Fast Simulations of Smoke Transport in Buildings

Wangda Zuo[1,*] and Qingyan Chen[2,3]
[1]Key Laboratory of Three Gorges Reservoir Region' s Eco-Environments under Ministry of Education, Chongqing University, Chongqing 400045, P.R.China
[2]School of Environmental Science and Engineering, Tianjin University, Tianjin 300072, China
[3]National Air Transportation Center of Excellence for Research in the Intermodal Transport Environment (RITE), School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907, USA
[*]Current Employer: Environmental Energy Technologies Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

ABSTRACT

In case of fire in buildings, real-time or faster-than-real-time simulations of airflow and smoke transport can reduce casualties. The simulations should be informative by providing airflow motion, temperature distribution and smoke concentration. By solving the Navier-Stokes equations and transportation equations for energy and smoke, Fast Fluid Dynamics (FFD) model can provide detailed information. If the computation is performed on a Graphics Processing Unit (GPU), the FFD simulation can be faster-than-real-time for a moderate size building with 10^7 grids and $\Delta t = 0.1s$.

Keywords: Fast fluid dynamics, Graphics processing units, Building simulations, Air flow

INTRODUCTION

Computer simulations of air distributions have been widely applied in buildings (Axley 2007; Chen 2009; Megri and Haghighat 2007; Nielsen 2004). Many applications require the simulations to be both informative and fast. For instance, how to manage the smoke transport in buildings in case of fire needs faster-than-real-time simulations. The faster-than-real-time simulation can provide very valuable information for the building evacuation to reduce casualties.

The most popular models for indoor airflow are nodal models and Computational Fluid Dynamics (CFD) models. Nodal models, including multizone models (Axley 2007) and zonal models (Megri and Haghighat 2007), assume that the air and species are uniform in a large space. This homogeneous assumption allows the nodal models to represent flow and species information in a building with a few nodes. Consequently, they need little computing effort. On the other hand, they are unable to describe the characteristics of flow in detail with the limited quantity of nodes. Moreover, the nodal models solve only the mass continuity, energy conservation, and species concentration equations but not the momentum equations (Wang 2007). Therefore, they fail to provide detailed and accurate information about the airflow and species transport.

By numerically solving the Navier-Stokes equations and other transport equations with an enormous number of computing nodes, the CFD can precisely capture the flow features (Chen et al. 2007; Ladeinde and Nearon 1997; Nielsen 2004). However, the CFD simulation usually requires long computing time. For instance, to precisely evaluate the annual energy performance of a small room of 3m×3m×3m with detailed airflow information, a coupled energy-CFD simulation will require at least 150 hours of computing time (Zhai and Chen 2003). Over 99% of the computing time was used by the CFD.

In order to accelerate the CFD simulation, some researchers (Beghein et al. 2005; Crouse et al. 2002; Mazumdar and Chen 2008) used multi-processor supercomputers or computer clusters. The speed was much faster but this approach required expensive computing facilities, a space for installing the computer, and a large cooling system to cool the computer (Feng and Hsu 2004). Hence, the multi-processor supercomputer or computer clusters is luxury for building designers and emergency management teams.

Ideally, one should be able to obtain detailed information about airflow motion, temperature distribution, and species concentration in faster-than-real-time with minimal costs. This investigation explored different approaches to meet that challenge.

FAST FLUID DYNAMICS MODEL

The first approach is the use of Fast Fluid Dynamics (FFD) that is an intermediate model between the nodal and CFD models. The FFD, developed by Stam (1999) for computer flow visualization, can efficiently solve the Navier-Stokes equations (1), energy equation (2) and species transport equations (3):

$$\frac{\partial U_i}{\partial t} = -U_j \frac{\partial U_i}{\partial x_j} + \upsilon \frac{\partial^2 U_i}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial x_i} + \frac{1}{\rho} S_{F,i}, \tag{1}$$

$$\frac{\partial T}{\partial t} = -U_j \frac{\partial T}{\partial x_j} + \alpha \frac{\partial^2 T}{\partial x_j^2} + S_T, \tag{2}$$

