Zuo, W. and Chen, Q. 2010 "Simulations of air distribution in buildings by FFD on GPU," *HVAC&R Research*, 16(6): 785-798.

SIMULATIONS OF AIR DISTRIBUTIONS IN BUILDINGS BY FFD ON GPU

Wangda Zuo, Ph.D.
Student Member ASHRAE

Qingyan Chen, Ph.D.
Fellow ASHRAE

National Air Transportation Center of Excellence for Research in the Intermodal Transport Environment (RITE), School of Mechanical Engineering, Purdue University, 585 Purdue Mall, West Lafayette, IN 47907-2088, USA

ABSTRACT

Building design and operation often requires real-time or faster-than-real-time simulations for detailed information on air distributions. By solving the Navier-Stokes equations and transportation equations for energy and species, Fast Fluid Dynamics (FFD) model can provide detailed information as a Computational Fluid Dynamics (CFD) model. Compared to the CFD, the FFD is 50 times faster with some compromise in accuracy. But the accuracy and speed of the FFD model can be further enhanced by improving its numerical schemes. In addition, it was found that the computing time of the FFD program can be reduced up to 30 times by executing on a Graphics Processing Unit (GPU) in stead of a Central Processing Unit (CPU). Furthermore, the FFD simulation can be accelerated by optimizing the GPU code and by using multiple GPUs. As a whole, it is possible to perform real-time simulation for a moderate size building with 10^7 grids and $\Delta t = 0.1s$ using the FFD on GPUs.

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, to design natural ventilation in a building, the designer needs to know the details of air velocity and temperature. In addition, the simulation should be fast enough to meet the rapid changes in design process. Another example is smoke and air management in case of building fire. If one can simulate detailed smoke distribution faster than real time, it could help the building fire management.

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:

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

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

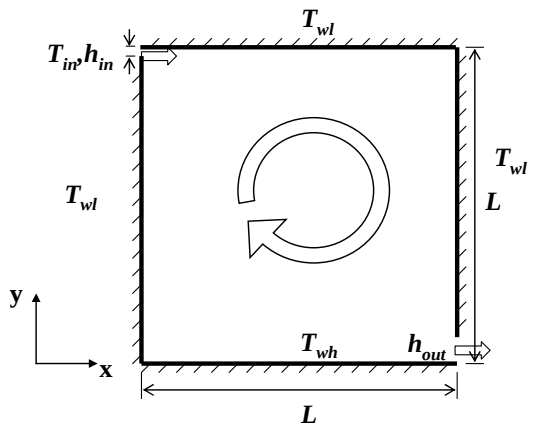
$$\frac{\phi^{(3)} - \phi^{(2)}}{\Delta t} = -U_j \frac{\partial \phi^{(2)}}{\partial x_j}, \tag{7}$$

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

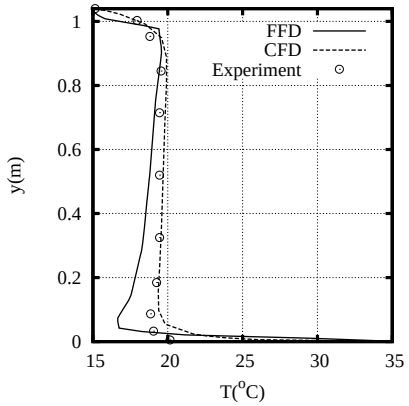
where superscripts (1), (2), and (3) represent temporary variables.

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.

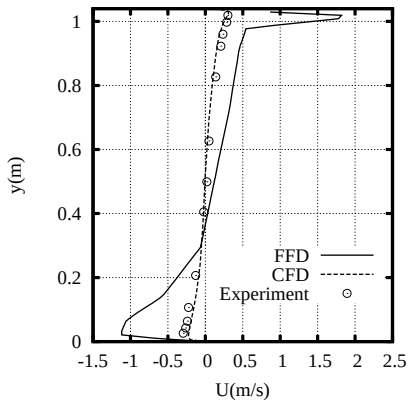
The performance of the FFD has been systematically evaluated by simulating different indoor airflows, including a fully developed plane channel flow (Kim et al. 1987), a forced convection flow (Nielsen 1990), a natural convection flow (Betts and Bokhari 2000), and a mixed convection flow (Blay et al. 1992). As a comparison, the same flows were also computed by using the commercial CFD software FLUENT (www.fluent.com) with standard RNG k-ε model (Yakhot and Orszag 1986). For instance, Figure 1 compares the prediction of FFD and CFD for a mixed convection flow which represents the airflow in a room with mechanical ventilation and floor heating. The gird resolution was 20 × 20 for both FFD and CFD. But grid distributions were adjusted to obtain the best result for each model. As shown in Figure 1(a), the cold air goes into the room through the upper-left corner. The temperature of supply air is at 15 ^{o}C, which is the same as that on the side walls and ceiling. The floor is heated to 35 ^{o}C, so the averaged room air temperature was around 19 ^{o}C. Figure 1(b) compares the predicted temperature distribution at the center of the room by using the FFD and CFD with the same numerical settings. The CFD prediction agrees with the experimental data (Blay et al. 1992) at most measured points. The FFD also captured the mixed feature of the air. However, there were some difference between the FFD prediction and the experimental data. Figure 1(c) compares the horizontal velocity. Again, the CFD results had a good agreement with the experimental data. The FFD got the correct direction of flow direction, but over-predicted the velocity.



