDINATOR :
Graz University of Technology (TUG)
Lessingstrasse 25/II
A-8010 Graz Austria

PARTNERS :
Christian Doppler Laboratory for Applied Computational Thermofluiddynamics (CD), Austria
Fraunhofer Gesellschaft zur Förderung der Angewandten Forschung e.V. (FIGD); Germany
Kungl Tekniska Högskolan (KTH), Sweden
Lyon Turin Ferroviaire (LTF), France
Ministere de l´Equipement, du Logement et des Transports (METL), France
Stadt Dortmund, StA 37 Feuerwehr (FDDO), Germany
European Virtual Engineering, S.A. (EUVE), Spain

REPORTING PERIOD : FROM 2001-11-01 TO 2004-09-31

PROJECT START DATE : 2001-11-01 DURATION : 34 Months (extended)

Date of issue of this report : 2004-09-01

Project funded by the European Community under the 'Competitive and Sustainable Growth' Programme (1998-2002)

1 Table of contents

2 Executive publishable summary

A Virtual Reality Real Time Fire Emergency Simulator (VIRTUALFIRES) has been developed using techniques of virtual reality and computational fluid dynamics. In the simulator, the observer will be able to visualise the fire and smoke development and the transport of heat and toxic combustion products inside a tunnel and walk or run through the virtual structure in the same way as through a real tunnel. The simulator will use and access a database, which contains the results of three-dimensional transient combustion (Computational Fluid Dynamics - CFD) simulations for particular tunnel geometries with associated safety installations such as fans, fire extinguishers, exits etc. for particular fire hazard scenarios.

The system has been developed as a fixed installation in a CAVE virtual environment and as a portable installation using a PC and a head-mounted display (HMD). Two systems are available: one where the CFD simulation is pre-calculated, stored into a data base and then displayed another where it is carried out in parallel to the visualisation. In the first system the user will be able to move through the data but will not be able to change the characteristics of the simulation, for example the ventilation characteristics. In the second system the user may make specific changes (for example switching on/off existing fans, sprinklers) during a session. To the best of our knowledge this is the first system that has been developed that can perform real time simulations. For the simulation of long tunnels a special technique has been implemented to couple 1D and 3D CFD simulations, which is also a novel feature.

The VIRTUALFIRES system will be a unique system that can be used for assessing the fire safety of tunnels, for training of rescue personnel and for planning rescue scenarios and will be able to replace or supplement real fire tests. The end users of this system will be rescue organisations such as the fire brigade and police, tunnel operators and government organisations concerned about tunnel safety. The system can be used for making an objective assessment of the fire safety of existing European tunnels. It can also be used for training drivers on how to behave in the case of a fire emergency in a tunnel.

The exploitation of VIRTUALFIRES has the following aspects:

- **Consulting:** owners of tunnels may ask for an assessment of the fire safety of a tunnel, designers of tunnels may require a virtual safety check etc. Several such requests have already been made. Such services may be sold by consortium members

- **Sale of the system:** consultants, tunnel operators and fire brigades may see the benefit of purchasing the system for in house use.

3 Objectives and strategic aspects

The method of virtual reality has been applied to develop a simulator for assessing the fire safety of tunnels and for training rescue personnel. In a virtual computer-simulation-based environment all data about the structure, the safety equipment, ventilation, the fire and smoke development and vehicles/passengers exist in computer memory only. These data are displayed in such a way that the user has the possibility to study all the hazardous effects of a real fire emergency. The project therefore aims to contribute to efforts by the Community to increase the fire safety of European tunnels. It also aims to extend the range of application of virtual reality techniques.

In the Virtual Reality Real Time Fire Emergency Simulator (VIRTUALFIRES) simulator, the observer will be able to visualise the fire and smoke development and the transport of heat and toxic combustion products inside a tunnel and walk or run through the virtual structure in the same way as through a real tunnel.

A prototype of the VIRTUALFIRES simulator has been developed and evaluated, which can be used by government authorities, tunnel operators and rescue personal. Currently the only method available to assess the fire safety of tunnels is to perform real fire tests. The advantages of virtual reality based simulations are that they offer virtually unlimited scope, are economical (the only expense being the computer time) and are environmentally friendly because no toxic waste is produced. As a result of the project's activity, through the availability of VIRTUALFIRES, the fire safety of tunnels will be increased because all European tunnels can be tested for fire safety and virtual fire tests can be made obligatory for new tunnels.

The work involved the development of computer software, which together with specialised hardware can be used to display in a virtual reality environment the geometry of a tunnel with all its structural elements and safety installations together with the results of computer simulations of fires. The simulator will use and access a database, which contains the results of three-dimensional transient combustion (Computational Fluid Dynamics - CFD) simulations for particular geometries, hazard scenarios, etc. The system has been developed as a fixed installation in a CAVE virtual environment and as a portable installation using a PC and a head-mounted display (HMD).

The main applications of VIRTUALFIRES are:

Training of rescue personnel. Currently this can be only done with real fire test, which are expensive and produce toxic material.

Objective assessment of the fire safety of tunnels. This is currently either done using a list of rather arbitrary rules or by performing real fire/smoke tests (i.e. Mont Blanc Tunnel).

Intervention management. The tunnel operators will be able to train the intervention procedures in order to mitigate the effects of a fire emergency.

The simulator can also be used for:

Training of drivers. Using the simulator truck and other drivers can be trained on how to behave in a case of a fire accident.

4 Scientific and technical description of the results

4.1 General description

The main deliverable of the project is the Virtual Reality Real Time Fire Emergency Simulator (VIRTUALFIRES) simulator that can be used for assessing the fire safety of tunnels, for training of rescue personnel and for planning rescue scenarios and will be able to replace or supplement real fire tests. The simulator consists of software components and specialized hardware which allows a three-dimensional visualization of results of three-dimensional transient combustion (Computational Fluid Dynamics - CFD) simulations. The general layout is shown in Figure 1. The main components are the user interface, the CFD controller, the data manager, the CFD solver (ICE) and the visualization module. These will be described in more detail in the following sections.

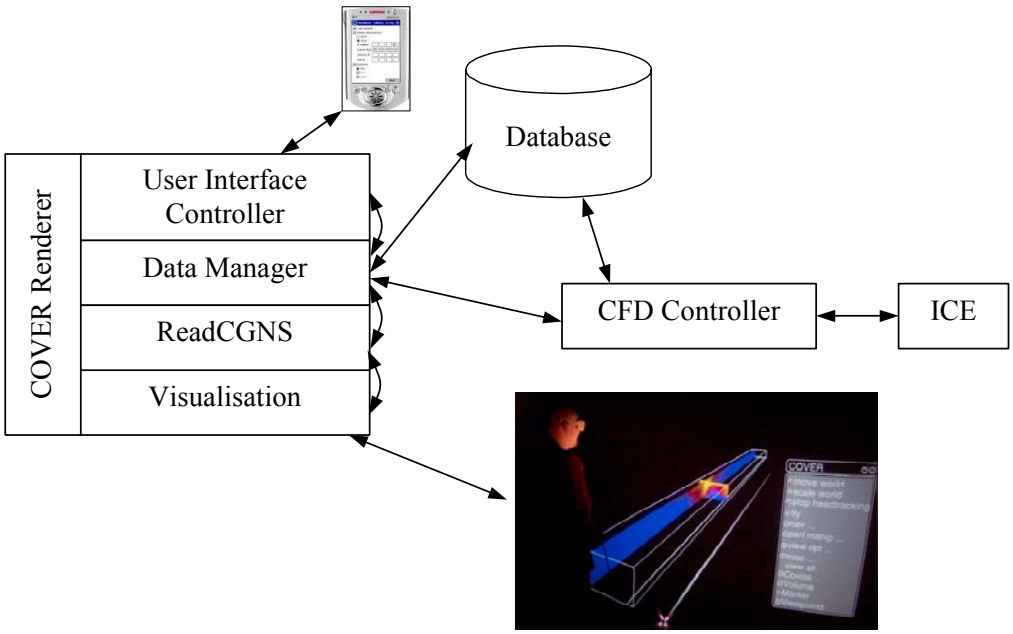


Figure 1: Diagram showing the components of VIRTUALFIRES

4.2 System capabilities

The VIRTUALFIRES system is ably to handle tunnels of any cross-section with a variety of installations. The installations can include:

- Fans
- Ventilation inlets and outlets
- Fire extinguishing nozzles
- Escape compartments, exits, lights etc.
- Cars, trucks or rolling stock

The shape of the tunnel cross-section is provided in a suitable format (AUTOCAD) and is used to generate the grid for the transient combustion calculation using ICE. With the currently available prototype this can, however, not be done automatically. The following additional input data are required:

- Fan capacity in ON position
- Atmospheric pressures at the tunnel portals
- Initial direction of the extinguishing nozzle
- Fire load

Once these data have been provided the user may :

- Define a mission and replay this mission using the forward/stop/rewind/start buttons on the graphic user interface (GUI, see Figure 2)
- During a concurrent session the user may switch on/off existing fans, fire extinguishers and restart a session

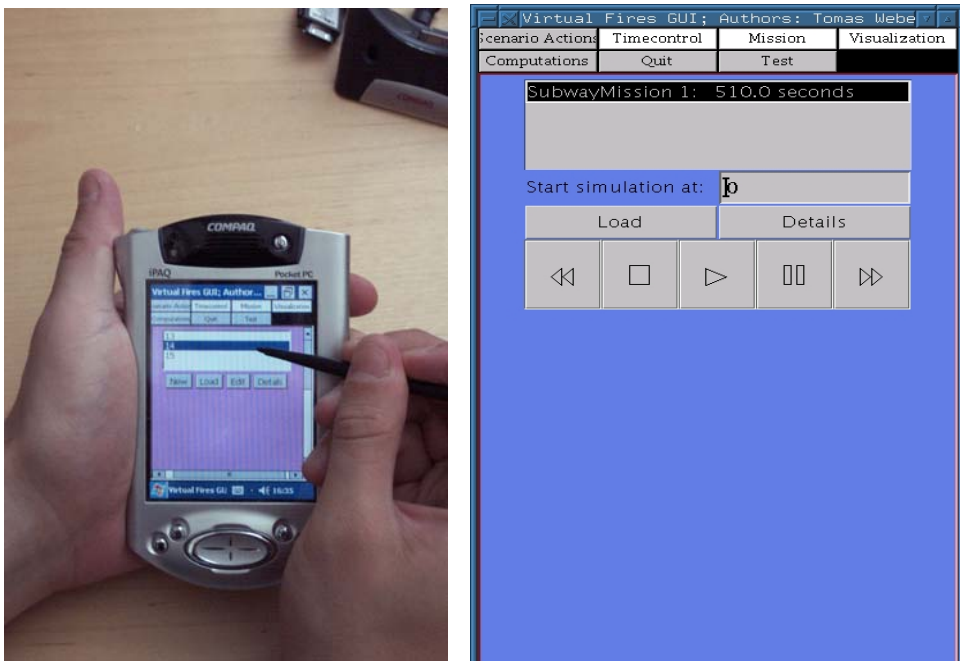


Figure 2: The GUI as presented on a PDA and a desktop, respectively

The following can be visualised:

- Smoke (using transparency values output by the CFD software) to check visibility of exist etc.
- Isosurfaces of temperature to check survivability
- Streamlines to check flow of smoke and efficiency of ventilation system

The different visualisation systems are shown in Figures 2-4.

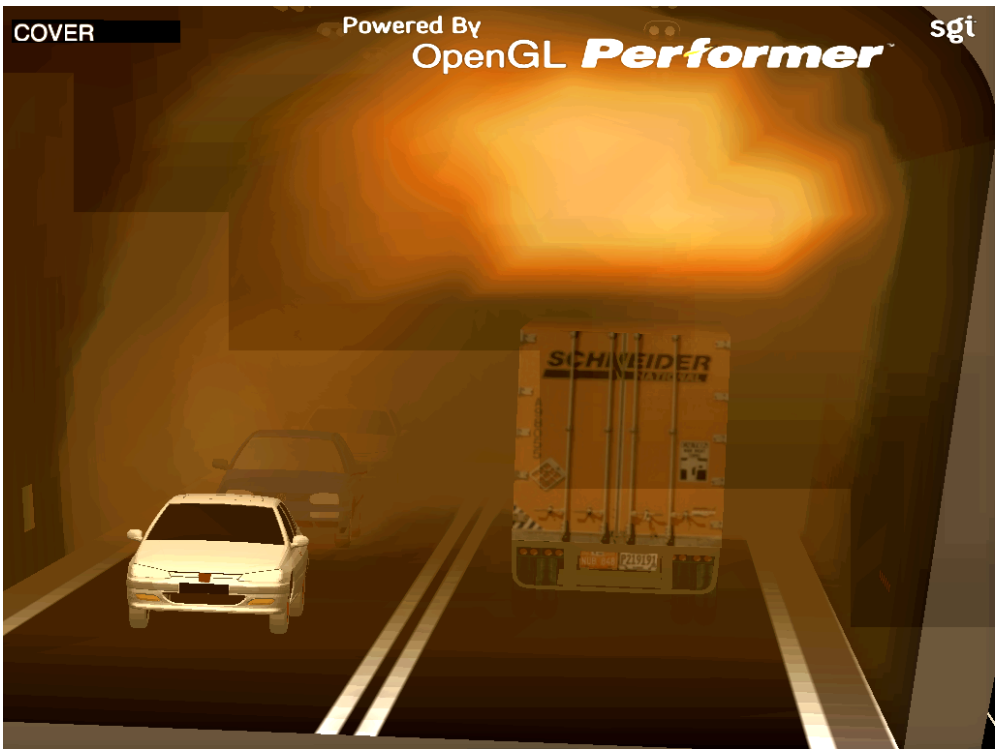


Figure 3: Visualisation of smoke

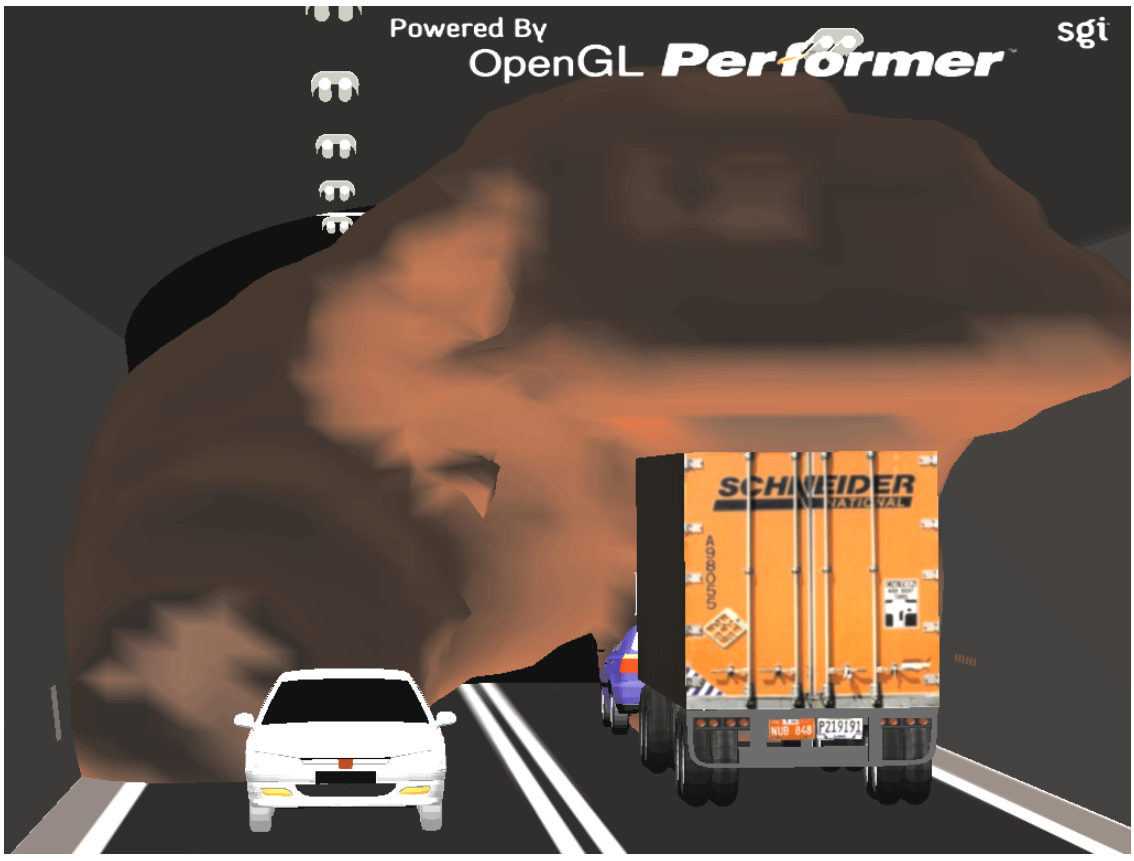


Figure 4: Isosurfaces

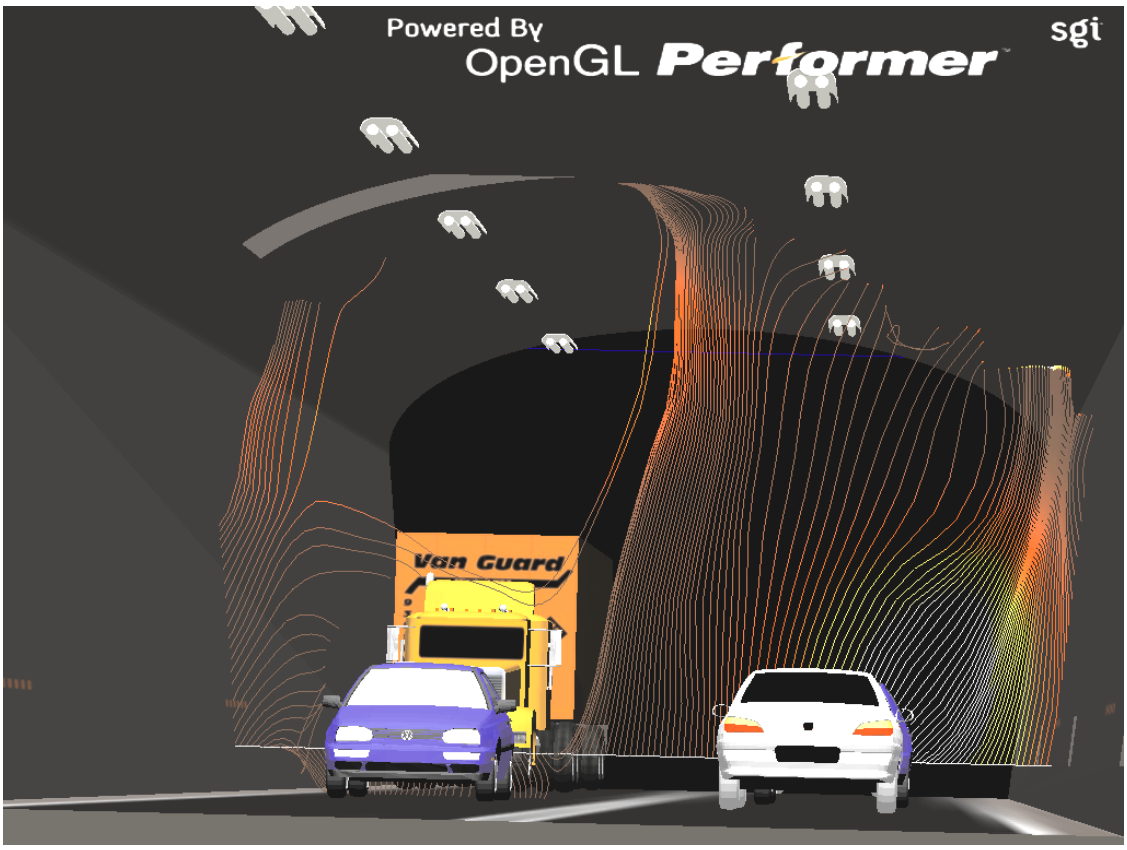


Figure 5: Streamlines

4.3 Data management

The datamanagement serves as the abstraction layer between the simulator and the underlying relational database system. It provides the services required for handling all data transfers between the distributed modules of the simulator and the database server.