$$\frac{\partial C_i}{\partial t} = -U_j \frac{\partial C_i}{\partial x_j} + k_{C,i} \frac{\partial^2 C_i}{\partial x_j^2} + S_{C,i}, \tag{3}$$

where $i, j = 1, 2, 3$, U_i = the ith component of the velocity vector, P = the static pressure of a flow field, and $S_{F,i}$ = the ith component of the source, such as buoyancy force and other external forces. The ν denotes the kinematic viscosity, ρ the density of fluid, T the temperature, α the thermal diffusivity, and S_T the heat source. The C_i is the concentration for ith species. $k_{C,i}$ and $S_{C,i}$ are corresponding diffusivity and source of ith species. Due to the similarity of equations (1), (2), and (3), one can write them in a general equation:

$$\frac{\partial \phi}{\partial t} = -U_j \frac{\partial \phi}{\partial x_j} + k \frac{\partial^2 \phi}{\partial x_j^2} + S + G, \tag{4}$$

where S is the source term and G is the pressure term. Corresponding variables and terms of equations (1), (2), and (3) in equation (4) are given in Table 1.

Table 1 Corresponding terms and variables in equation (4)

Variables	Momentum Equation (1)	Energy Equation (2)	Species Transport Equation (3)
ϕ	U_i	T	C_i
K	ν	α	$k_{c,i}$
S	$S_{F,i}/\rho$	S_T	$S_{C,i}$
G	$-\frac{1}{\rho}\frac{\partial P}{\partial x_i}$	0	0

The FFD method applies a time-splitting method (Ferziger and Peric 2002) to solve the governing equations (4). The purpose of the splitting method is to divide a complex problem (equation) into several simple ones (Ferziger and Peric 2002; John 1982; Levi and Peyroutet 2001) since solving these simple equations is mathematically easy and numerically fast. Then solutions of these simple equations can be integrated into an approximated solution for the complex equation. The splitted equations in the FFD are as follows:

$$\frac{\phi^{(1)}-\phi^{(n)}}{\Delta t}=S, \tag{5}$$

$$\frac{\phi^{(2)}-\phi^{(1)}}{\Delta t}=k\frac{\partial^2\phi^{(2)}}{\partial x_j^2}, \tag{6}$$

$$\frac{\phi^{(3)}-\phi^{(2)}}{\Delta t}=-U_j\frac{\partial\phi^{(2)}}{\partial x_j}, \tag{7}$$

$$\frac{\phi^{n+1}-\phi^{(3)}}{\Delta t}=G, \tag{8}$$

where superscripts (1), (2), and (3) represent temporary variables.

The FFD computes sequentially the above four equations. The source is added through equation (5). Then the FFD calculates diffusion equation (6) by using a first order implicit scheme. After that, advection equation (7) is solved with a semi-Lagrangian solver (Courant et al. 1952). For the momentum equation, the FFD solves pressure equation (8) together with continuity equation by using a pressure-correction projection method (Chorin 1967). It is worth to notice that there is an extra projection step before the advection step in the implemented FFD code, which is to provide a divergence-free velocity field for the semi-Lagrangian solver in the advection equation.

CASE STUDY

Our case study used the forced convection case from Nielsen (1990). Figure 1 shows the sketch of the case, where $L = 3H$. The inlet height, h_{in}, was 0.056 H and the outlet height, h_{out}, was 0.16 H. The Reynolds number was 5000 based on the inlet height and inlet velocity, which can lead to turbulent flow in the modeled room. The experiment was designed to produce two-dimensional flow field. This study employed

a 37 × 37 non-uniform grid and a time step of 0.5s for both the FFD and CFD simulations.

