An interactive thermal fluid simulator for the design of HVAC systems

Michael Pfaffinger,
Christoph van Treeck, André Borrmann, Petra Wenisch, Ernst Rank

Computational Civil and Environmental Engineering, Technische Universität München, Munich, 80290, Germany; PH +49 89 289-23045; FAX +49 89 289-25051; email: pfaffinger@bv.tum.de

Abstract

This paper describes the current state in the development of a computational steering environment (CSE) for indoor air-flow simulation. The system is especially desiged to provide an interactive evaluation of thermal comfort. It consists of an integrated, VR-based visualization client, a parallel computational fluid dynamics (CFD) kernel and a 3D grid generator component. The CSE is intended to support engineers with an easy-to-use tool for designing HVAC systems (heating, ventilating and air conditioning), for example, in the early stages of a project. By utilizing high-performance computing, the tool enables the investigation of indoor air-flow scenarios approaching closely to "real-time".

The user interface for visualizing simulation results and the "steering" of the simulator have been combined to form a single application front-end. This allows an intuitive access for steering the simulation, changing the set-up of geometry and boundary conditions and subsequently exploring the results. The simulation kernel is based on a hybrid lattice Boltzmann method with extensions for large-eddy simulations of turbulent convective flows.

The next steps in the project are to improve the VR client, enhance the treatment of boundary conditions in the computational kernel and extend the simulator in conjunction with partners in order to integrate a local thermal comfort model.

Introduction

Computational fluid dynamics (CFD) tools are widely available nowadays, but the evaluation of indoor thermal comfort still remains a complex task. Limited by the complexity of mesh generation and set-up of a numerical model, the application of these tools is restricted to a few parameter studies in practice. Furthermore, results have to be interpreted in order to predict the climatic impact on the occupants of a room.

Simulations of this kind are of growing importance in many areas of application, such as the design of HVAC systems in offices, of train carriages, aircraft cabins or machine rooms. For example, they make it possible to predict if and where there may be a risk of draughts due to badly positioned inlets or outlets, or whether the total capacity of a system is sufficient to maintain operative room temperatures on a local level at an early design stage.

To improve this situation we combined the different applications into a single computational steering system (CSE). This environment consists of an integrated, VR-based visualization client, a fast space-tree based grid generator and a simulation kernel. By using the grid generator software even complex geometries are processed on-the-fly so modifications to the geometry, such as adding and removing objects, can be realized during the computational run-time, i.e. without stopping and restarting. The numerical model is based on a hybrid thermal Lattice Boltzmann method (hTLBE) [4].

Numerical method

The numerical process is based on the Lattice Boltzmann (LB) method, which is an efficient means of solving a range of fluid flow and transport problems, see for example [2], [3], [4], [5], [9]. Unlike classical methods, which are based on the discretized Navier-Stokes equations, LB methods have their origin in statistical physics. Starting with the Boltzmann equation, LB methods compute the advection and local collision of an ensemble of particle distribution functions with the linearized collision operator [4]. The Boltzmann equation is discretized with respect to its microscopic velocity space using an explicit finite difference (FD) scheme. The resulting evolution scheme offers certain advantages in terms of generating the numerical grid and an efficient implementation on high-performance computers.

The physical model of our code is based on the hybrid thermal Lattice Boltzmann (hTLBE) model as proposed by Lallemand and Luo [4], [5]. In this model, the LB approach for solving the momentum equations is explicitly coupled with a finite difference scheme for solving the diffusion-convection equation, i.e. the solution is used to compute the buoyant force in the sense of a Boussinesq approximation [8]. In our case, both numerical methods are computed using the same underlying Cartesian grid. The hTLBE model has been further extended by a sub-grid scale turbulence model [3], [8]. For free convective flow problems the results have been validated in both 2D and 3D; details can be found in [9].

In order to improve the numeric stability of the method, a so-called multiple relaxation time (MRT) approach, as proposed by d’Humières [2], is used. It is based on a transformation of the discretized particle distribution functions from the phase space into a physically equivalent moment space. This makes it possible to individually choose relaxation parameters relating to the kinetic modes.

Space-tree based discretization

A fast 3D grid generator, as implemented in our code [12], is essential for the interactive concept in which the geometry can be modified during run-time. The LB method is typically implemented on a Cartesian grid and allows for a fast and fully automatic grid generation. In terms of computing time it is superior to conventional mesh generation approaches, as used in finite volume based codes, for example.

The grid generator transforms a facetted geometry model of an object incorporated in the application into a uniform Cartesian grid representation. For this reason, a space-tree based partitioning algorithm is used. When the grid generation process is initiated, the algorithm starts with one or more root octants that enclose the geometric objects. If an object intersects an octant, this octant is further subdivided into new sub-octants in a recursive process. Grid points belonging to non-fluid objects are accordingly identified and marked with the corresponding boundary condition, which is specified in a geometry file or may be added/modified interactively during run-time. For instance, an object with approx. 30,000 facets is processed in less then one second. An example is given in Figure 1.