It consists of 2 main modules:

1) the datamanager

2) the CFD-controller

The datamanager is the component providing all storage and retrieval related services for the simulator. It is built on top of the MySQL++ library for the connection to SQL-server and provides an object oriented API for the simulator.

Espacially it provides the following functionality:

- Large binary file transfer to/from the database server
- Storage and retrieval of all visualisation models
- Handling all CFD simulation related datasets
- Creating/modifying/deleting missions and their associated events

During the integration process the datamanager was built into a COVISE module first, but then was shifted into Cover PlugIns, namely from the DMC module to the UIC PlugIn.

The CFD-controller is a distributed component providing the functionality for steering the CFD solver on a remote machine. It uses the ACE communication framework for its command handling. This component can steer multiple instances of the serial version of the ICE solver, but is also able to steer the parallel MPI version of the solver on a cluster.

4.4 User interface

The functionality in the user interface is implemented with several components:

- The Graphical User Interface. This is the part of the system with which the user interacts and which initiates all other actions in the system.
- The User Interface Controller (UIC). This handles the communication between all other components.
- The geometry handler (geomHandler). This handles the placement and interaction with all graphical objects that are not strict data visualisation objects, i.e. the tunnel model, vehicles, fire-fighting equipment etc.
- Navigation in the CAVE with the PDA. A module in the PDA to simulate the standard Wand buttons.

4.4.1 GUI

The interface to control the interaction with the Virtual Fires system has been realized as a two-dimensional GUI with menus and buttons. This GUI can be used in a desktop environment as well as in a portable device; the graphic appearance is the same in both environments.

The portable device is a personal digital assistant; the chosen model is an HP iPAQ Pocket PC with embedded support for wireless LAN communication. A position tracker is attached to the PDA and therefore this device is used also for pointing and selection in the 3D space of the CAVE.

The GUI is implemented in Java. In the PDA environment we use the CrEme 3.25 Java-environment from NSIcom (http://www.nsicom.com/).

The GUI has been rewritten to a large extent since the first version. The functions of the user interface have been reworked in order to be more intuitive. The definition and editing of missions have been redesigned, as has the way the selection of visualisation methods is done. The functionality allows the user to:

The list of implemented functions contains:

- Loading pre-defined configurations and scenarios.
- Describing a mission.
- Starting, stopping and continuing a simulation.
- Adjust the speed of visualisation of a simulation, forwards and backwards

- Detaching from and attaching to an ongoing simulation.
- Stop an ongoing simulation
- Selecting alternate mission timelines and timesteps within them.
- Choosing visualization method.
- Navigating in the visualization.

4.4.2 UIC

The UIC is a COVER plugin. Its primary purpose is to receive commands from the GUI and forward them to the relevant receiving components, primarily the Data Manager Controller (see deliverable D3.5 for the details on the communication between UIC and DMC). The UIC has been continuously updated to match the communication requirements of all other components, so the new functionality in UIC is the communication functions for the new functionality in GUI, DMC, geomHandler and HVR. The ambition has been to keep little knowledge in the UIC code itself, but rather let the objects contain sufficient knowledge that processing can be as data-driven as possible.

4.4.3 geomHandler

As one of the aims of the VIRTUALFIRES project has been realistic representation of not only fire, but the actual environment where the fire occurs, a number of 3D models of tunnels, vehicles, fire-fighting equipment and so on have to be inserted in the visualisation. geomHandler is a COVER plugin, receiving commands from the UIC. For each simulation scenario the set of comprising objects is retrieved from the database and the geomHandler inserts them into the scene graph. The geomHandler is a new addition since the previous prototype and has been tested in a number of versions to find the most efficient way of handling the graphical data.

4.4.4 Navigation in the CAVE with the PDA

A module in the PDA has been implemented to simulate the standard Wand in the CAVE environment. You can now navigate completely with the PDA and use the buttons on the PDA for selection instead of the Wand buttons.

4.5 Visualisation of fire and smoke

For the VIRTUALFIRES project, IGD had the task to develop and implement a parallel visualization tool for the rendering of numerical simulation data in to a CAVE or a HMD. According to the different end users of the VIRTUALFIRES system, there where a number of different demands, which the rendering has to be able to do.

- Tunnel operators want the system to be capable of displaying the distribution of the temperature and the toxic gases. Furthermore they rely on a realistic rendering of the fire and the smoke in order to determine the visibility conditions in case of tunnel fire and to assess the validity of different evacuation plans.

- Fire-fighting personnel will use the system in a training scenario. Also here, a realistic rendering of fire and smoke is needed, because this is the only way to "embed" the training personnel into the scene. The also applies to a demonstration, where a scenario should be presented as impressive as possible.

- To be as flexible as possible a number of different scientific visualization methods must be developed which can be chosen according to the type of data actually displayed. Any visualization currently active can be assigned to any dataset read from the simulation output can be processed for rendering. All visualization parameters available for a given method must be changed interactively.

- Finally the rendering system must be scalable, i.e. it must adapt to a wide number of machines with a performance ranging from a laptop to a workstation driving the CAVE.

In the prototype the visualization system is embedded into a scenegraph system ("COVER"), which contains all geometrical and texture information necessary for the rendering. It natively provides different interfaces for rendering to the desktop or CAVE and HMD devices and supports a communication protocol for exchanging information between the different parts of the VIRTUALFIRES system.

4.5.1.1 The work done

The skeleton of the rendering engine developed by IGD is a parallel processing kernel, which assigns the available computation time to the different working procedures and a data source manager.

The data source manager is responsible for receiving the different datasets (containing temperature fields, velocity fields etc…) from the simulation part of the system. It transfers the data into a format suitable for efficient rendering and stores it as long as they are needed or – in case of a concurrent simulation – another dataset is available.

As the datasets are coming in different packages, it also is responsible for synchronizing these packages and releasing them for further use.

The actual visualization is done using different visualization methods which can be added one by one to the given system, without any interference to the others. In fact any visualization method developed "logs in" to the system, which collects all methods in a visualization method pool, which are now available for selection via the given interface. Aside from a number of parameters commonly shared by all visualization methods, the methods can have their own parameter sets, thus maximizing the flexibility while developing and adding new methods to the pool.

The probe is another main concept within the system. It encapsulates a field of the dataset – the data source, a visualization method and an area of interest. It can be seen as a bounding box, which can be moved freely throughout the data volume, rendering the field of the dataset given that lies inside the box with the visualization method selected. The probe concept enables us to the integrate different visualizations (even of different data) into a single image, thus allowing to analyze the correlations between them.

Interactivity is a crucial demand to the VIRTUALFIRES system as a whole. Correspondingly this has some implications to the design of the rendering engine. Certainly the most important of all is the use of efficient and parallelized rendering algorithms which are able to produce the geometry information, which is passed to the graphics hardware as fast as possible. Furthermore the assignment of computation resources done by the kernel is a key feature to the system. If

looked upon closely, actually any input from the user necessitates a new computation. Those input include :

- The moving of a probe
- The changing of the dataset (f.e. if a new simulation step has been computed and new data is available)
- The changing of the internal parameters of a visualization
- The movement of the user

Theoretically any new input can trigger an new computation, which could result in collapse of the performance if this is done too often, hence a scheduler collects all parameters belonging to a given probe and triggers a single computation after a appropriate time interval, if resources are free. Furthermore, the user has full control over the complexity of a visualization (usually measured by the number of geometry elements in the scene). While being able to trade off the complexity/accuracy of a scene the visualization can be adapted to virtually any systems performance. Additionally if a given hardware is capable of using multiple processors (via multi-threading), the computation of the visualization methods can be parallelized, without any overhead due to preprocessing or postprocessing work.

All these features enable us to control all aspects of the scientific or realistic visualization (short of the actual rendering, which depends on the power of the underlying graphics hardware) in a responding time acceptable by the user. Including the change of time as a (non–user) input, the user is able to see and analyze the development of the simulation, i.e. the development and spreading of the fire and smoke in the tunnel, the motion of (toxic) gases and visibility conditions in real time (i.e. as fast as the simulator runs).

For the VIRTUALFIRES prototype system, four different visualization methods have been developed and incorporated. Two of them are able to display vector field data (velocity fields in our case), the other two are able to display scalar field data (temperature, smoke/toxic gases density or combinations of both). For increased flexibility, only vector fields and scalar fields are accounted as different kinds of data – any visualization method can be assigned to any source field of the given type.

The actual visualization methods include

- Streamlines visualization
- Line integral convolution
- Isosurface extraction
- Realistic fire & smoke rendering

Streamlines and Line integral convolution both are scientific visualization methods for vector fields. They are suitable to display the flow of gases or particles through the simulation domain. The former uses geometry (linestrips), the latter uses noise, which is distorted by the flow field depicted on a intersection plane. Using appropriate coloring schemes, additional information can be carried through this visualizations: A selection of “color-maps” is available which map the sampled temperature or density onto the lines or the plane.

The extraction of an isosurface is a scientific visualization method which is restricted to the depiction of scalar fields. The algorithm creates a surface of points which have the same temperature or smoke density value. Hence the hazard regions (due to high temperature or toxicity) inside the tunnel can be easily identified. As the simulation propagates, the isosurfaces move on through the tunnel, affected by the air flow.

One common parameter in all four visualization methods is the sampling rate: In general, with the benefit of increased accuracy the sampling rate can be increased up to a maximum amount. Usually this creates lines and surfaces with "smoother" features, at the cost of more geometry information to be passed to the graphics hardware and a decreased rendering performance.

The last and most important visualization method is the realistic depiction of smoke and fire inside the tunnel – using so-called "volume-rendering". Since it also is the most complex method the approach and performance differs depending on the type of graphics hardware available. Basically it is an algorithm to render (semi-)transparent gases or fluids in a three-dimensional domain using 3D-textures. A 3D texture simply is a box type grid with a given resolution, where all scalar field data is mapped onto. An array of semitransparent planes, perpendicular to the viewing direction, is laid through the volume. Where a plane intersects the volume the texture is sampled onto the plane (i.e. the plane will be "(3d-)textured"), using an appropriate coloring scheme (using "color-maps", as mentioned above). As the planes are basically near-transparent, the viewer can look "behind" them. In fact, this algorithm integrates all sampling values along a ray shot from the eye-point of the user: If the volume contains regions with high smoke density, the resulting image is likely to be opaque when looking at these region, thus blocking the sight to anything behind. When looking at regions with clean air, the image simply will show the background of the scene (i.e. the tunnel walls).

The big problem with this approach is, that the resampling must be done even only if the spectator moves. In all other visualization methods the geometry does not depend on the viewing-position or –direction – one just moves around a rather "fixed" geometry. The consequence is that the geometry must be set up and rendered multiple times in a second. Depending on the capabilities of the graphics hardware, two options arise:

- The hardware is capable of 3D-texturing. In this case the texture and the planes is passed to the hardware (a "cheap" operation, since the texture is fixed anyway and planes can be defined with a few points) and the sampling and the integration is done in hardware, producing the final pixel color. This usually is fast enough to outweigh the problem, that the volume must be rendered multiple times.

- The hardware is not capable of 3D-texturing. In this case the sampling must be done in software, which may be quite expensive. More importantly, this operation may create large amounts of geometry as the planes must be tessellated using the sampling points. Unfortunately hardware not capable of 3D-texturing is unlikely to be able to set up and render large amounts of geometry (due to the extended setup time).

Both cases have been encountered, and both algorithms have been implemented and tested: The software solution currently runs on the SGI Onyx II, which is used to drive the CAVE engine and on the Linux based machines. Using a CAVE there is still another aspect to be considered: The algorithm as described above only displays planes which lie in front of the spectator. This is sufficient for desktop or HMD solutions. However, in the CAVE, the display covers up to six (rather than one) sides around the user. Consequently not only one array of planes should be processed but six ones, thus creating an array of nested semitransparent cubes, and increasing the workload sixfold.

For performance reasons, the cubes version has been tested but is not incorporated into the demo-version. As even the CAVE the user mostly focuses on the front side of the visualization, this is less of a problem as originally imagined.

In a still evolving prototype, we also tested the possibilities of a realistic depiction of fire and smoke, with enhancing features such as flames or smoke plumes. In a prototype using the hardware shader extensions of the latest graphics boards, we checked out the limits of 3D-texturing (currently working on windows based machine).

Shaders are programs which are executed in dedicated graphics hardware which define how the sampling coordinates and sampling results of one or more textures are combined to produce the final pixel color.

The features depicted with the volume-rendering algorithm can not be more detailed than the underlying computational grid, providing the data source for all visualizations. That the "fire" does not move between to time-steps of the simulation is even more undesirable. On the other hand, the grid resolution can not be increased arbitrarily. To model the appearance and the dynamics of the flames additional data has to be included. In our case, a noisy fractal distortion field is applied to the sampling grid prior to sampling. Noise usually generates a number of virtually unpredictable features, and the fractal guarantees that these features appear in different magnitudes of size, which gives an realistic impression as the flames are subject to turbulent flow. To avoid falsification, which makes the original data unusable, the distortion must be controlled by a factor which guarantees, that the distance between the original and the distorted sampling stays within the given bounds.

The visualizations include all features, which are demanded by the end-users. The development of tunnel fires can be analyzed and assessed and hazardous regions can be easily identified. The visualization of fire and smoke helps assessing the visibility conditions for both the fire-fighting personnel and the people enclosed in the tunnel and thus helps designing and validating the evacuation plans.

4.6 CFD solver

4.6.1 Description of ICE

4.6.1.1 About Lattice Boltzmann Methods

The Lattice Boltzmann Method (LBM) is a recent method for solving fluid flow problems based on kinetic theory. The basic idea is to solve a discretised form of the discrete Boltzmann equation. It can be shown that the solution of the Lattice Boltzmann equation (LBE) is equivalent to the solution of the Navier-Stokes equations within the limit of small Mach numbers. Compared to conventional Computational Fluid Dynamics (CFD) tools the LBM is in a relatively early rapidly developing stage.

There are some features, which makes the LBM a serious competitor to classical fluid mechanics methods [23]:

- Navier-Stokes solver need to treat the non-linear convective term, whereas in the Lattice Boltzmann Method convection is a simple advection of particle distribution functions to the next neighbor node.

- There is no need to solve a Poisson equation for pressure as it is locally obtained by an equation of state.

- As the LBM works on orthogonal equidistant grids and only involves communication with the next neighbors parallelisation can be done very effectively.

- Time consuming grid generation can be avoided for Lattice Boltzmann Methods by using a simple marker-and-cell approach1.

- As the LBM is explicit in time it is very well suited for unsteady simulations.

Unfortunately there are several drawbacks of Lattice Boltzmann Methods of which the most severe are the following [11]:

- The simple structure of the discretised velocity space makes it difficult to recover the energy equation. Most thermal LBM suffer from severe numerical instabilities. Furthermore almost all models use at least implicitly some kind of the Boussinesq approximation. One straightforward solution to overcome the problem of numerical instability is to consider temperature as a passive scalar. A survey of existing models can be found in [10].

- Due to the low Mach number expansion of the equilibrium distribution function in the LBM, it is limited to incompressible or nearly incompressible flows.

Looking at the shortcomings of the LBM mentioned at the beginning regarding the engergy equation and compressibility one may of course ask if it is the method of choice to simulate combustion processes. If one focuses on real time simulations there are actually some strong arguments in favour of the Lattice Boltzmann Method:

- As only regular rectilinear grids are used and only next neighbor communication is required code parallelisation can easily be achieved.

- The LBM as an explicit numerical scheme does not require iterations. This in turn means - as long as certain a-priori known stability criteria are taken into account - that divergence problems do not occur and the required computational effort and hence solution time to perform real time simulations can be precisely determined.