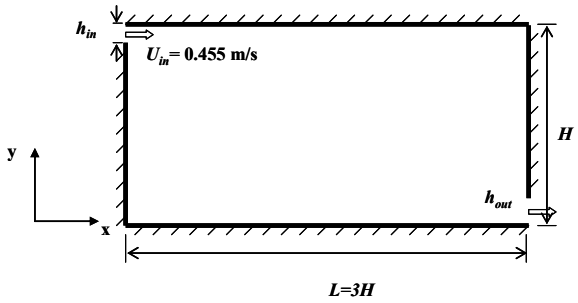
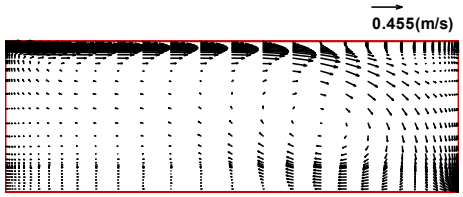
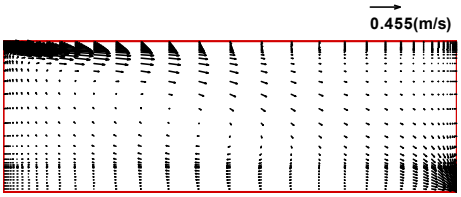


Figure 1. Sketch of the forced convection flow in a room

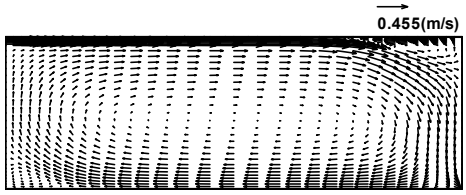
The FFD simulations were performed with laminar flow assumption and the assumption of turbulent viscosity equal to 100 times of laminar viscosity. The CFD simulation assumed the flow to be laminar. Figure 2 compares the velocity fields and Figure 3 the velocity profiles in two vertical sections predicted by the FFD and CFD approaches. The experimental data from the literature on the two sections are also used for comparison. All the simulations gave reasonable results of airflow patterns as shown in Figure 2. Although none of the simulations give precisely results, the FFD models performed not worse than the CFD model.



(a) FFD with laminar assumption



(b) FFD with υ_t=100υ

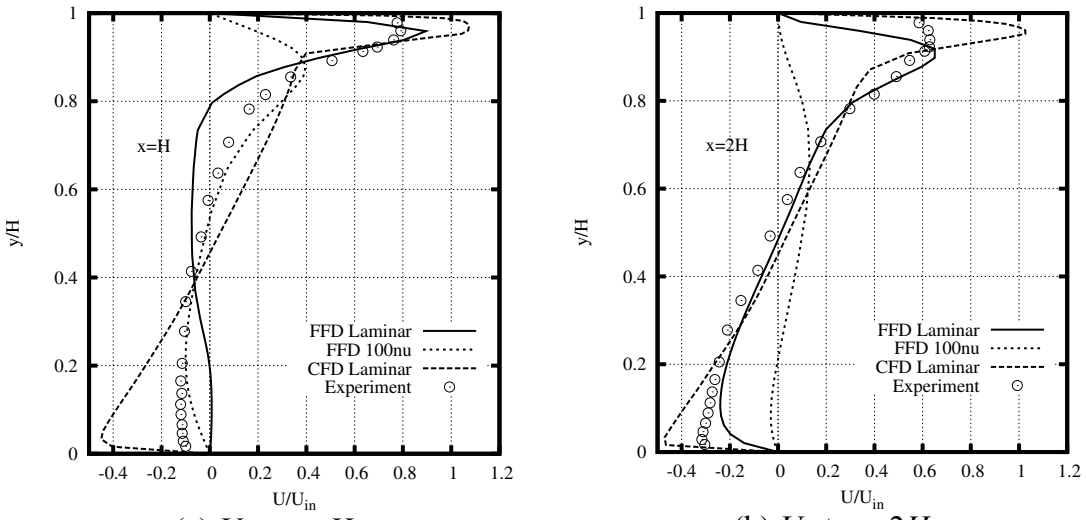


(c) CFD with laminar assumption

Figure 2. Comparison of velocity field predicted by the FFD and CFD in the forced convection

The FFD was also used to simulate smoke transportation. The FFD code has an interactive interface that allows releasing smoke in any location of the simulated domain by simply clicking the mouse. Then the code will calculate the transport of the smoke and visualize it on screen. Figure 4 shows a screen print of white smoke dispersion for this case. The white smoke was released in the upper-left corner close to the inlet. One can clearly see turbulent vortices of smoke in the center of the room. A part of the smoke was expelled from the outlet in the bottom-right corner. The distribution of the smoke looks plausible. Unfortunately, the experimental data of

contaminant concentration is not available for this case so that the accuracy cannot yet be validated.