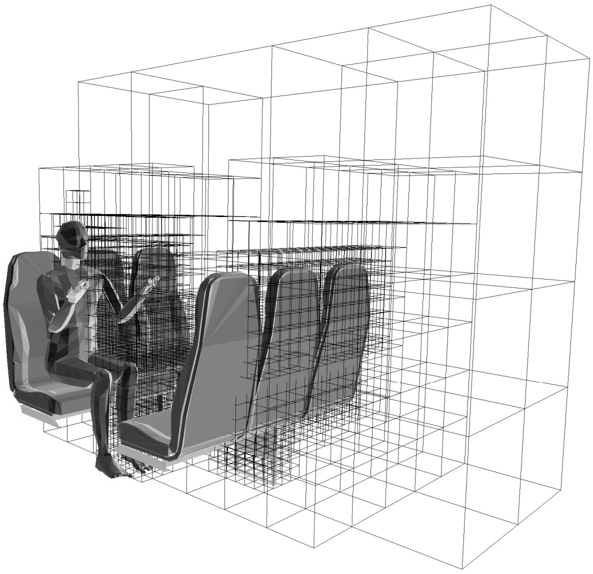

Figure 1: Octree of a facetted geometry model automatically generated during the grid generation process

Computational steering environment

As discussed above, the different components of the CFD kernel with the grid generator and the VR-based visualization client have been integrated into a computational steering environment (see [11]). The user interface for visualizing the geometry and the results, and for steering the simulation have been combined into a single front-end application. The application makes use of the OpenInventor library from Mercury Computer Systems (see [6]) and can be used on desktop workstations, laptop computers or even virtual reality environments running either Windows XP or Linux-based systems.

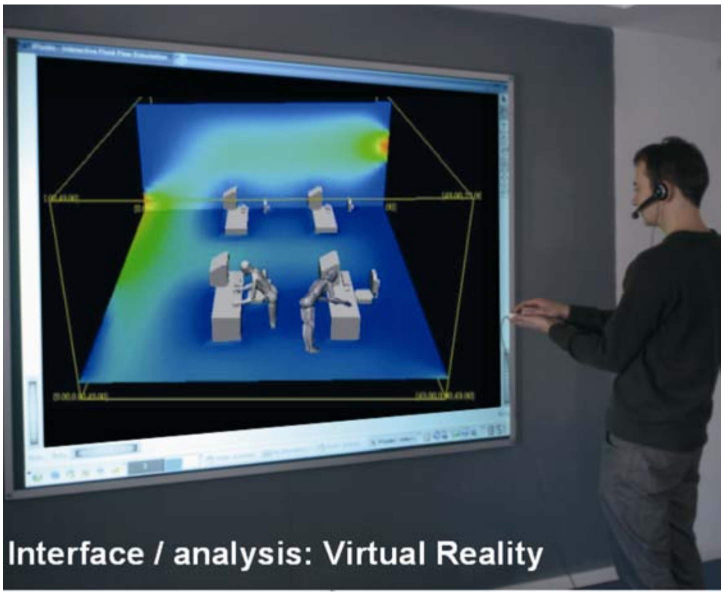

Figure 2: Visualization and steering front-end run on a Linux workstation

By displaying the results together with the original geometry, the application offers an intuitive way of exploring the results (see Figure 2). The interaction with the objects is implemented using draggers for both the visualization of the results and representation of the geometry. These draggers allow the objects to be moved, scaled and rotated. It is possible to load new objects or delete objects during run-time (see [1] and [10]). Boundary conditions, such as the temperature or the inflow velocity, can either be specified in the geometry file or may be edited interactively using a dialog window (see Figure 3). The boundary condition information is stored together with the original (facetted) geometry. Changes have impact on the original geometry data, as loaded into the CSE. Information on boundary conditions, as defined for the geometry, is thereby independent from the applied simulation kernel. Each time changes are made to an object, data is sent to the simulation kernel and the grid is regenerated.

Besides the mentioned stand-alone application of the CSE, a cooperative version (see [1]) is available, in which visualization clients share the same geometry database and work together on a project. Clients may be attached and detached during simulation. It is even possible to use different simulation cores or to visualize data from the same computation on each client application in different ways.