- The stress tensor required for the large eddy simulation model is locally available, therefore the computational work is significantly reduced compared to conventional CFD codes employing e. g. two-equation turbulence models.

- Furthermore due to the simple structure of the algorithm computational programming of the LBM is straightforward.

There were multiple objectives which have been solved during the course of the project:

- Development and implementation of a flow solver based on the Lattice Boltzmann Method and capable of simulating turbulent buoyant fluid flow problems.

---

[1]In the marker-and-cell approach the entire geometry is subdivided into voxel and each voxel belonging to the solid structure is set inactive. For highly complex structures like porous media the geometries may be obtained by computer tomographic methods and the computational grid can be constructed automatically by the aforementioned approach.

- Implementation and testing of combustion models suitable to simulate compartment and tunnel fires.

- Parallelisation of the solver in order to achieve real time simulation capabilities.

4.6.1.2 The LBE model used for Virtual Fires

The Lattice Boltzmann Equation (LBE) is a finite difference approximation to the discrete Boltzmann equation using a Bhatnagar-Gross-Krook (BGK) model for the collision term.

For the derivation and theoretical background the interested reader is referred to the excellent reviews published by Luo [11] and Chen and Doolen [3]. A lot of information can also be found in the recent book by Succi [20].

In what follows the notation introduced by Qian [16] is used. The term dXqY means a model in 3-dimensional space using 19 discrete lattice vectors.

MPICE uses a multiple relaxation time (MRT) formulation of the lattice Boltzmann equation. The basic idea is to perform the translation step within the velocity space and the relaxation within the moment space. The mapping between the velocity and moment space is done by a simple transformation operation. The single relaxation time is replaced by a diagonal matrix containing one relaxation parameter for each hydrodynamic mode. An overview about the MRT model can be found in [4].

The evolution equation for a particle distribution function on a lattice consisting of discrete lattice vectors is given by

$$\left| f_\alpha (x_i + e_{\alpha,} t + 1) \right\rangle = \left| f_\alpha (x_{i,} t) \right\rangle - \mathbf{M}^{-1}\mathbf{S}[\left| m_\alpha (x_{i,} t) \right\rangle - \left| m_\alpha^{(eq)} (x_{i,} t) \right\rangle] + \mathbf{F} \qquad \text{(Eq. 1)}$$

where $\boldsymbol{f_\alpha}$ is the particle distribution function related to the lattice vector $\mathbf{e_\alpha}$, "(eq)" denotes the equilibrium particle distribution, **S** is the diagonal relaxation matrix and **F** is the body force term due to buoyancy.

The relaxation matrix is given by

$$\mathbf{S} = diag(s_0, s_1, \ldots s_{ndir-1}) \qquad \text{(Eq. 2)}$$

The moments are related to the density (ρ), momentum (**j**), momentum flux (**q**), energy (**e**), kinetic energy squared (ε), elements of the stress tensor (**p**, π) and there are some moments without obvious physical meaning (**m**) :

$$\left| m \right\rangle = (\rho, e, \varepsilon, j_x, q_x, j_y, q_y, j_z, q_{z,} 3p_{xx,} 3\pi_{xx}, p_{ww}, \pi_{ww}, p_{xy,} p_{yz,} p_{xz,} m_x, m_y, m_z)^T \qquad \text{(Eq. 3)}$$

Note that the equation above is in dimensionless form. The lattice structure for the D3Q19 model used in **MPICE** is depicted in Fig. 6 [12].

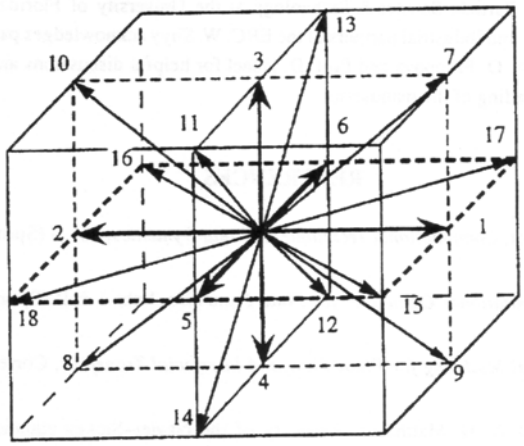


Figure 6: D3Q19 model

MPICE uses a Large Eddy Simulation (LES) model to simulate turbulent flows. The basic idea of a LES is to resolve only the largest unsteady turbulent motions. The role of the small eddies is limited to give the satisfactory role of dissipation that it is concentrated at small scales. The large energy carrying length scales of turbulence are very problem dependent, whereas the small scales are assumed to be more universal. Detailed information about the LES model used for the Virtual Fires project can be found in *Deliverable 5.5 - Software User Guid V2.0* [17].

TRANSPORT EQUATIONS FOR SCALAR QUANTITIES IN THE LBGK FRAMEWORK

The transport equations for scalar quantities, e.g. smoke, are solved within the framework of the LBGK method by introducing an additional distribution function for each quantity.

The particle distribution functions for the scalar quantities evolve on the same lattice than the pressure distribution function for the fluid.

The LBE for the transport of species is given by

$$\mathbf{g}_i(\mathbf{x}+\mathbf{e}_i,t+1)=\mathbf{g}_i(\mathbf{x},t)+\frac{1}{\tau}\left[\mathbf{g}_i^{eq}(\mathbf{x},t)-\mathbf{g}_i(\mathbf{x},t)\right]+\mathbf{Q} \qquad \text{(Eq. 4)}$$

In the above equation **Q** is the volumetric heat source term. The equilibrium distribution function is given by

$$\mathbf{g_i^{eq}}=\mathbf{t_i}\phi\left(1+\frac{\mathbf{e}_{i\alpha}\mathbf{u}_\alpha}{\mathbf{c_s^2}}+\frac{\mathbf{u}_\alpha\mathbf{u}_\beta}{2\mathbf{c_s^2}}\left(\frac{\mathbf{e}_{i\alpha}\mathbf{e}_{i\alpha}}{\mathbf{c_s^2}}-\delta_{\alpha\beta}\right)\right) \qquad \text{(Eq. 5)}$$

4.6.1.3 Representation of the fire

MPICE supports two different model of fire representation

- Volumetric Heat Source Model (VHS)

- Simple Chemically Reacting System (SCRS)

4.6.1.4 Volumetric Heat Source Model

In the VHS model the fire is represented as a source of smoke and energy in a pre-defined region. This model is widely used for simulating fires in enclosures. The evolution of smoke and energy is governed by the transport equation for a scalar species with source terms in pre-defined regions. The convective transport is calculated using the velocity of the carrier fluid.

In **MPICE** it is possible to use pre-defined or user-defined smoke and energy emission charts like the one in Fig. 7 [2]. The energy and some emission charts can be described by defining the initial slope (increasing fire load), the final slope (decreasing fire load) and the overall duration of the fire.

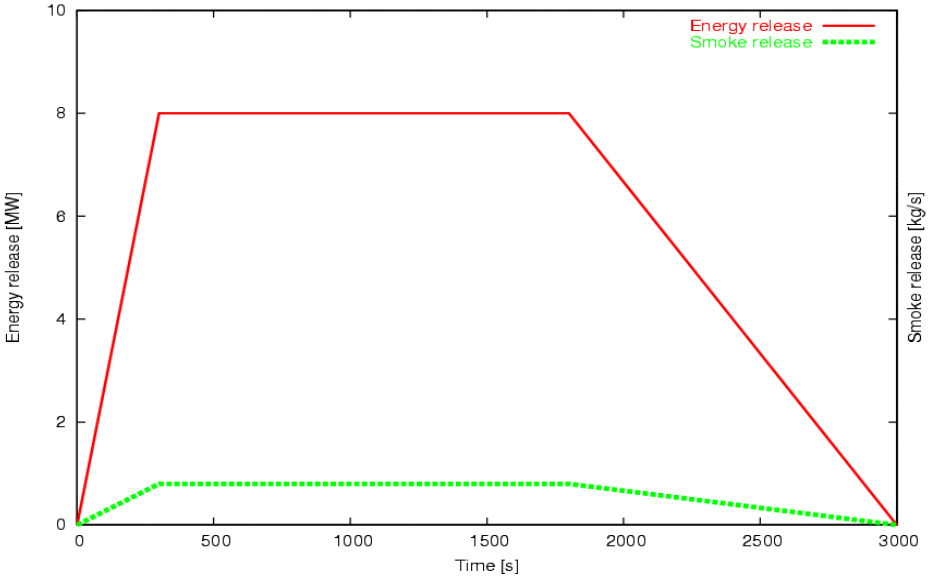


Figure 7: Energy and smoke release rates

As the density difference outside the pre-defined fire region may be relatively small the Boussinesq approximation is assumed to be justifiable.

4.6.1.5 Simple Chemically Reacting System

Real combustion phenomena involve a large number of chemical reactions, but their overall effect can often be described by very simple means: Fuel and oxidizer react to combustion products and due to the chemical reaction heat is released. Under the assumption that the chemical species react as fast as they mix, all that remain is to compute how fuel mixes with oxidizer.

Spalding [19] gives a definition of the simple chemically reacting system (SCRS):

- The simple chemically reacting system involves a reaction between two reactants (fuel and oxidant) in which these combine, in fixed proportions by mass, to produce a unique product:

1 [kg] fuel + s [kg] oxidant → (1 + s) [kg] product (Eq. 6)

- The specific heats of all mixture components (fuel, oxidant, product, diluent) are equal and independent of temperature. This assumption is not necessary, but justified as there are major simplifications already made in the reaction chemistry.

- All transport coefficients are equal at any point in the mixture, however they need not to be uniform.

An often presented example is the combustion of methane which is modelled as the following overall one-step reaction:

$$CH_4 + 2\ O_2 \leftrightarrow CO_2 + 2\ H_2O \qquad \text{(Eq. 7)}$$

The implication of the above is that all intermediate species are not considered in the combustion reaction.

The right hand side of eq.(7) is considered as one single product. The nitrogen of the air is supposed to be a simple inert species which does not enter in the reaction equation.

Using the equality and temperature independence of the specific heats of all mixture components and assuming fast chemistry a new variable, the mixture fraction *f*, can eventually be introduced.

After solving for the mixture fraction all other important scalar quantities of interest can be derived from it without solving individual transport equations. The dependence of these variables on the mixture fraction is depicted in fig. 8.

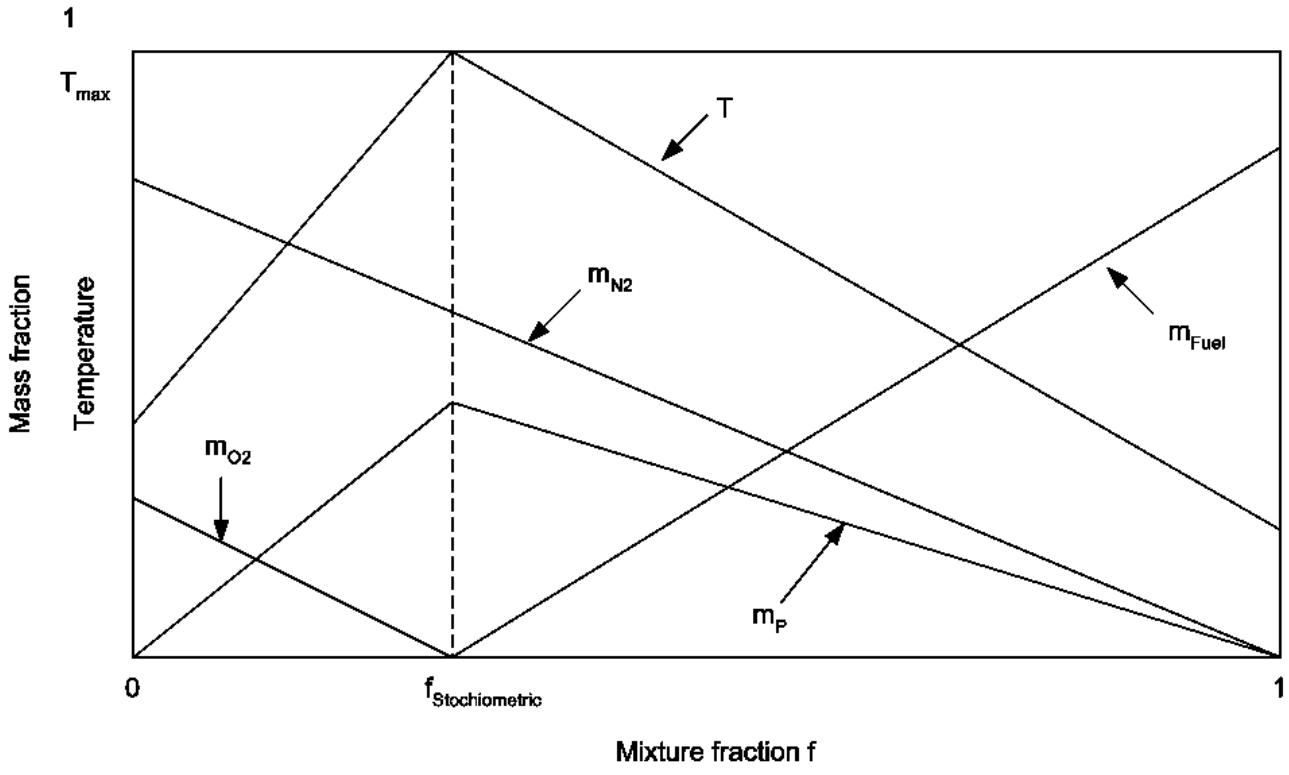


Figure 8: The Simple Chemically Reacting System

In turbulent flows not only the instantaneous filtered values but also the sub grid scale contributions influence the combustion process. Therefore strictly speaking another quantity accounting for these effects should be considered (e.g. mixture fraction variance). This will result in a non-linear deviation from the above depicted linear relationships. However in the present work this non-linear behaviour is neglected for keeping the computational models simple and suited for real time simulations.

The same energy and smoke emission charts as for the VHS model can be used. Contrary to the VHS model the transport equation for the mixture fraction is free of source terms. The fire is modelled by an inlet stream which only contains fuel (i. e. the mixture fraction in this inlet stream is equal one). The heat release rate is therefore determined by the fuel mass flow through the inlet and the calorific value of the fuel.

4.6.2 Concurrent CFD

4.6.2.1 *Why Parallelisation?*

Computer programs are usually sequential meaning that their execution is done by only one processor. The computational time may vary extremely from a few seconds up to several weeks depending on problem size. Therefore speeding up the computation is of particular interest. One can rely on the steady increasing performance of processor technology or try to parallelise computer codes. The idea of parallelisation is to spread the work load over several processors and therefore speed up computation.

There mainly exist two classes of parallel programming paradigms:

- The shared memory paradigm consists of sharing data through a common memory by using compilation directives. This paradigm allows to use parallel machines without major changes to the sequential code, but becomes inefficient for larger numbers of involved processors or in inhomogenous environments.

- The distributed memory paradigm consists in distributing the data to the processors to share the work load. Processors requiring information located on another processor have to communicate through messages. As the messages are sent over a network connecting the processors the amount of communication should be minimal. The distributed memory model requires much more programming effort but leads to more efficient codes and can be even used for cluster solutions.

More in depth information about parallel programming can be found e. g. in Foster [5] and Pacheco [15].

4.6.2.2 Selected Programming Model

A distributed memory approach is used for the parallelisation of the flow solver. Distributed memory systems are characterised by a high scalability and the large physical memory available. The execution performance is influenced by the balance between CPU speed and network speed.

In the MPMD (multiple program - multiple data) programming model each processor executes its own program. The communication between the processors is performed by sending and receiving messages.

Although it is probably the most difficult approach to parallel programming it is selected due to the following reasons:

- It promises the highest performance on distributed memory systems with a large number of processors.

- It is the most portable approach.

- The programmer has all freedom to optimize communication

There are a few message passing libraries available, which contain functions for sending and receiving messages. The Message Passing Interface (MPI), the de facto standard, is used for portability. A full description of the MPI standard is given in [18] and [21]. Up-to-date information can be found on the website of the MPI forum [6].

4.6.2.3 Domain Decomposition

There exist a number of possibilities to decompose the computational domain into smaller sub domains. For strictly reagular computational meshes as used within the Virtual Fires project two very simple methods promise the best performance due to a minimum amount of communication between the processors.

The simplest one is a one dimensional decomposition (ODD) which works along the direction of longest extent of the domain. This method works for any number of processors but will lead to bad surface to volume ratios2 for higher number of processors as can be inferred from fig. 9.

Alternatively for the recursive co-ordinate bisection method (RCB) the number of sub domains must be a power of 2, but in most cases it leads to better topologies of the sub domains (fig. 10).

---

[2]The term surface to volume ratio characterises the number of computational mesh cells located on a inter-processor boundary to the number of computational mesh cells allocated to an individual processor. Information located on inter-processor boundary cells must be communicated to the corresponding neighbor processor. Obviously the higher the number of inter-processor boundary cells the higher the communication effort and therefore the total performance decreases.