(a) U at $x = H$ (b) U at $x = 2H$

Figure 3. Comparison of horizontal air velocities predicted by the FFD and CFD with the experimental data in the forced convection

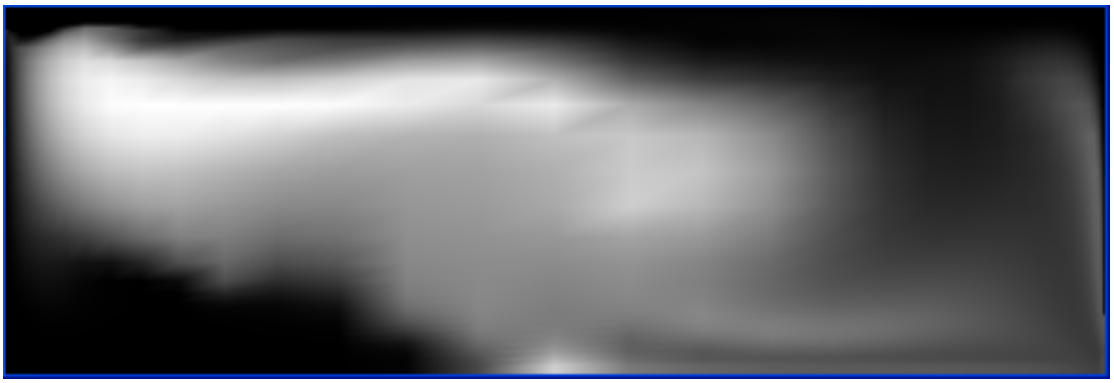


Figure 4. A screen print of white smoke dispersion simulated by FFD

The simulations were performed on a HP workstation with a single INTEL Xeon (TM) CPU at 3.60 GHz. Table 2 lists the computing time ratio of the FFD and CFD simulations. The CFD simulations were carried out by using the two-dimensional version of commercial code FLUENT (http://www.fluent.com). Note that the FFD and CFD used the same amount of grids and time steps for the same case. This investigation defined a computing time ratio, N, as

$$N \equiv t_{physical} / t_{elapsed}, \tag{9}$$

where $t_{physical}$ is the physical time of flow motion and $t_{elapsed}$ is elapsed computing time used by FFD or CFD simulations. When $N = 1$, the simulation is real-time; and when N

> 1, the simulation becomes faster-than-real-time. For this simple case, all the simulations were faster-than-real-time but FFD was 35 times faster than CFD.

Table 2 Computing speed of the FFD and CFD simulations for the forced convection

	FFD,Laminar	*FFD,100υ*	*CFD,Laminar*
N	49.9	48.2	1.37

GRAPHICS PROCESSING UNIT

The computing time can be further reduced by running FFD on Graphics Processing Units (GPU). GPU was originally designed for computer graphics. Its structure is highly parallelized for imaging processing. A GPU can have a few hundreds of processors so it is powerful. It is possible to use GPU for general purpose computing, including linear algebra (Bell and Garland 2008; Ries et al. 2009), signal processing (Tenllado et al. 2008), molecular dynamics (Anderson et al. 2008; Yang et al. 2007), and indoor airflow simulations (Zuo and Chen 2009, 2010).

The knowledge of general parallel programming on multiple CPUs can be applied for the GPU programming, although some details may be different due to the specific structure of GPU hardware. Our investigation adopted the CUDA language (NVIDIA 2007) on a NVIDIA GeForce 8800 GPU. CUDA treats CPU as "host" and the GPU as "device". The host controls the entire program, initializes data, and writes out results. The device conducts parallel computing with initialized data from the host. After the computation, results will be sent back to the host. CUDA further divides the device (GPU) into three levels: grid, block and thread. A GPU consists of grids and each grid includes multiple blocks, which is made up of many threads. A thread is the basic computing unit and a GeForce 8800 GPU can have as many as 12,288 threads running at the same time. For simplicity, our implementation defined only one grid with multiple blocks, which have 256 threads in each block. To associate the threads and mesh data, we assigned only one grid to one thread. Thus, if the number of grids is multiple of 256, the GPU needs the same quantity of threads. This is a balanced allocation. Otherwise, the allocation is not balanced. For example, to carry 257 grids, the program will need 2 GPU blocks with 512 threads in total, although 255 threads are not associated with any grids. Unfortunately, the unbalanced allocation will have serious consequence on the performance.