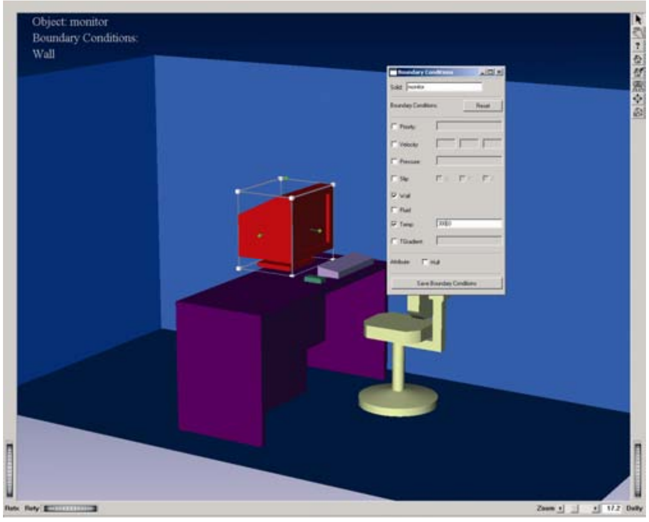

Figure 3: Modification of boundary conditions during run-time using a dialog box

The visualization and steering clients connect to a common, so-called "GeomServer" which holds the geometry information, coordinates the changes to the model and locks an object for other users while a designated user makes changes. While the communication in the stand-alone application is based on MPI for both the parallelized CFD core and between computation and visualization, the communication in the cooperative version between the client, the "GeomServer" and the simulation servers is based on the Orbacus CORBA library (see [7]). Another advantage of using CORBA-based communication is that the GeomServer, the visualization and steering clients as well as the simulation servers may be executed on different hardware platforms without the need for special MPI implementations such as PACX-MPI.

Implementation in a sample VR environment

As part of a research project, we implemented a special version of the CSE to meet the needs of our industrial partner Siemens AG (Corporate Technology) when it comes to presenting results to customers.

The VR room of the participating department of Siemens AG is equipped with a passive stereo setup complete with standard hardware. The whole screen is subdivided into three smaller sub-screens (see Figure 4). Each of these sub-screens has two projectors with polarization filters for the right and the left channel, respectively. The system is built as a visualization cluster, i.e. each projector is directly connected with a visualization computer. Therefore, not only the different views but also the right and left-hand channel of a scene visualized in stereo mode have to be rendered on different computers that need to be synchronized accordingly. In addition, a separate computer for handling the presentation and controlling the individual clients of the visualization cluster is also installed.

The visualization and steering application is started on the master machine in non-stereo mode and all of the menus and messages of the CSE appear on this node in an exclusive manner. This means that these menus do not interfere with the stereo impression, as seen by the audience. The camera positions for the stereo projection on the VR screen are controlled by the master application, i.e. if the person giving the presentation rotates the scene, or if it is zooming in or out, the view as produced by the visualization cluster for the 3D screen follows instantaneously.

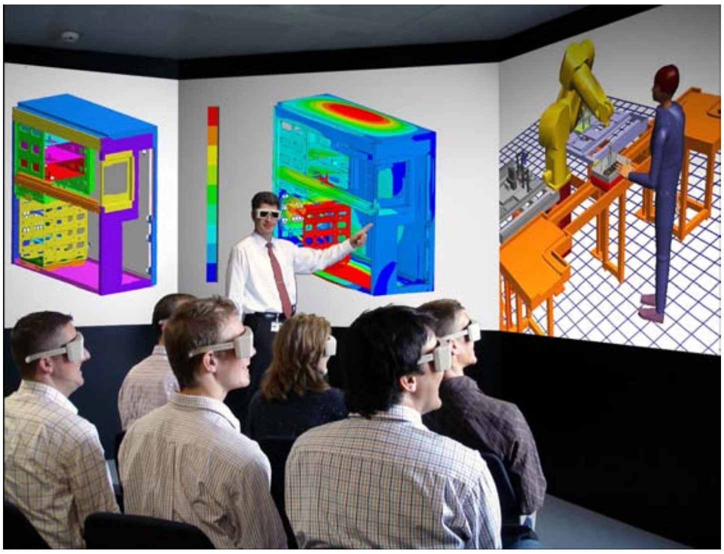

Figure 4: Screen arrangement in a VR room (courtesy of Siemens AG, Corporate Technology)

Unfortunately, the set-up described above is not directly supported by the visualization library used in the CSE. Based on the cooperative version of the CSE, we implemented our own stereo viewing mode, where each machine runs the same visualization and steering client program. Certain flags in a configuration file determine the mode in which the client starts and which configuration values are to be used for the stereo viewing. The values passed to the visualization determine the camera offset, among several other parameters, between the right and left-hand channel and accordingly determine the stereo impression.

The server that handles all the geometry and collaborative information, the “GeomServer”, has been extended to include information on the camera position etc. If a visualization client is started as a visualization slave and connects to the “GeomServer”, it becomes remotely controlled by the master client. All the changes made to the view of the scene on the master client are distributed straight away to all the connected slave nodes via the “GeomServer”.