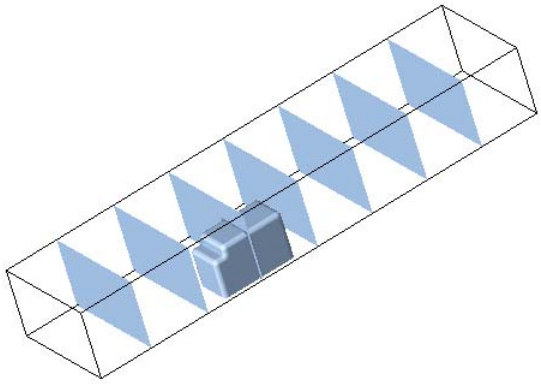


Figure 9: One-dimensional domain decomposition

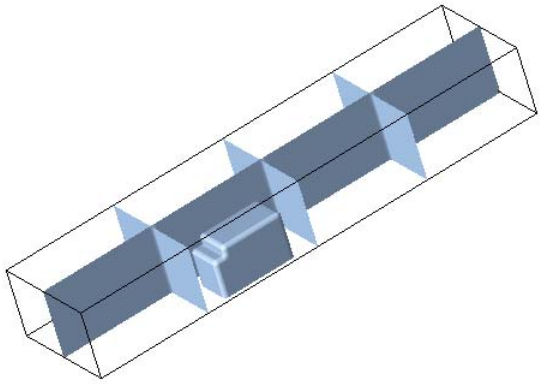


Figure 10: Recursive co-ordinate bisection method

4.6.2.4 Parallelisation Strategy

The parallelisation is done in a way that only particle distribution functions are communicated between the involved processors. The hydrodynamic variables which are available locally on each processor are collected by the root processor only on demand (e.g. for writing a result or restart file). Fig. 11 shows a scheme of the parallel algorithm. The parts which are executed in parallel are colored red, parts which requires serial execution are colored blue.

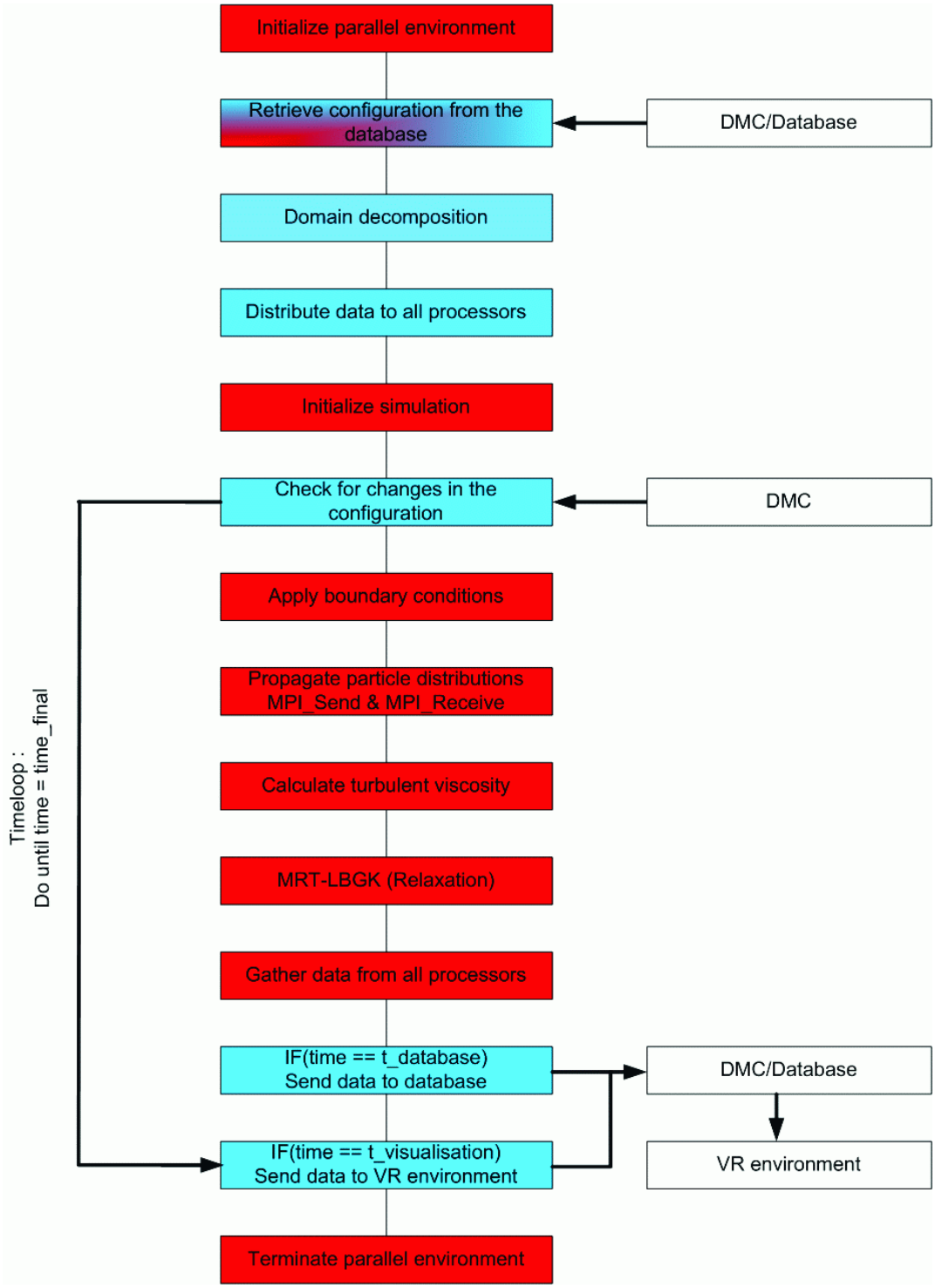


Figure 11. Structure of MPIce and communication with the data manager client (DMC)

4.6.2.5 Available Hardware

At the Parallel Dator Centrum at the Kungl Tekniska Högskolan in Stockholm, Sweden [8], the HP Itanium 2 cluster "Lucidor" is used for the Virtual Fires project. "Lucidor" is made of 90 nodes each equipped with two 900 [MHz] Itanium 2 "McKinley" processors. The theoretical peak performance (TPP) of each node is about 7.2 [GFLOP/s] resulting in a total TPP of 648

[GFLOP/s]. The processors are interconnected via Myrinet [7] which is a full-duplex 2+2 [GBit/s] network connection.

4.6.2.6 Parallel Performance of the Linux cluster Lucidor

The following performance data were obtained by measuring the calculation time of the parallel LBE code in a "normal use mode", i. e. the standard I/O, which is one entire data set per second real time and some monitoring, is included. The hydrodynamic variables (i.e. pressure and velocity) and two scalar quantities (which can be for example mixture fraction and temperature or mixture fraction and some marker for an extinguishing agent) were calculated. The initial set up and the domain decomposition is not considered.

The computational domain consists of a simple straight channel with square cross section. The simulations were carried out for two typical domain sizes used for the Virtual Fires simulator. The smaller one consists of 75,000 cells (Testcase A), which is the target domain size for the real time compuations, the other one consisting of 240,000 cells (Testcase B). Additional one test case made up of 2,000,000 cells is considered (Testcase C).

For all configurations five simulations calculating 5000 time steps, which corresponds to 100 [s] real time, were carried out and the necessary wall clock time were averaged afterwards. Below the computation time needed for 1 [s] real time is given.

MPICH [9] is a freely available implementation of the MPI standard and was used as communication library on the HP Itanium cluster. The optimisation level used for the 64 bit Intel Fortran compiler `efc` was `-O2`.

	Number of processors				
Testcase	1	2	4	6	8
A	21.11[s]	10.05[s]	4.91[s]	3.25[s]	2.27[s]
B	80.69[s]	40.34[s]	17.93[s]	12.04[s]	9.07[s]
C	683.47[s]	310.67[s]	151.88[s]	103.56[s]	80.41[s]

	Number of processors				
Testcase	12	16	24	32	64
A	1.49[s]	1.28[s]	1.05[s]	1.07[s]	1.13[s]
B	5.93[s]	4.98[s]	3.43[s]	2.68[s]	2.06[s]
C	52.57[s]	41.68[s]	28.72[s]	21.56[s]	11.93[s]

Tab. 1: Lucidor - Computation time

	Number of processors									
Testcase	1	2	4	6	8	12	16	24	32	64
A	1.0	2.1	4.3	6.5	9.3	14.2	16.5	20.1	19.8	18.7
B	1.0	2.0	4.5	6.7	8.9	13.6	16.2	23.5	30.1	39.2
C	1.0	2.2	4.5	6.6	8.5	13.0	16.4	23.8	31.7	57.3

Tab. 2: Lucidor - Speed Up

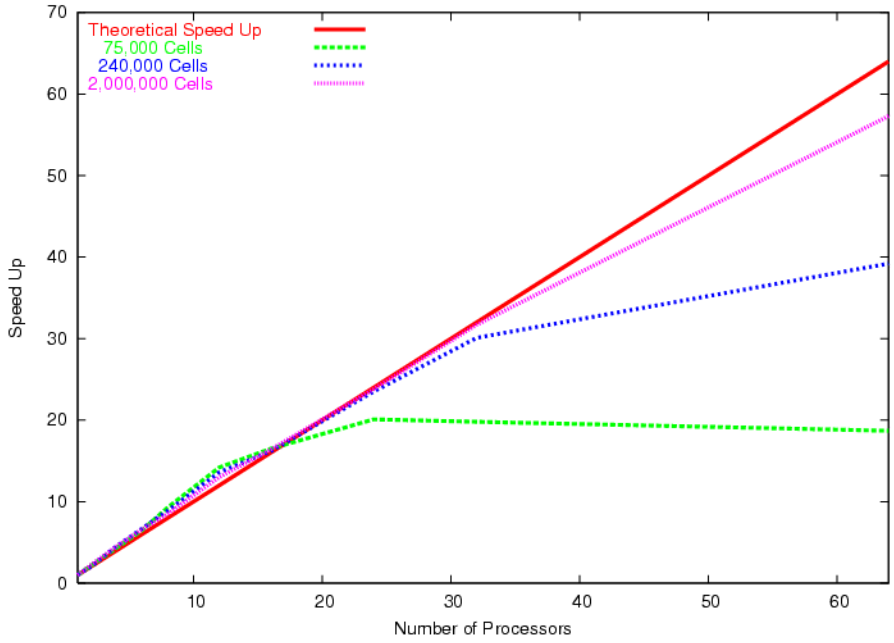


Figure 12: Lucidor: Parallel Speed Up

The parallel speed up for Testcase A is better than the theoretical value for up to 16 processors, but breaks down dramatically for higher numbers of processors. This is not astonishing if one takes into account that the number of cells for each processor is only about 1200 if 64 processors are used. The one way data rate between the processors is about 1.1 [GBit/s], which is about 55 [%] of the theoretical value of the Myrinet connection. Additionally the network latency sums up to 2 [ms], not to forget one collective data movement per second where all processors send their data to the root processor for I/O.

For higher numbers of processors the performance becomes better if larger domains are used. For Testcase C consisting of 2,000,000 cells the speed up is only about 10 [%] below the theoretical value for 64 processors.

The performance figures for Testcase A consisting of 75,000 cells show that a real time simulation is possible using about 20 processors. For this configuration one second real time event takes roughly one second CPU time. Judging this figures one must keep in mind that the I/O is included in the measured CPU time and two scalar species are considered.

Even for Testcase B consisting of 240,000 cells a real time simulation seems to be feasible in the near future as for 64 processors only about 2 seconds CPU time are required for one second real time.

4.6.2.7 Discussion

To judge the above results in view of real time simulation a few things have to be considered:

- Real time simulation in connection with real time visualisation only make sense if a suitable number of data sets can be displayed in a very short time. Ideally this would be about 25 data sets per second. In reality one data set which can be transfered to a visualisation system and displayed is assumed to be sufficient. In practice it turned out that at the moment one data set per second is the maximum which can be transfered from the computing machine to the visualisation device via a Gigabit connection.

- From physics and numerics one can conclude that the maximum timestep for stable simulations is in the range of 1/50 [s]. Therefore 50 timesteps per second real time must be calculated as a minimum.

- The number of computational cells which can be calculated in real time is rather small at present time (about 60,000 to 75,000 cells), but in view of the obtained performance data real time simulations consisting of 250,000 cells are feasible in the very next future.

One can summarize that it is possible to successfully perform real time simulations using MPICE.

4.6.3 1D/3D coupling

The idea of coupling between 3D and 1D models is based on the attempt to use benefits of the both models : the 3D model should be used near the source of fire where the flow is turbulent and the detailed description is needed, while the 1D model should be enough to describe the flow far from the fire.

The lattice Boltzmann method (LBM) is used to compute the fluid flow in the 3D area. As this method is already described in a previous section of this report, the concentration will be on the 1D model as well as the coupling between the two models in the following.

4.6.3.1 1D model – Navier-Stokes equations

In order to obtain the 1D model, only averaged values of all quantities over the cross-section area of the tunnel are considered i.e. all the variations over the cross-section are neglected. The Navier-Stokes equations are employed to obtain density and momentum while the transport equation is used to compute the smoke distribution.

Further details are given in Appendix A1.

4.6.3.2 3D/1D coupling

At each time step, the inner boundaries of the 3D and 1D model have to be updated. These update procedures have to be well defined in order for the coupled model to function properly.

The fact that both the 3D and the 1D models are hyperbolic systems of equations gives us the natural choice for inner boundary updates: the characteristic values. The Navier-Stokes equations are treated separately from the transport equation.

For more details see Appendix A2.

4.6.3.3 3D/1D solver - the algorithm

The coupled 3D/1D solver originates from a 3D solver based on the lattice Boltzmann method. The 3D computations in the 3D/1D solver have remained unchanged while additional features have been added as 1D computations, updating of the inner boundaries, grid expansion and grid reductionThe most distinguishing characteristics of the 3D/1D solver can be placed into two different categories: grid expansion and grid reduction.

4.6.3.4 3D grid expansion

In coupled 3D/1D models, the most interesting part of the flow should be simulated in the 3D area i.e. the turbulent flow as well as the flow affected by fans, vents, etc. However, the dynamic nature of the flow makes it very difficult to predict the size of this area in advance. On the other hand, the 1D model is well-defined as long as the variations of the field values over the cross-section of the tunnel are small. When these variations grow the flow looses its "one-dimensional" nature and cannot be well approximated by such a model.

These two requirements, large enough 3D area and "one-dimensionality" of the flow in the 1D area, are satisfied by the grid expansion/reduction model.

At each time step, the variation of the field values is checked at the inner 3D boundaries. If this variation is large the 3D grid is expanded whilst the 1D grid is reduced. Again Appendix A3 provides more details.

4.6.3.5 3D grid reduction

As the 3D grid can be expanded, it can be reduced as well in the 3D/1D solver. A typical scenario for the grid reduction can be the following: In the beginning of the simulation the flow of species is moving in one direction thus causing the 3D grid to be expanded once or several times. At a specified time the flow is changing direction which can result in a decreased concentration of smoke in the outermost generated grids. With time this concentration in the outermost grid can be too low. In that case the grid extension has no purpose anymore and can be disposed.

At specified time intervals the cross-section variation of the smoke concentration at the 3D grid extension neighboring to the outermost grid extension is checked . If the variation is too low then the 3D grid is reduced. In this case the outermost 3D grid extension is disposed and the 1D grid is expanded. The initial field values on the new part of the 1D grid are the averaged field values from the disposed 3D grid extension.

All 3D grids can be disposed except the initial one. Thus the initial 3D grid is the smallest possible 3D area in the coupled 3D/1D solver.

4.6.3.6 Test case

Several tests have been performed where the 3D/1D model is compared to the 3D model only. In this article, the scenario with a burning bus in a short section of the Gleinalm tunnel is shown.

In this scenario, a 100 meter long section of the Gleinalm tunnel is modeled. Near the center of this section a source of fire is placed in a 12 meter long bus. Except the bus, two cars are placed in the tunnel. At the top of the tunnel, three vents are modeled: one for fresh air inlet and two for exhaust air outlet.

At the beginning of the simulation the bus starts to burn. The smoke distribution is simulated with both models under 100 seconds. The outcome of the simulation is shown in the following figure after 8 seconds, 20 seconds and 100 seconds. Here, the cross-section averaged values of smoke concentration are compared.

As we can see, after 8 seconds there is no difference between the two models. The 3D area is still equal to the initial 3D area. The flow of smoke has slowly started to move to the left part of the tunnel. After 20 seconds, the 3D grid has been expanded to the left. Also here, almost no difference can be seen between the two models. In the lower graph we see the obtained results at the end of the simulation. Here we can observe that no drastic change in the smoke distribution has happened in the last 80 seconds. A good agreement between the two models can be seen as in the previous two graphs.

The results in this test show similar results obtained in the other tests where the coupled 3D/1D model is compared to the 3D model only. That is, the agreement between the models is quite good.

4.6.3.7 Conclusion

The coupled 3D/1D model has been developed to replace the 3D model in long tunnels. Also in the tunnels of moderate size, the 3D/1D model should be much more efficient and much cheaper then the 3D model.

The test included in this article, as well as some other performed tests, shows that the agreement between the two models is quite good in short tunnels. The smoke distribution, but also the density, velocity and temperature distributions in the 3D model are well approximated with the 3D/1D model. The coupling between the 3D and 1D areas woks well as long as the cross-section variation of the field values on the 3D boundary cells is small. This is, more or less, insured by the grid expansion/reduction model.

However, no real conclusion can be made before the 3D/1D model is tested and validated in long tunnels. Even though the obtained results seem promising.

4.6.4 Validation

4.6.4.1 General Remarks

The following simulations are carried out using the MRT LBM with Smagorinsky type subgrid scale model described above. For the VHS model the energy source terms are equally distributed over a distinct volume, whereas for the mixture fraction model the fire is modelled by an fuel inlet stream. The combustion of methane is assumed for all cases and therefore the methane inlet mass flow is choosen to match the required heat release rate.

Unless otherwise noted the Smagorinsky constant is $C_s = 0.13$ and the turbulent Prandtl number is equal 0.2. The walls are always assumed to be adiabatic.

4.6.4.2 Tunnel Fire Experiment

A model tunnel fire experiment in a small scale tunnel was simulated by Xue et al. [22]. Fig. 13 shows a sketch of the experimental set up. The main test section is 6 [m] long with a rectangular cross section of 0.3 [m] height and 0.9 [m] width. Unfortunately the material of which the walls are made of is not exactly known, but a major section of the tunnel consists of perspex, whereas in the vicinity of the fire source silica glass and aluminium are used. The fire source is located 1.5 [m] from the inlet of the test section at the center of the tunnel floor. The burner area is 0.18 [m] x 0.15 [m]. An adjustable fan is used to induce a longitudinal velocity. The heat release rate was 3.15 [kW] and a longitudinal flow velocity of 0.13 [m/s] is present. Temperatures are reported for 3 cross sections, which are located 0.9 [m] upstream, 3.3 [m] and 5.1 [m] downstream, respectively.