Considering the computing speed, the FFD on GPU is much faster than that on CPU. When the allocation of grids and threads is balanced, the speed up can be 30 times. Even at an unbalanced situation, the GPU code is still 10 times faster than the CPU version. As a whole, the FFD on GPU can be 350 to 1000 times faster than the CFD on CPU for the forced convection case. In other words, if a CFD simulation on CPU needs 24 hours, the FFD on GPU can provide the same mount of information within 2-5 minutes. With this speed, the FFD on GPU can do a real-time simulation at with half million grids and Δt = 0.1s.

Running the FFD on better GPUs is another way to reduce computing time. The GPU used in our study was purchased in 2007, which is not the fastest nowadays. For instance, a NVIDIA Tesla C2050 GPU (NVIDIA 2010) is 4 times faster than ours. In addition, the performance can be even higher by using multi-GPUs systems. For instance, a Tesla

C2050 GPU system with 4 C2050 GPUs can be around 2 Teraflops for double precision and 4 Teraflops for single precision. Utilizing only 5% of this computing capacity, the FFD on a Tesla GPU system can be about 558 times faster than the FFD on a CPU and 27,900 times faster than the CFD on a CPU. With a time step size of 0.1s, this speed is sufficient for a real-time flow simulation with 10^7 grids, which is enough for a moderate size building.

CONCLUSIONS

This investigation used an FFD model to provide the same detailed information of airflow and smoke distribution in a room as a CFD model. The FFD was 35 times faster than the CFD. If the FFD simulation was performed on a GPU, the speed can be 10 to 30 times faster. As a whole, the FFD on GPU is 350 to 1000 times faster than the CFD on CPU. The speed can be further accelerated by optimizing the implementation and utilizing better GPUs or GPU clusters, so it is possible to do real-time flow simulation for a moderate size building with 10^7 grids and $\Delta t = 0.1$s.

ACKNOWLEDGEMENT

The work of Dr. Wangda Zuo was partially funded by the visiting scholar program at Chongqing University's Key Laboratory of Three Gorges Reservoir Region' s Eco-Environments under Ministry of Education.

NOMENCLATURE

C	Concentration of species
k_C	Diffusivity of species
M	Mass flow rate
P	Static pressure of flow field
S_C	Source of species
$S_{F,i}$	ith component of the source in momentum equation
S_T	Heat source
T	Temperature
U	Horizontal velocity
U_i	ith component of the velocity vector
x	Coordinate at horizontal direction

Greek Symbols

α	Thermal diffusivity; model coefficient
Δt	Time step size
Δx	Mesh size
ϕ	Field variable
υ	Kinematic viscosity of fluid
ρ	Density of fluid

Subscripts

in	Inlet boundary
out	Outlet boundary
perp	Perpendicular

REFERENCES

Anderson, J A, Lorenz, C D, and Travesset, A. 2008. *General purpose molecular dynamics simulations fully implemented on graphics processing units.* Journal of Computational Physics, Vol. 227(10), pp 5342-5359.

Axley, J. 2007. *Multizone Airflow Modeling in Buildings: History and Theory.* HVAC&R Research, Vol. 13(6), pp 907-928.

Beghein, C, Jiang, Y, and Chen, Q. 2005. *Using large eddy simulation to study particle motions in a room.* Indoor Air, Vol. 15(4), pp 281-290.

Bell, N and Garland, M. 2008. *Effcient Sparse Matrix-Vector Multiplication on CUDA*.

Chen, Q. 2009. *Ventilation performance prediction for buildings: A method overview and recent applications.* Building and Environment, Vol. 44(4), pp 848-858.