Here again, the simulation kernel runs on a standard Linux-based cluster, for example. It distributes results to both the master and the slave nodes via the CORBA-based simulation server, as described above. Compared to the version in the original collaborative CSE, no changes to the simulation server have been necessary.

In order to use the visualization client for data produced in a non-interactive way by other CFD packages, we intend to implement an interface for reading those datasets and sending them to the visualization engine.

Application

As indicated in the first section, the range of applications is quite large: from office space to machinery rooms or from cars to carriages and airplane cabins. Due to computer resources, the results of an interactive simulation with a limited resolution can be used to improve the model. An optimized configuration may be posted as a batch job at a higher resolution. By way of an example, we present the model of a train carriage with the simulation of free convection-type flow (in cooperation with Siemens AG, Corporate Technology), (see Figure 5). The buoyant forces result from heaters placed under the seats. With 40 million degrees of freedom the model is still comparatively coarse. It was computed on a Linux cluster within a few hours computing time. The grid generation process took only about 1.5 seconds on a single PC.

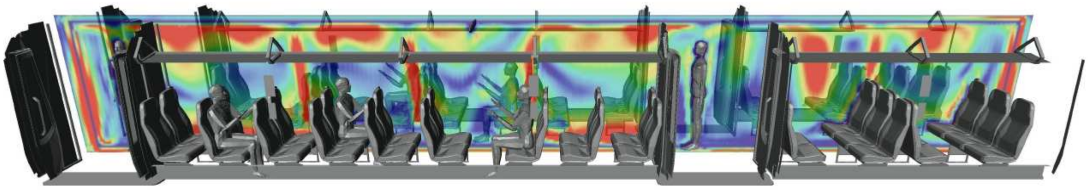

Figure 5: Turbulent convection in a passenger train cabin. Visualization of the average velocity field. Red (blue) colors indicate high (low) flow velocities.

Acknowledgements

The authors are grateful to the Bayerische Forschungsstiftung (Munich, Germany), KONWIHR (Competence Network for Technical High Performance Computing in Bavaria, Germany) and to Siemens AG, Corporate Technology for their financial support. Part of this work is being sponsored by grant No. AZ 630/04 awarded by the Bayerische Forschungsstiftung. Other partners in this project are the Fraunhofer Institute for Building Physics (Holzkirchen, Germany), the companies Mueller-BBM (Planegg, Germany) and Siemens AG (CT, Munich, Germany) as well as the Leibniz Computing Centre of the Bavarian Academy of Sciences and Humanities (Garching, Germany).

Bibliography

[1] Borrmann A., Wenisch P., van Treeck C., Rank E.: Collaborative computational steering: Principles and application in HVAC layout, Integrated Computer-Aided Engineering (ICAE), vol. 13(4): 361-376, 2006.

[2] d'Humières 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, 437-451, 2002.
[3] Krafczyk M., Tölke J., Luo L.-S.: Large-eddy simulations with a multiple-relaxation-time LBE-model, Int. J. Modern Physics B, 17:33-39, 2003.
[4] Lallemand P., Luo L.-S.: Theory of the lattice Boltzmann method: Acoustic and thermal properties in two and three dimensions, Phys. Rev. E, 68, 036706, 2003.
[5] Mezrhab, A., Bouzidi, M. and Lallemand, P.: Hybrid lattice-Boltzmann finite-difference simulation of convective flows, Computers and Fluids, 33, 623–641, 2004.
[6] Mercury OpenInventor Bibliothek, http://www.tgs.com, 2006.
[7] Orbacus CORBA library, http://www.orbacus.com, 2006
[8] van Treeck C.: Gebäudemodell-basierte Simulation von Raumluftströmungen. Dissertation, TU München, Shaker Verlag, ISBN: 3-8322-3375-X, 2004.
[9] van Treeck C., Rank E., Krafczyk M., Tölke J., Nachtwey B.: Extension of a hybrid thermal LBE scheme for Large-Eddy simulations of turbulent convective flows, Computers and Fluids, 35: 863-871, Elsevier, 2006.
[10] van Treeck C., Rank E.: ComfSim - Interaktive Strömungssimulation und lokale Komfortanalyse in Innenräumen. In: Bayerische Forschungsstiftung (Ed.), Annual Report (Jahresbericht) 2005, 66, Munich.
[11] Wenisch P., van Treeck C., Borrmann A., Rank E., Wenisch O.: Computational steering on distributed systems: Indoor comfort simulation as a case study of interactive CFD on supercomputers, Int. J. Parallel, Emergent and Distributed Systems, accepted for publication, 2006.
[12] Wenisch, P. and Wenisch, O.: Fast octree-based Voxelisation of 3D Boundary Representation-Objects, Technical report, Lehrstuhl für Bauinformatik, Technische Universität München, 2004