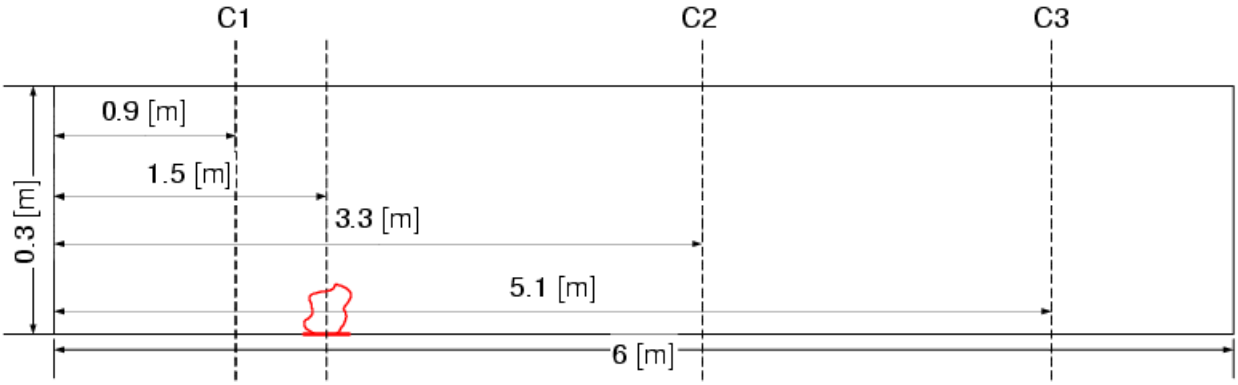


Figure 13: Computational Model

The computational grid used in the simulation consists of 300 x 45 x 15 cells. A Smagorinsky constant of 0.14 is used and the turbulent Prandtl number is 0.2. The longitudinal ventilation is accomplished by applying a velocity boundary condition at the left boundary of the computational domain, whereas a constant pressure boundary condition is used on the other side. All walls are assumed to be adiabatic.

4.6.4.3 Results

At station C1 (fig. 14), which is the one nearest to the fire source, all models including the VHS model used by Xue et al. [22] perform very poor. In the experiment the temperature is

continuously increasing, whereas all simulations predict a nearly constant temperature above a height of 0.15 [m]. The temperature obtained by the LBE combustion models is above the reference solution. This can be explained by the assumption of adiabatic walls in the LBE calculations, whereas Xue et al. [22] applied conducting wall boundary conditions.

The experimental trend is even not very well captured at station C2 (fig. 15). Again the LBE combustion models over predict the temperature compared to the reference solution, but in most regions the difference is small. The temperature just below the ceiling is somewhat higher in the experiment than it is obtained by the LBE mixture fraction model.

At measuring station C3 (fig. 16) furthermost from the fire the LBE VHS model corresponds quite well to the experimental data up to a height of 0.2 [m]. Also the LBE mixture fraction model agrees with the temperature from the experiment up to 0.1 [m] height and over predicts the temperature beyond. In the experiment the temperature below the ceiling is decreasing. This is not reproduced in the LBE simulations due to the adiabatic walls. In general both LBE combustion model perform quite well some distance away from the fire. This is not surprising as the error induced by using the Boussinesq approximation is largest in near fire regions where temperature differences are considerable.

It is interesting to note that the simulations done by Xue et al. \cite{FSJ_Xue_Comparison_2001} which do not use the Boussinesq approximation do not yield much better results compared to the LBE calculations at the stations near the fire source. This indicates that the complex interaction between air flow and fire is not modelled adequately.

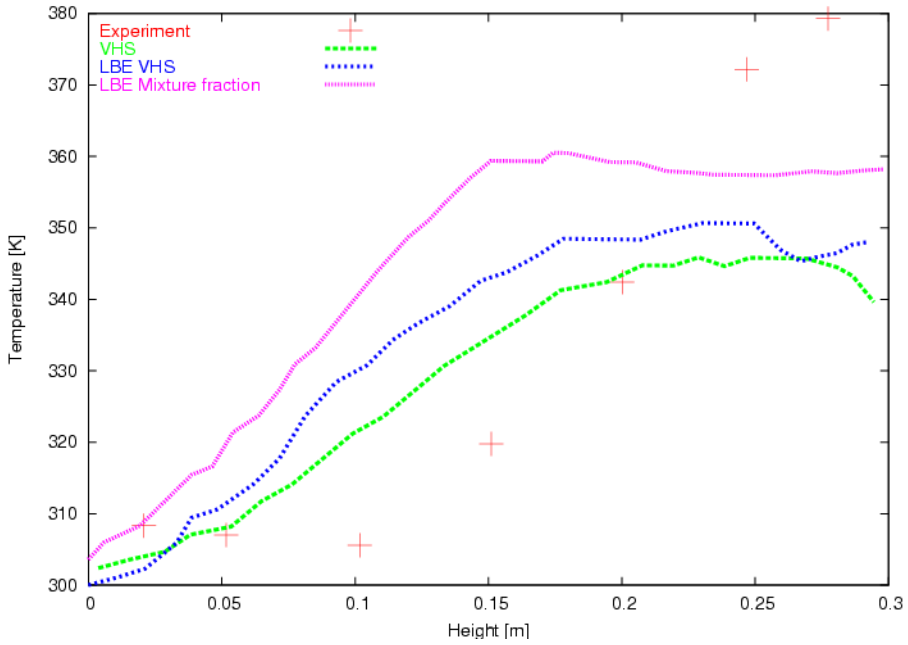


Figure 14: Tunnel fire: Temperature distribution at station C1 (upstream)

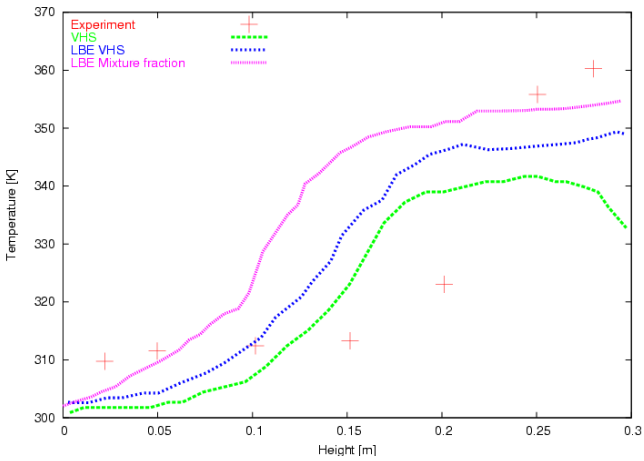


Figure 15: Tunnel fire: Temperature distribution at station C2 (downstream)

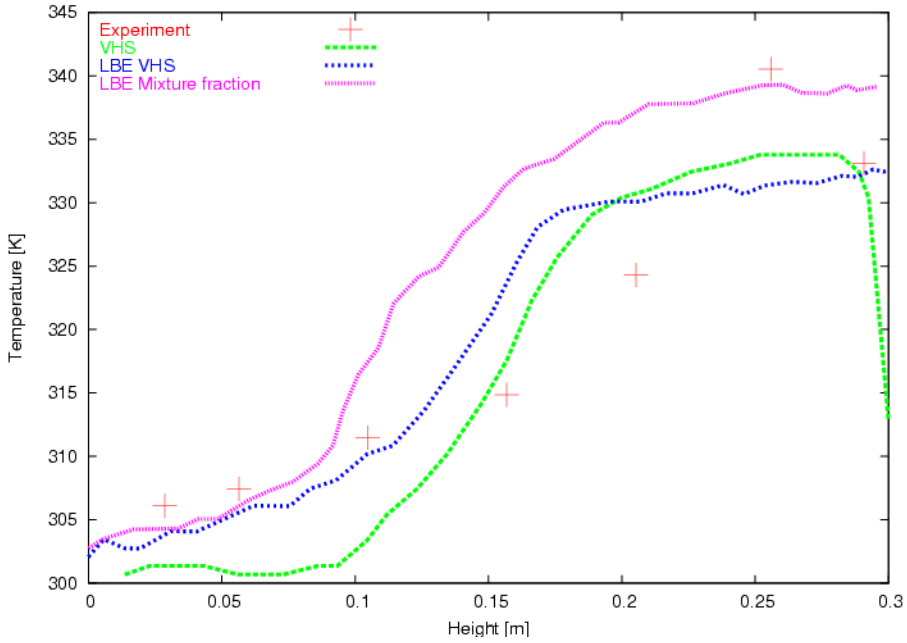


Figure 16: Tunnel fire: Temperature distribution at station C3 (downstream)

4.6.4.4 Discussion

A few concluding comments on the performance of the tested LBE combustion models have to be made:

- The use of the Boussinesq approximation for modelling buoyancy effects is doubtful especally in high temperature regions and for large temperature gradients.

- Outside the near fire region both the VHS as well as the mixture fraction model give resonable results compared to reference simulations and experimental data [22].

- As adiabatic walls are assumed especially the LBE mixture fraction model tends to overpredict temperatures. Nevertheless adiabatic boundaries are used as conducting walls

increase the computational effort and therefore contradict the requirements of real time simulation.

- Differences in the results for both LBE combustion models are a consequence of the different representations of the fire.

- The use of a constant value for the Smagorinsky coefficient may be considered as a weak point. Nevertheless if the Smagorinsky coefficient is in a suitable range (Cs around 0.13) resonable results are obtained.

The test case highlighted some weak points of the implemented models. If one compares the performance of the LBE combustion models to the solutions obtained by a commercial code [22] the results are satisfactory. In view of the many additional unknown factor in real fire hazards the limitations of the LBE combustion models can be accepted. Nevertheless there is much space for future improvements.

4.6.4.5 Real World Applications

In what follows the application of the Virtual Fires simulator is demonstrated by several example calculations of tunnel fires. As this are typical simulations which are set up for real time calculations all of them consists of about only 60,000 computational cells. For all cases the *"fire"* is represented by a 400 [K] isosurface, a temperature which is considered lethal to people.

4.6.4.6 General Remarks

All simulations are carried out using the MRT LBM with Smagorinsky type subgrid scale model. Buoyancy is again considered by means of the Boussinesq approximation. If the mixture fraction model is used to represent the fire the combustion of methan is assumed and the methan inlet mass flow is adjusted to match the required heat release rate. The Smagorinsky constant is $C_s = 0.13$ and the turbulent Prandtl number is equal 0.2. The walls are always assumed to be adiabatic.

4.6.4.7 Gleinalm Tunnel

The Gleinalm tunnel is located in Styria, Austria and is part of the Phyrn motorway connecting Graz with the upper part of Styria. The southern portal is located about 35 [km] in the north of Graz, the northern one in the vicinity of St. Michael in der Obersteiermark. The Gleinalm tunnel is one of the longest tunnels in Austria with a length of 8.320 [km]. The altitude of both tunnel portal is about 800 [m]. The regular cross section of the Gleinalm tunnel can be seen in fig. 17.

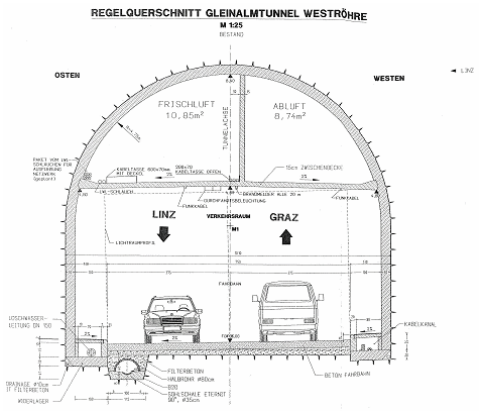


Figure 17: Regular cross section of the Gleinalm tunnel

4.6.4.8 Ventilation System

As in most Austrian single lane tunnels a full cross ventilation system is in operation. The ducts are located above the tunnel ceiling. In the ventilation system of the Gleinalm tunnel the fresh air openings of 0.605 [m] * 0.25 [m] size are located every 6 [m] above the lane directed to St. Michael (north) and are part of the tunnel ceiling. The maximum fresh air rate is about 135 [m³/s/km]. The exhaust air openings with a size of 3.0 [m] * 3.0 [m] are located every 100 [m] above the road track directed to Graz (south).

Although both tunnel portals are located at an altitude of about 800 [m] there can be a substantial bariometric pressure difference. The Gleinalm massif is a strong weather divide between the southern and the northern part of Styria. Long term experience indicates that the typical weather situation is high bariometric pressure at the southern portal and low bariometric pressue in the north.

4.6.4.9 Computational Model

A tunnel section of 75 [m] length is modelled. The computational set up consists of 150 x 22 x 16 cells, so the cells size is about 0.5 [m]. The tunnel walls are assumed to be adiabatic. To model axial flow a velocity inlet conditions is set at one side of the computational domain, whereas a fixed pressure boundary is set on the other side. A small sized HGV fire with a heat release rate of 20 [MW] is assumed. The VHS model is used to represent the fire. The Smagorinsky constant is equal 0.13, wheras the turbulent Prandtl number is 0.2. No radiative heat losses are considered.

4.6.4.10 Simulation Results

Two cases are considered:

Testcase A. The first case assumes a very favourable situation, where a medium sized fire occurs just below an extraction opening. The longitudinal velocity is very small (u = 0.1 [m/s]). The temporal development of the fire can be seen from fig. 18 - 21. After the initial, increasing stage the ventilation system is able to control the fire and to restrict the 400 [K] zone to a narrow area around the buring vehicle.

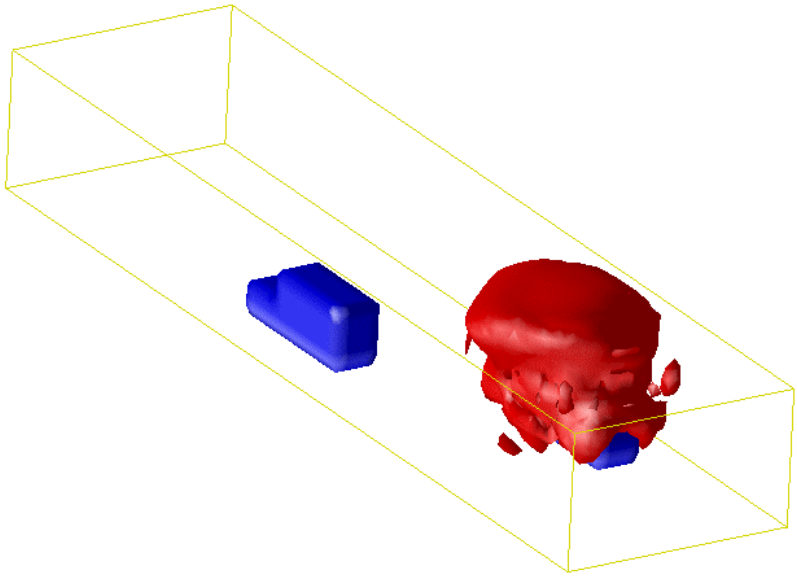


Figure 18: Gleinalm tunnel(Testcase A): Time = 10[s]

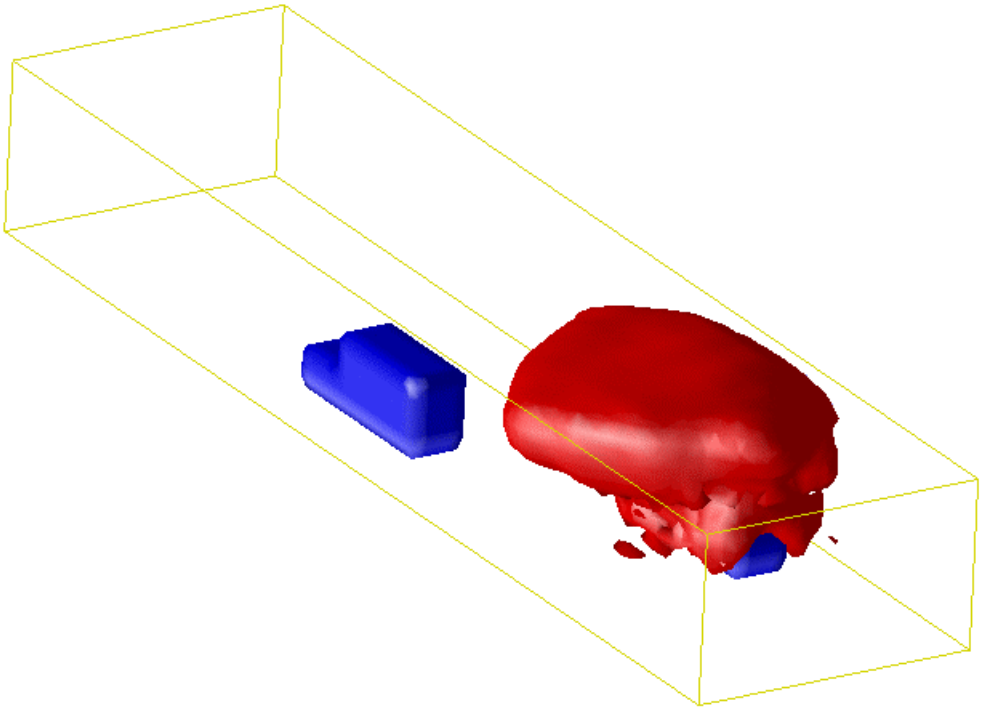


Figure 19: Gleinalm tunnel(Testcase A): Time = 30 [s]