Chorin, A J. 1967. *A Numerical Method for Solving Incompressible Viscous Flow Problems.* Journal of Computational Physics, Vol. 2(1), pp 12-26.

Courant, R, Isaacson, E, and Rees, M. 1952. *On the Solution of Nonlinear Hyperbolic Differential Equations by Finite Differences.* Communication on Pure and Applied Mathematics Vol. 5, pp 243–255.

Crouse, B, Krafczyk, M, Kuhner, S, et al. 2002. *Indoor air flow analysis based on lattice Boltzmann methods.* Energy and Buildings, Vol. 34(9), pp 941-949.

Feng, W and Hsu, C. 2004. *The Origin and Evolution of Green Destiny*. in *IEEE Cool Chips VII: An International Symposium on Low-Power and High-Speed Chips*. Yokohama, Japan.

Ferziger, J H and Peric, M, 2002. *Computational methods for fluid dynamics*. 3rd, rev. ed. Berlin, New York: Springer.

John, R 1982. *Time-Split Methods for Partial Differential Equations*. Ph.D. thesis, Department of Computer Science, Stanford University,

Ladeinde, F and Nearon, M D. 1997. *CFD applications in the HVAC&R industry.* ASHRAE Journal, Vol. 39(1), pp 44-48.

Levi, L and Peyroutet, F. 2001. *A time-fractional step method for conservation law related obstacle problems.* Advances in Applied Mathematics, Vol. 27(4), pp 768-789.

Mazumdar, S and Chen, Q. 2008. *Influence of cabin conditions on placement and response of contaminant detection sensors in a commercial aircraft.* Journal of Environmental Monitoring, Vol. 10(1), pp 71-81.

Megri, A C and Haghighat, F. 2007. *Zonal Modeling for simulating indoor environment of buildings: Review, recent developments, and applications.* HVAC&R Research, Vol. 13(6), pp 887-905.

Nielsen, P V. 1990. *Specification of a two-dimensional test case*. Aalborg University.

Nielsen, P V. 2004. *Computational fluid dynamics and room air movement.* Indoor Air, Vol. 14, pp 134-143.

NVIDIA, 2007. *NVIDIA CUDA Compute Unified Device Architecture-- Programming*

Guide (Version 1.1). Santa Clara, California: NVIDIA Corporation.

NVIDIA. *Tesla C2050/C2070 GPU Computing Processor*. 2010 [cited; Available from: http://www.nvidia.com/object/product_tesla_C2050_C2070_us.html.

Ries, F, Marco, T D, Zivieri, M, and Guerrieri, R. 2009. *Triangular matrix inversion on Graphics Processing Unit*. in *Conference on High Performance Networking and Computing* Portland, Oregon.

Stam, J. 1999. *Stable Fluids*. in *26th International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'99)*. Los Angeles.

Tenllado, C, Setoain, J, Prieto, M, et al. 2008. *Parallel implementation of the 2D discrete wavelet transform on Graphics Processing Units: Filter Bank versus Lifting.* IEEE Transactions on Parallel and Distributed Systems, Vol. 19(3), pp 299-310.

Wang, L 2007. *Coupling of Multizone and CFD Programs for Building Airflow and Contaminant Transport Simulations*. Ph.D. thesis, Mechanical Engineering, Purdue University, West Lafayette, Indiana.

Yang, J K, Wang, Y J, and Chen, Y F. 2007. *GPU accelerated molecular dynamics simulation of thermal conductivities.* Journal of Computational Physics, Vol. 221(2), pp 799-804.

Zhai, Z and Chen, Q. 2003. *Solution characters of iterative coupling between energy simulation and CFD programs.* Energy and Buildings, Vol. 35(5), pp 493-505.

Zuo, W and Chen, Q. 2009. *Fast parallelized flow simulations on graphic processing units*. in *the 11th International Conference on Air Distribution in Rooms (RoomVent 2009)*. Busan, Korea.

Zuo, W and Chen, Q. 2010. *Fast and informative flow simulations in a building by using fast fluid dynamics model on graphics processing unit* Building and Environment, Vol. 45(3), pp 747-757.