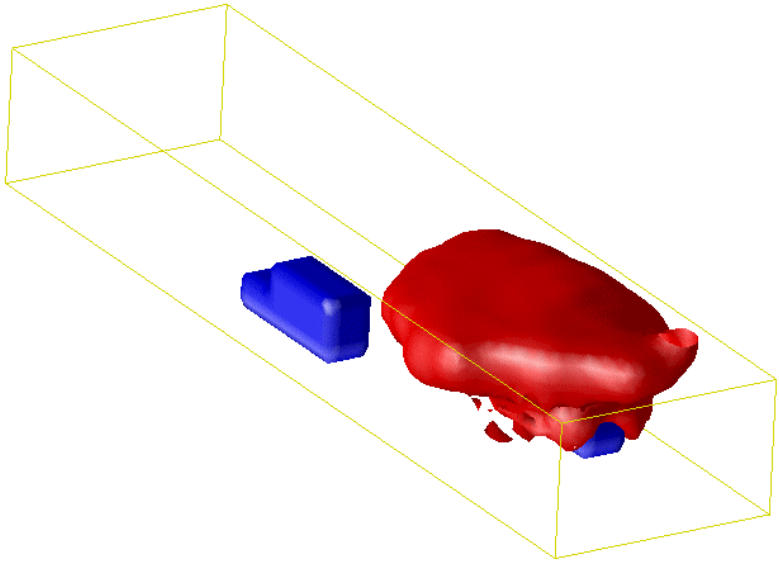


Figure 20: Gleinalm tunnel(Testcase A): Time = 50 [s]

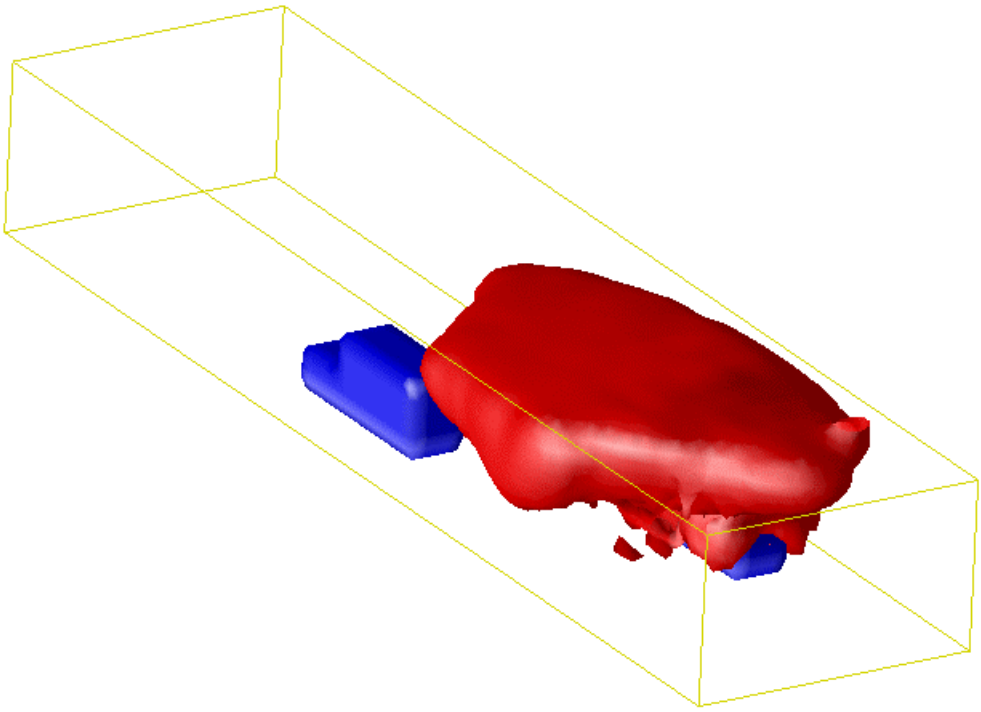


Figure 21: Gleinalm tunnel(Testcase A): Time = 200 [s]

Testcase B. It is known from several studies (see e.g. [14]) that in case of fire the longitudinal velocity must be reduced to a minimum. Therefore the same situation as in Testcase A is considered except that the axial ventilation is now 4.5 [m/s] in the beginning. In emergency the

tunnel operator attempts to act against this longitudinal ventilation by e. g. activating jet fans. Ideally this yields a zero axial velocity near the fire region.

Fig. 22 shows the inital stage of the fire. As it can be seen from fig. 23 the fire reaches the two passenger cars represented by the blue boxes. At this point in time it is assumed that the tunnel operator takes some intervention measures. In this case the longitudinal velocity is reduced (this can be done by operating jet fans or curtains as proposed by Öttl et al. \cite{Oettl_VKM_2002}). Figs. 24 - 27 show how the fire is reduced in size and the exhaust air system is able to extract the hot gases.

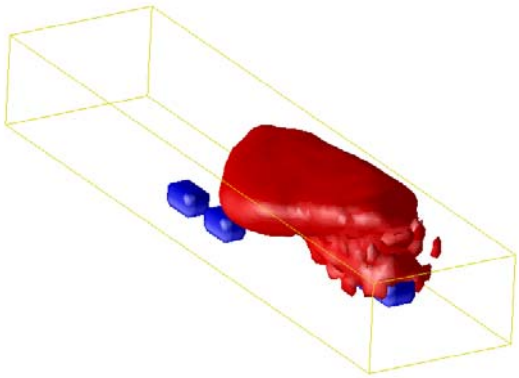


Figure 22: Gleinalm tunnel(Testcase B): Time = 30 [s]

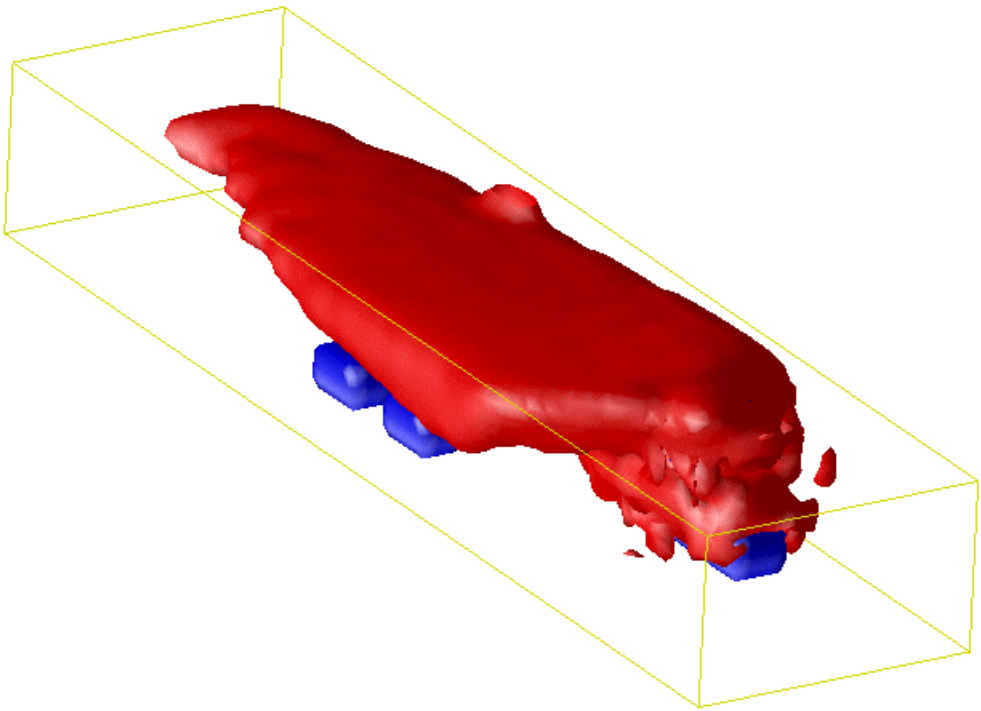


Figure 23: Gleinalm tunnel(Testcase B): Time = 60 [s]

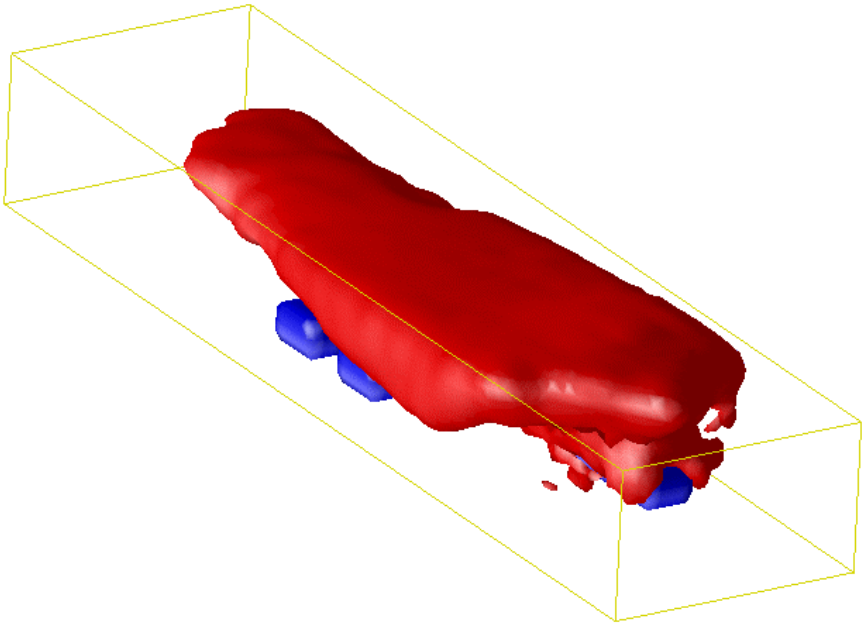


Figure 24: Gleinalm tunnel(Testcase B): Time = 120 [s]

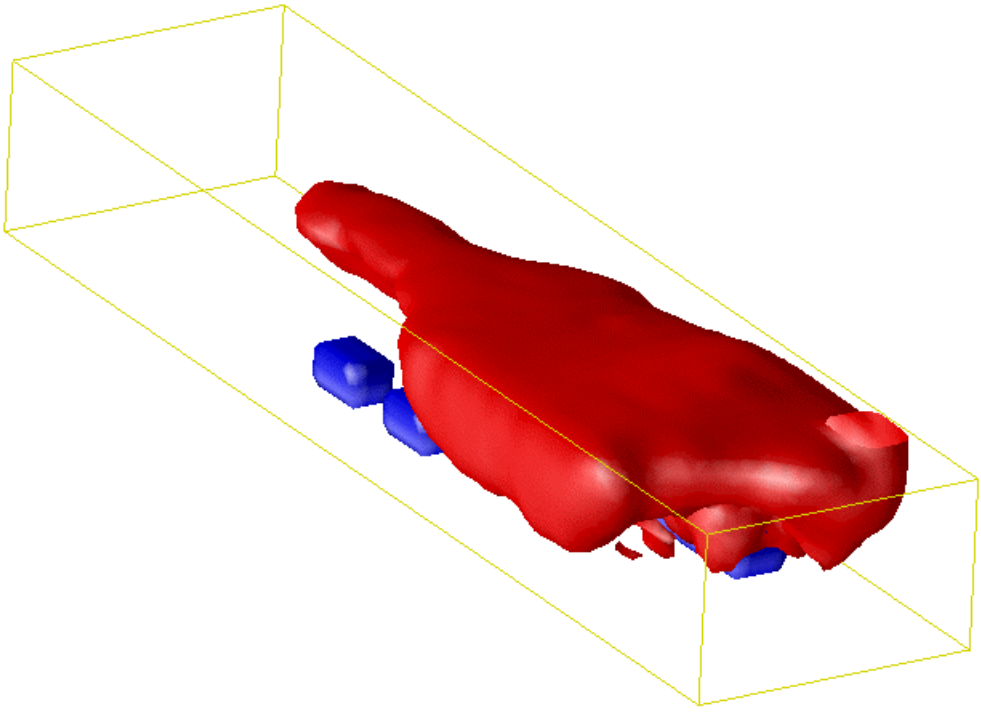


Figure 25: Gleinalm tunnel(Testcase B): Time = 150 [s]

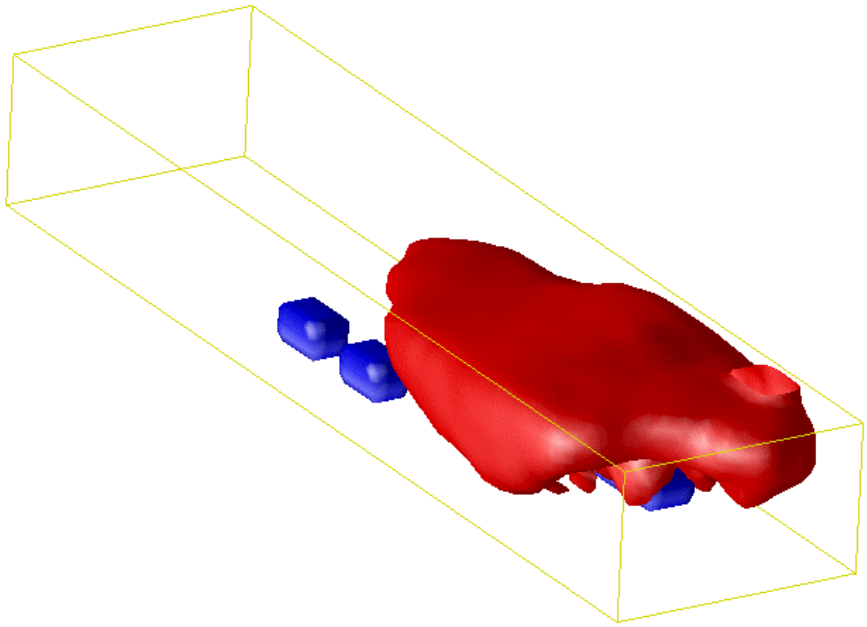


Figure 26: Gleinalm tunnel(Testcase B): Time = 180 [s]

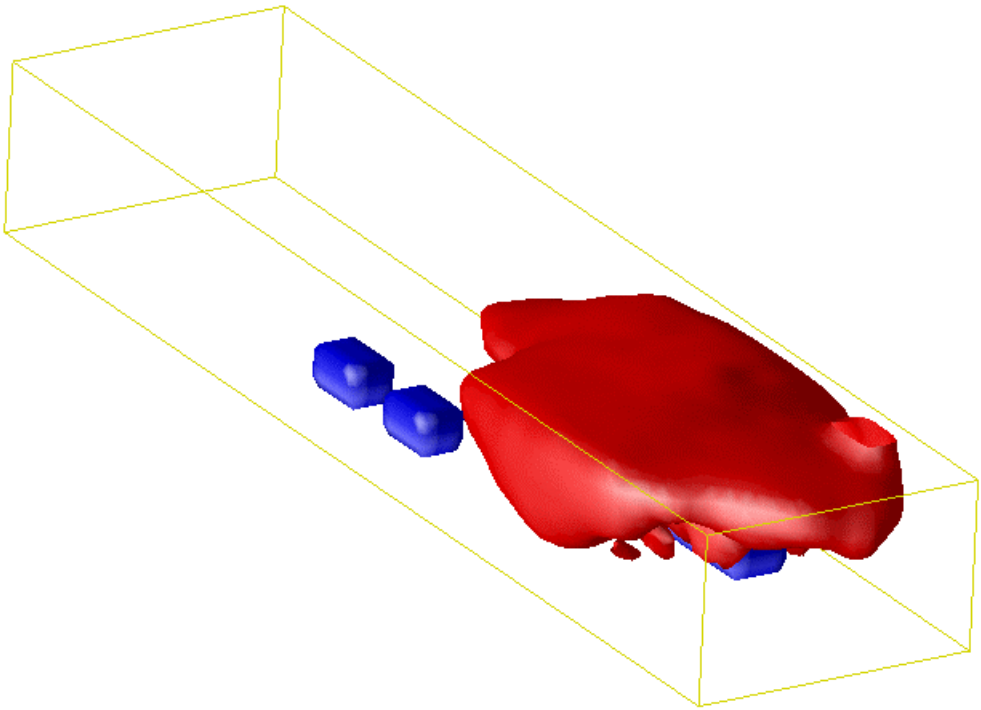


Figure 27: Gleinalm tunnel(Testcase B): Time = 240 [s]

4.6.4.11 Mont Blanc Tunnel

The Mont Blanc tunnel connects France and Italy beneath the Mont Blanc massif. The tunnel is located at an altitude of about 1,300 [m] with the Italian portal lying 107 [m] higher than the French one. The tunnel length is 11,600 [m] with a maximum slope of 2.4 [%] on the French side. The Mont Blanc tunnel is a two directional tube with an open cross section of about 46 [m²]. It was in operation for 34 years before a catastrophic fire accured on March 24th, 1999 on the Italian side of the tunnel. A cross section of the tunnel is shown in fig. 28. Details can be found in e. g. Brousse et al. [1], [13].

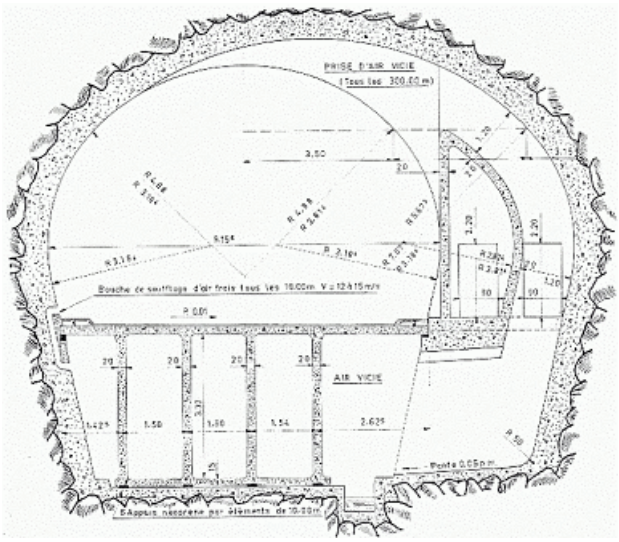


Figure 28: Regular cross section of the Mont Blanc tunnel (Courtesy of CETU)

4.6.4.12 Ventilation system

The ventilation system is of semi-transverse type with ducts lying beneath the road surface. Four fresh air ducts start from each portal and each services a quarter of the tunnel half-length. A fifth duct was originally build for extraction, but can also be used for fresh air supply. The fresh air inlet openings of 0.32 [m] x 0.12 [m] size are located every 10 [m] at the lower part of the lateral wall in the Italy-France direction. An exhaust air extraction opening is located every 300 [m] at the upper part of the opposite site. The extraction openings are of 1.5 [m] x 1.0 [m] size. The control of the naturally induced longitudinal velocity is realised by means of jet fans.

Furthermore according to Brouses et al. [1] strong natural ventilation effects can occur due to a bariometric pressure difference on both sides of the tunnel. This pressure difference lies typically in the rage of 300 [Pa] but can be up to 1,000 [Pa]. As a consequence of this pressure difference the longitudinal velocity can reach values of up to 15 [m/s]. In emergency cases the velocity must be reduced to about 1.5 [m/s] as fast as possible to avoid destratification of the smoke.

4.6.4.13 Computational Model

The computational model of the Mont Blanc tunnel is depicted in fig. 29. Again a 75 [m] long tunnel section is considered. The model consists of 150 x 22 x 16 cells. The walls are assumed to be adiabatic. The mixture fraction model was used to represent the fire. To induce a natural flow

fixed pressure boundary conditions are applied on both side of the tunnel. The exhaust air openings are modelled by fixed pressure outlets. Again the Smagorinsky constant is equal 0.13 and the turbulent Prandtl number is 0.2.

4.6.4.14 Simulation Results

A medium size fire of 30 [MW] is assumed. A pressure difference between the portal induces a strong longitudinal velocity (u = 3.5 [m/s]) from the left to the right. Gravity is acting against the direction of the longitudinal velocity. The exhaust air extraction is active. A zone of backlayering can be seen near the left border of the computational domain resulting from gravity which is acting against the natural ventilation. A large portion of the considered tunnel section is readily filled with hot gases. In the developing stage of the fire the 400 [K] region extends even beyond the location of the extraction opening (fig. 31). Once the flow has been stabilised the ventilation system is able to restrict the fire zone to the area between the initial fire region and the extraction opening (fig. 32).

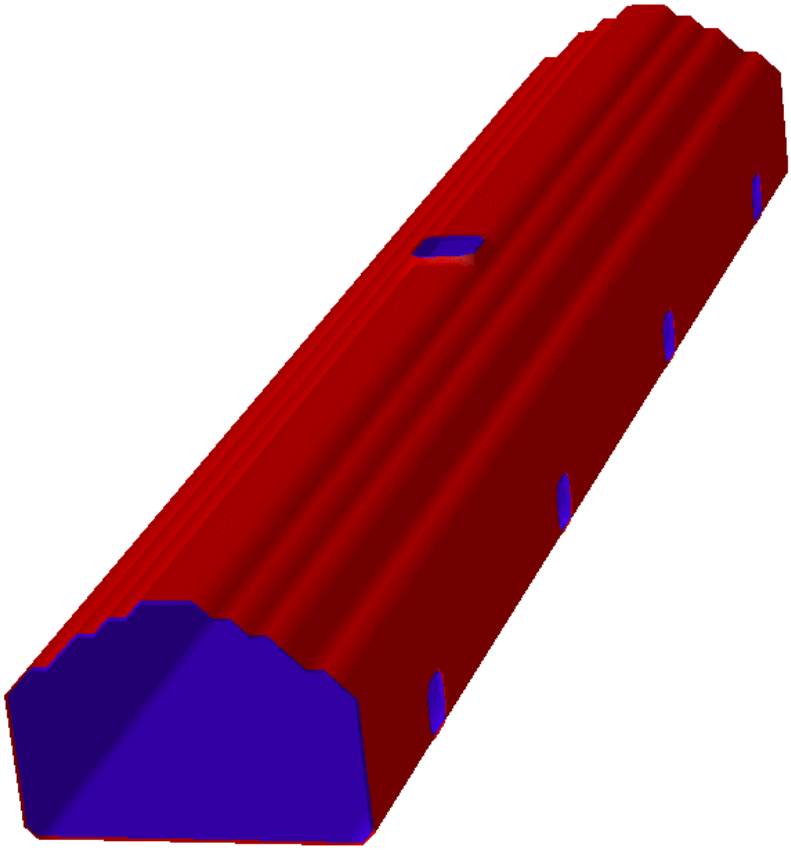


Figure 29: Mont Blanc tunnel: Computational domain

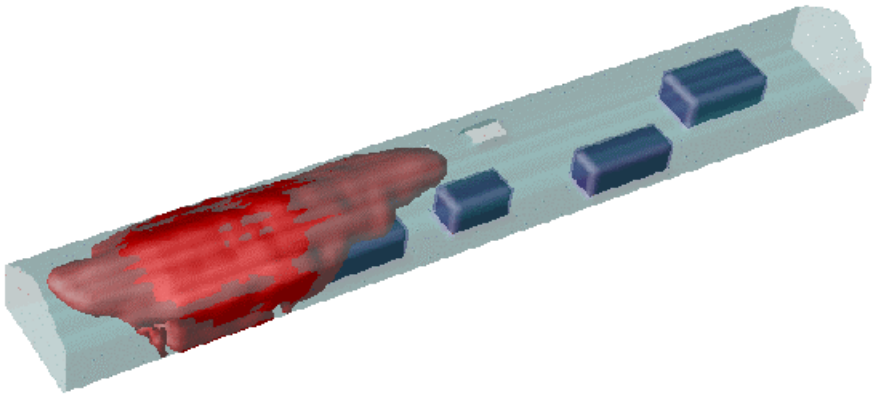


Figure 30: Mont Blanc tunnel: Time = 30 [s]

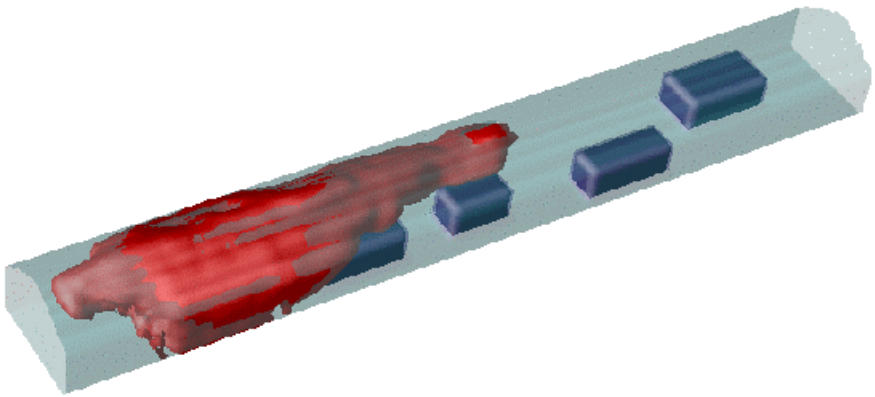


Figure 31: Mont Blanc tunnel: Time = 60 [s]

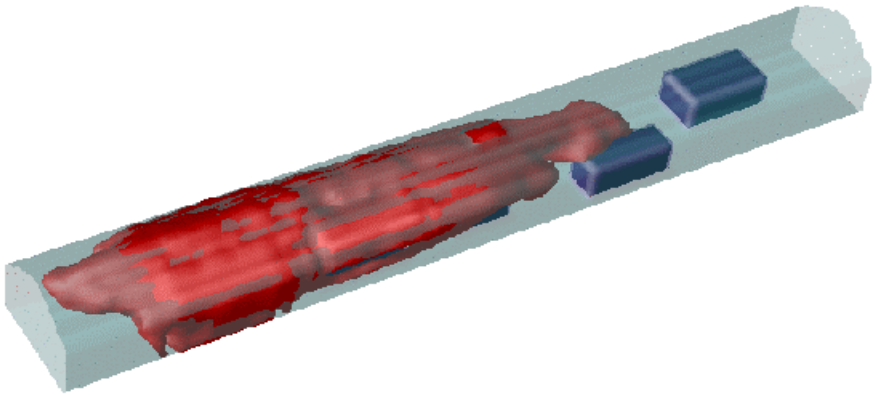


Figure 32: Mont Blanc tunnel: Time = 90 [s]

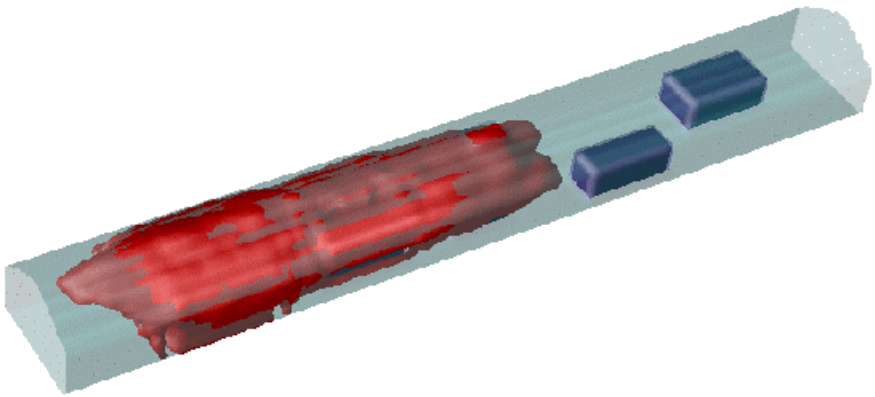


Figure 33: Mont Blanc tunnel: Time = 120 [s]

4.6.4.15 Dortmund Subway Station

As a last test case a simplified model of a subway station provided by the Fire Department Dortmund is presented. The geometry is displayed in fig. 34. According to the Fire Department Dortmund there is no artificial ventilation system in the station. Train movement within the

subway system increases the axial flow velocity within the tubes considerably by a "piston" effect.

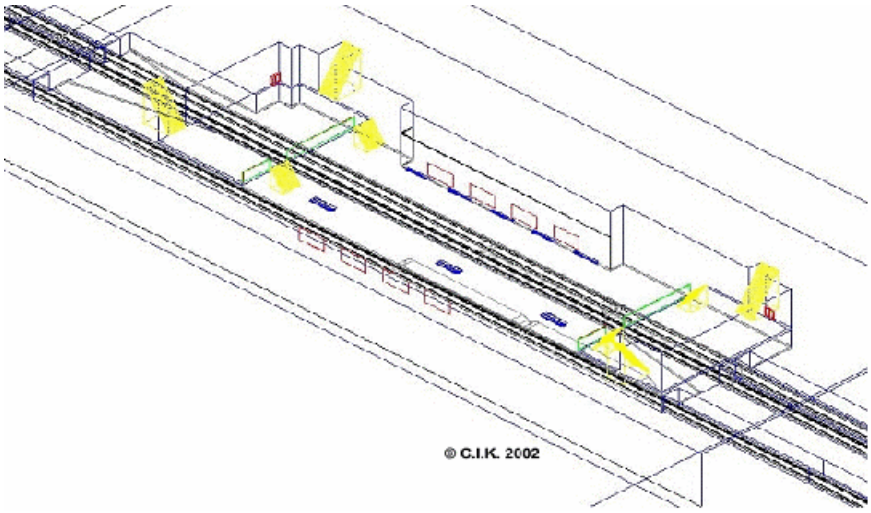


Figure 34: Dortmund Subway Station (provided by Fire Department Dortmund)

4.6.4.16 Computational Model

The computational model consists of 110 x 37 x 22 cells. It is assumed that there is only natural ventilation acting within the station. As a consequence of this a pressure difference is applied between the two sides of the computational domain. An axial flow velocity of 4.5 [m/s] which may result from train movement is considered. For modelling the open ceiling within the computational domain also a constant pressure boundary condition is specified. Radative heat losses are neglected. The default Smagorinsky constant of 0.13 is used. Again the turbulent Prandtl number is 0.2.

Within the simplified geometry a subway train consisting of two wagons is located. This train is subdivided into three "virtual" parts, each of them representing a potential fire region of 2 [MW]. The initial fire region is ignited at the simulation start time. The other fire regions are activated as the solution progresses according to a temperature treshold level. If the computed surface temperature of a train section exceeds an assumed ignition temperature of 380 [K] this condition is met. Within activated fire regions additionally smoke is released. A smoke concentration equal 1.0 is assumed at the fire region. The smoke isosurfaces depicted in fig. 35 to 38 represents smoke concentrations of 0.02, 0.05 and 0.10, respectively.

4.6.4.17 Simulation Results

Figs. 35 to 38 show how fire and smoke are spreading. After some time i. e. when the arbitrarily set temperature treshold of 370 [K] is exceeded the second (fig. 36) and the third (fig. 37) fire region are activated. After 800 [s] (see fig. 38) the whole tunnel downstream the fire is filled with air hotter than 400 [K] and a considerably amount of smoke. Due to the induced axial velocity it is not possible for the hot air and the smoke to escape through the open ceiling. As there is no ventilation system installed there is no possibility to avoid that the tubes are filled with smoke.

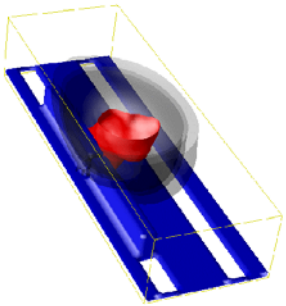


Figure 35: Subway station (perspective side view), t = 100[s]

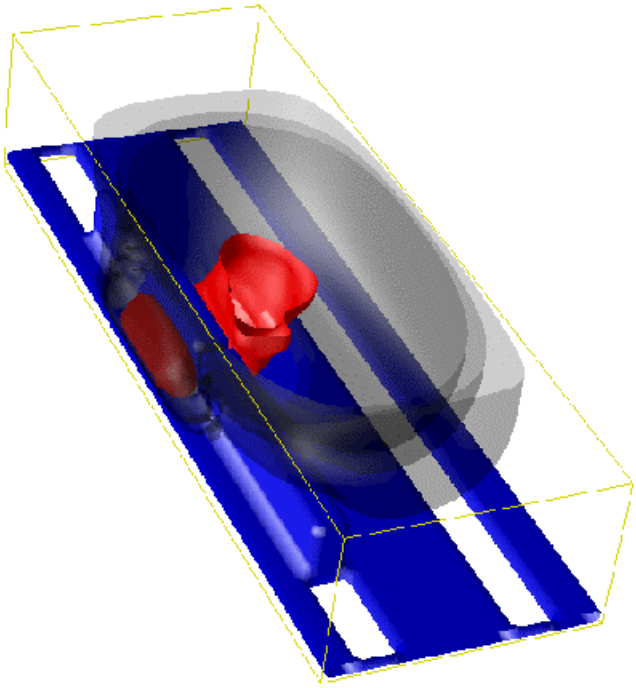


Figure 36: Subway station: Second train section starts burning (perspective side view), t = 300[s]

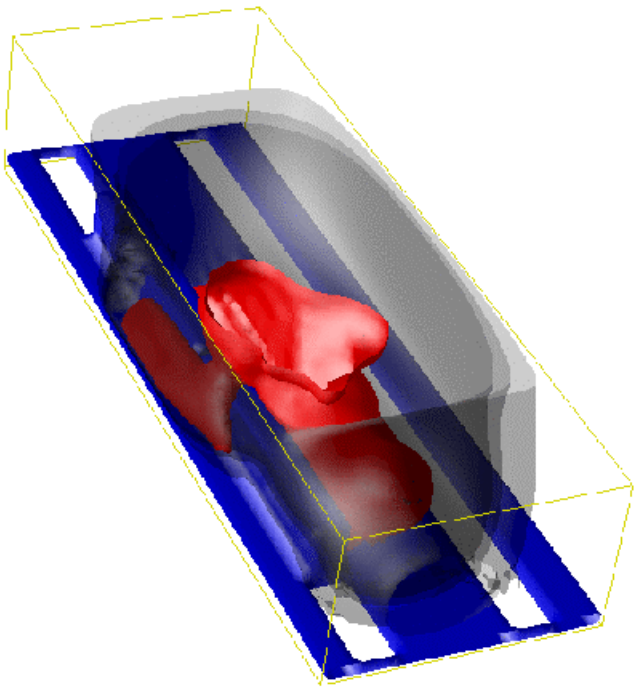


Figure 37: Subway station: Third train section is burning (perspective side view), t = 500[s]

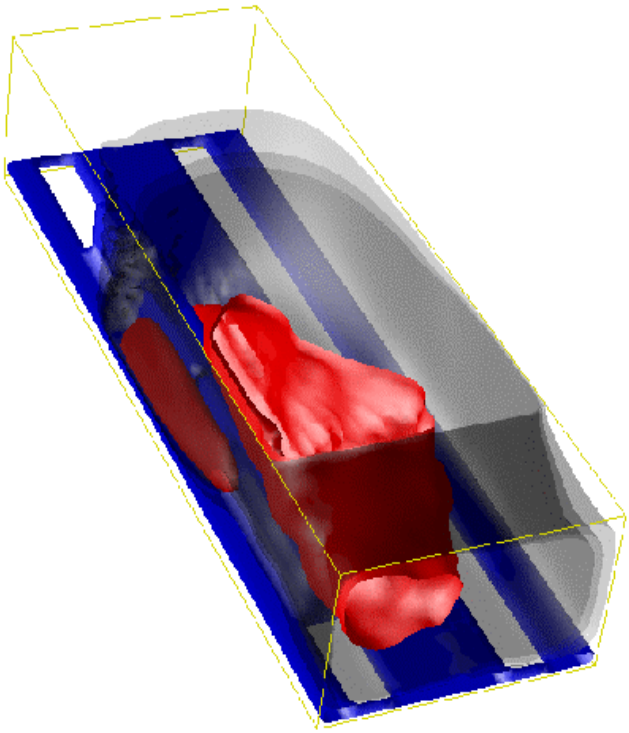


Figure 38: Subway station (perspective side view), t = 800[s]

4.7 Evaluation

A total of 3 evaluation sessions were performed in Lyon, Dortmund and Stockholm. The first two evaluations were made with the PC based version, the last one with the CAVE version of VIRTUALFIRES. At the sessions representatives of the fire departments, tunnel operators and regulatory authorities were present. The results of the end-user evaluation are presented in Deliverable 6.1, while the results of the validation against specifications are presented in D6.2 and can be summarised as follows:

- VIRTUALFIRES was generally well received

- Tunnel operators and regulatory authorities have expressed interest in using the software for the checking of existing tunnels.

- Some criticism and recommendations for improvements were made

The project achieved a highly ambitious goal of having a real time interactive simulation of a fire emergency situation running in a fully immersive environment like the CAVE.

5 List of deliverables

No.	Deliverable	Del. type	Delivered (month)	Delivery (planned)
D2.1	Report on available developer tools	Report	2,5	2
D2.2	Selection of developer tools	Spec.	3,5	2,5
D2.3	Report on available system capabilities (hardware)	Report	2,5	2
D2.4	Specification of planed system capabilities (software)	Spec.	12	2,5
D2.5	Report on existing VR-systems, Adaptability to VIRTUALFIRES	Report	3,5	3,5
D2.6	Specification of selected VR System & required extensions	Spec.	6,5	4
D1	Project presentation	Publicity	1	3
D3.1	Report on geometrical data base	Report	11	5
D3.2	Report on CFD data base	Report	11	5
D2	Dissemination and Use plan	Report	13	6
D3.3	Report on compression/ optimisation	Report	12	6
D3.4	Beta version of data management for first prototype	Prototype	16	15

No.	Deliverable	Del. type	Delivered (month)	Delivery (planned)
D3.5	Beta version of data management for 2nd prototype	Prototype	23	20
D3.6	Release version of data manager	Serial	29	26
D4.1	Specification on methods of displaying CFD data	Report	14	12
D4.2	Specification of user interface in VR-environment	Report	14	12
D4.3	Report on hardware HMD specification	Report	12	13
D4.4	Integration of HMD system completed	Prototype	18	16
D4.5	VR implementation with limited functionality and speed	Prototype	24	18
D4.6	VR implementation with full functionality optimised for speed	Prototype	29	25
D4.7	VR integration with real-time CFD data generation	Prototype	29	25
D4.8	VIRTUALFIRES users manual	Report	30	26
D5.1	CFD database containing results of 6 computational studies	Database	11	11
D5.2	Report: Interactive Field Simulation Techniques; Solver and Data Flow Parallelization	Report	11	11
D5.3	Parallelisation efficiency report	Report	25	24
D5.4	Software User Guide V1.0	User guide	15	14
D5.5	Software User Guide V2.0	User guide	27	26
D6.1	Report on CAVE / HMD installation	Report	35	26
D6.2	VIRTUALFIRES results report	Report	35	28
D7.1	Definition of cases	Report	29	8
D7.2	Journal articles	Report	35	28
D7.3	Conference papers and exhibition at conferences	Report	35	28
D7.4	Webpage	Website	6	6

6 Comparison of initially planned activities and work actually accomplished

It can be stated that the project objectives as described in section 2 of the Annex have been achieved. A prototype of VIRTUALFIRES has been developed. The main innovations originally planned were (section 2 of Annex 1):

- Real time simulation of 3-D transient combustion, concurrent with the visualization.
- Realistic display of fire and smoke
- Efficient handling of large amounts of data

In the proposal it was pointed out that to achieve concurrent fire simulations was a major challenge. Initial estimates of the required run time for 1 minute of real time on a single processor machine for 100 000 grid points was 60 hours. However, a grid of this size would only be able to simulate a small proportion of a tunnel but for realistic modeling the correct boundary condition are important at the edges of the grid to reflect the rest of the tunnel. Suggestions for achieving real time simulation (1 minute real time = 1 minute CPU time) given in the proposal were:

1. Solve only for the fluid mechanics in those regions where user interaction takes place.
2. Consider the flow regions in front and behind the active region in a one-dimensional fashion as mentioned above and with proper coupling to an already existing Lattice-Boltzmann Code (LB-Code) which will be used for the flow/combustion simulation in the region of user interaction.
3. Parallelise the LB-Code on an equidistant rectilinear grid for optimum performance.
4. Integrate the complete flow solver package (1D and 3D solver) into the VR environments for real time simulation and visualisation of tunnel fires in a concurrent fashion.

The real time simulation was achieved by successfully implementing points 2) to 4). We claim that this is the first time that a concurrent real time fire simulation has been achieved. A recent INTERNET search revealed no other system in the state of development of VIRTUALFIRES capable of achieving this. Another innovation claimed is the coupling of one- and three-dimensional grids in order to efficiently simulate long tunnels.

With respect to the realistic visualisation of fire and smoke some innovative display methods were developed (as have been shown in 4.5). Unfortunately due to the architecture of the rendering program COVISE it was not possible to integrate these methods into VIRTUALFIRES. However, a workaround was devised which gives an acceptable realism.

The handling of large amount of data that is produced by the CFD solver has also been tackled by an efficient data management system.

Another innovation claimed is the use of a PDA in a CAVE environment for the GUI, which allows to select existing missions, visualise data stored in a data base, adjust the speed of the visualisation. Start/restart/stop the computation and change some properties.

7 Management and coordination aspects

All partners were dedicated to the project and motivated. In some cases deliverables were submitted late. In many cases there were compelling reasons for this which included unexpected problems with third party software. Some problems were experienced with the rendering software COVISE which was selected as a basis for VIRTUALFIRES. However, there was good cooperation with the company which markets COVISE (Vircinity) and the director of the company Dr. Wierse attended the consortium meeting in Dortmund so that problems could be discussed and resolved as quickly as possible. One of the problems the project had to deal with was the continuous rapid development of hardware since the conception of the project more than 4 years ago. For example at one stage KTH decided to discontinue the maintenance agreement with Silicon Graphics for the CAVE which was potentially disastrous for the project. Fortunately this could be resolved. Another aspect which resulted in delays was the fact that it became obvious fairly late in the project that the innovative visualisation methods devised by FIGD could not be implemented into VIRTUALFIRES. However, a concentrated effort by the consortium helped to overcome the problems albeit with some delay.

The person which may be contacted concerning the follow-up of the project are:

Prof. Gernot Beer
Institut fuer Baustatik
Graz University of Technology
Lessingstasse 25
A - 8010 GRAZ
Tel: +43 316 873 6180
e-mail: beer@ifb.tu-graz.ac.at

8 Results and conclusions

The main result of the project is a tested and evaluated prototype of VIRTUALFIRES which will be commercially exploited. During the development a number of innovations were made, which advance the state of art in virtual reality, computational fluid dynamics, the handling of large amounts of data and the use of GUIs in CAVE environments.

The conclusion is that despite setbacks, which are not uncommon in research projects, the objectives have been achieved and the project has been successful in advancing European science and technology.

9 Acknowledgements

We grate fully acknowledge the guidance and critical review of this project supplied by the project officer and the reviewers, which helped us improve our focus and to achieve the result reported here.

10 References

[1] Brousse, B., Voeltzel, A., Le Botlan, Y., Ruffin, E., Ventilation and Fire Tests in the Mont Blanc Tunnel to better understand the Catastrophic Fire of 24 March 1999, as provided by Centre d'Etudes des Tunnels (CETU), Bron, France (2001).

[2] Centre d'Etudes des Tunnels, Les Etudes Specifiques des Danger (ESD) pour les Tunnels de Reseau Routier, Ministére de l'Equipment, des Transports et du Logement, Direction des Routes, 2001.

[3] Chen, S., Doolen, G. D.Lattice Boltzmann Method for Fluid Flows,Annu. Rev. Fluid Mech. 30, 329 – 364 (1998).

[4] D'Humieres, D., Ginzburg, I., Krafczyk, M., Lallemand, P., Luo, L.-S., Multiple-Relaxation-Time Lattice Boltzmann Models in Three Dimensions, Phil. Trans. R. Soc. Lond. A 360 (2002), 437 – 451.

[5] Foster, I., Designing and Building Parallel Programs, Addison-Wesley, 379 pp, 1995.

[6] http://www.mpi-forum.org

[7] http://www.myri.com

[8] http://www.pdc.kth.se

[9] http://www-unix.mcs.anl.gov/mpi/mpich/

[10] Lallemand, P., Luo, L.-S., Theory of the Lattice Boltzmann Method: Acoustic and Thermal Properties in Two and Three Dimensions, Phys. Rev. E 68, 2003.

[11] Luo, L.-S., The Lattice Gas and Lattice Boltzmann Methods: Past, Present, and Future, Proceedings Int. Conf. Appl. Comp. Fluid Dyn., Beijing, China, October 17 – 20, 2000.

[12] Mei, R., Shyy, W., Yu, D., Luo, L.-S., Lattice Boltzmann Method for 3-D Flows with Curved Boundaries, J. Comp. Phys. 161 (2000), 680 – 699.

[13] Ministere de l'Equipement, des Transports et du Logement (France), Task Force for Technical Investigation of the 24 March 1999 Fire in the Mont Blanc Vehicular Tunnel, English translation of the original French report avialable at http://www.equipement.gov.fr.

[14] Öttl, D., Sturm, P., Almbauer, R., Öttl, W., Thurner, A., Seitlinger, G., A New System to Reduce the Velocity of the Air Flow in the Case of Fire, International Conference on Tunnel Safety and Ventilation, April 2002, Graz (Austria), VKM-THD 80, pp 279 - 286 (2002).

[15] Pacheco, P. S., Parallel Programming with MPI, Morgan-Kaufmann, 418 pp, 1997.

[16] Qian, Y. H., D'Humieres, D., Lallemand, P., Lattice BGK models for the Navier-Stokes equations, Europhy. Lett, 17 (1992), 479 – 484.

[17] Redl, C., Brandstätter, W., Project Deliverable 5.5 – Software User Guide V2.0, 2003.

[18] Snir, M., Otto, S., Huss-Ledermann, S., Walker, D., Dongarra, J., MPI: The complete Reference, MIT Press, 336 pp, 1996.

[19] Spalding, B. D., Combustion and Mass Transfer, Pergamon Press, (1979).

[20] Succi, S.,The Lattice Boltzmann Equation for Fluid Dynamics and Beyond, Numerical Mathematics and Scientific Computation, Oxford University Press, 2001.

[21] The MPI Forum, MPI-2: Extensions to the Message Passing Interface, University of Tennessee, 362 pp, 1997.

[22] Xue, H., Ho, J. C., Cheng, Y. M., Comparison of Different Combustion Models in Enclosure Fire Simulation, Fire Safety Journal 36, p 37 - 54 (2001).

[23] Yu, D., Mei, R., Luo, L.-S., Shyy, W., Viscous Flow Computations with the Method of Lattice Boltzmann Equation, Prog. Aero. Sc. 39, pp 329 - 367 (2003)

11 Appendices

Appendix 1 – One dimensional Navier-Stokes equations

The one dimensional Navier-Stokes equations together with the species transport equation are given by:

$$\begin{aligned} &\frac{\partial \rho}{\partial t}+\frac{\partial(\rho u)}{\partial x}=0 && (\textit{mass equation}) \\ &\frac{\partial(\rho u)}{\partial t}+\frac{\partial(\rho u^2)}{\partial x}=-\frac{\partial p}{\partial x}-\frac{P}{A}\tau_w+\mu\frac{\partial^2 u}{\partial x^2}+g\omega && (\textit{momentum equation}) \\ &\frac{\partial c}{\partial t}+\frac{\partial(cu)}{\partial x}=\frac{\partial^2 c}{\partial x^2} && (\textit{transport equation}) \end{aligned} \qquad (1)$$

where p is the pressure, A is the cross-section area of the tunnel, P is the perimeter of the cross-section, τ_w is the friction term, μ is the artificial viscosity coefficient and $g\omega$ is the gravity term.

The pressure is correlated to the density by

$$p = c_s^2 \rho$$

where c_s is the sound speed given by

$$c_s = \frac{c}{\sqrt{3}}$$

By adopting the lattice Boltzmann approach where

$$c = \frac{\Delta x}{\Delta t}$$

and setting $\Delta x = \Delta t$ we obtain the pressure p i.e.

$$p = \frac{\rho}{3}$$

Due to the friction from the walls, the friction term τ_w is included in the momentum equation i.e.

$$\tau_w = \frac{C_F}{2}\rho u|u| + K\rho\frac{\partial u}{\partial t}$$

where C_F is the quasi-steady friction constant and K is the unsteady friction constant given by

$$K = 1.555 - 0.8522(\mathrm{Re})^{0.079} + 0.2043(\mathrm{Re})^{0.158} - 0.0183(\mathrm{Re})^{0.2367}$$

This friction model is developed for unsteady flows by Carstens and Roller. More details about this model as well as other friction models ca be found in [1].

The gravity term consists of the gravity g and

$$\omega = \beta(c - c_0)$$

where β is the coefficient of smoke expansion and c_0 is the average concentration of smoke. This term is included in the momentum equation by following the well-known Boussinesq approximation. More information about the gravity term can be found in [2].

The numerical method used to solve this system of equations is the Runge-Kutta method of order 4 while the central differences are employed for space discretization.

Appendix 2 - 3D/1D coupling

Consider the mass and momentum equations in (1). These equations can be written as

$$\frac{\partial Q}{\partial t}+\frac{\partial E}{\partial x}=F \qquad (2)$$

where

$$Q=\begin{bmatrix}\rho\\ \rho u\end{bmatrix},\quad E=\begin{bmatrix}\rho u\\ \frac{1}{M}(\rho u^2+p)\end{bmatrix},\quad F=\begin{bmatrix}0\\ -\frac{PC_F}{2AM}\rho u|u|+\frac{\mu}{M}\frac{\partial^2 u}{\partial x^2}\end{bmatrix}$$

and

$$M=1+\frac{PB}{A}$$

By using

$$\frac{\partial E}{\partial x}=\frac{\partial E}{\partial Q}\frac{\partial Q}{\partial x}=A\frac{\partial Q}{\partial x}$$

we obtain (2) in quasi-linear form

$$\frac{\partial Q}{\partial t}+A\frac{\partial Q}{\partial x}=F$$

where

$$A=\begin{bmatrix}0 & 1\\ \frac{1}{M}(-u^2+\frac{1}{3}) & \frac{2u}{M}\end{bmatrix}$$

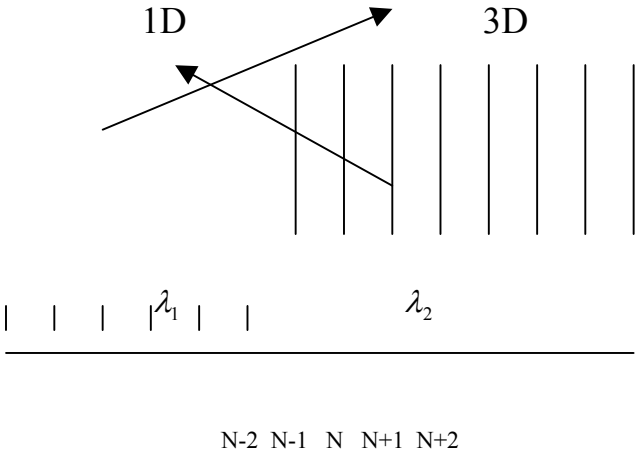
The eigenvalues of the matrix A are

$$\lambda_{1,2}=\frac{u}{M}\pm\sqrt{\frac{1}{M}(\frac{u^2}{M}-u^2+\frac{1}{3})}$$

and the corresponding eigenvectors are

$$r_1 = \begin{bmatrix} 1 \\ \lambda_1 \end{bmatrix}, \quad r_2 = \begin{bmatrix} 1 \\ \lambda_2 \end{bmatrix}$$

Assume now that we have the following situation: the last point in the 1D area is denoted by



N while this point is the first x-point in the 3D area. At each time step the field values at this point are updated by

$$R_{N-1}^{-1} Q_N = R_{N-1}^{-1} Q_L$$

where

$$R = \begin{bmatrix} r_1 & r_2 \end{bmatrix}$$

and L is determined by the characteristics. In this particular case (see the figure above) the first characteristic points into the 3D area i.e. $\lambda > 0$ thus $L = N - 1$ for $Q(1)$. The other characteristic points into the 1D area i.e. $\lambda < 0$ thus $L = N + 1$ for $Q(2)$.

The concentration of smoke at the inner boundaries is updated by following the similar procedure as for the mass and momentum. It is worth noticing that $\lambda = u$ in the transport equation which makes this procedure trivial.

Appendix 3 – 3D grid reduction

When the 3D grid is expanded, new 3D grid extension is generated. The width and the height of the 3D grid extension remain unchanged while the length is pre-computed in the initialization procedure. The grid is initialized with the 1D field values scaled by the 3D boundary values i.e.[3]

$$c_{i,j,k} = c_i^{1D} \frac{c_{j,k}^{3D}}{\frac{\sum_{j}^{NJ}\sum_{k}^{NK} c_{j,k}^{3D}}{NJ \cdot NK}}$$

where i, j, k are indices in x-, y- and z-directions, NJ and NK are number of cells in y- and z-directions and $3D$ denotes the values on the 3D boundary grid.

The 3D grid can be expanded to the left and/or to the right except in two cases:

- if the total number of 3D cells has reached the upper limit the 3D grid cannot be expanded at all and
- if the 3D grid has reached the end of the tunnel than it cannot be expanded in that direction, of course.

References:

[1] Shuy E. B., Apelt C. J., *Friction Effects in Unsteady Pipe Flows*, 4th International Conference on Pressure Surges, September 21-23 1983

[2] Guo Z., Shi B., Zheng C., *A coupled lattice BGK model for the Boussinesq equations*, Int. J. Numer. Meth. Fluids 2002, 39:325-342

[3] ρ and ρu are updated in the similar way.