(a) Configuration



(b) T at x = 0.52m

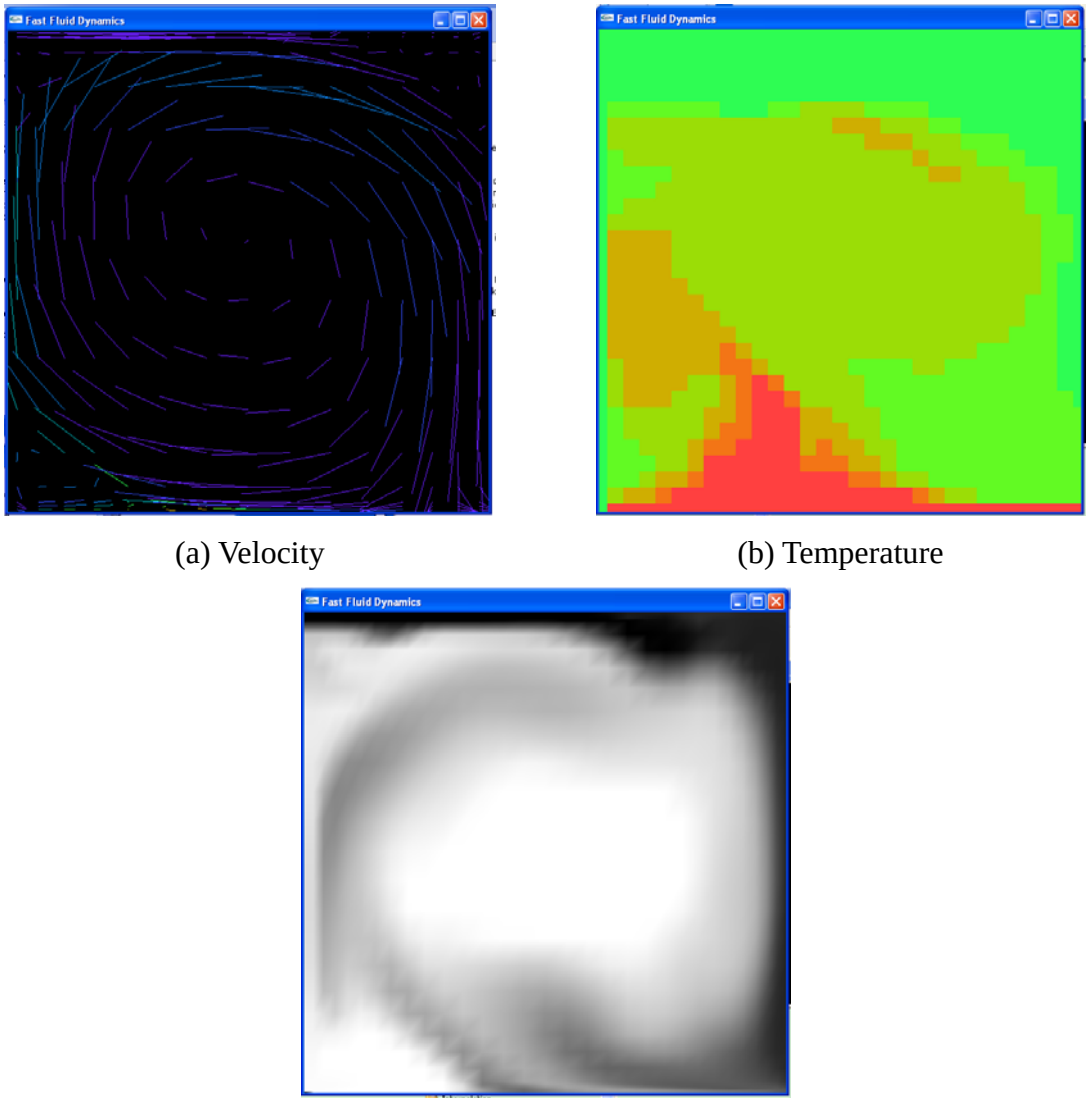


(c) U at x = 0.52m

Figure 1 Comparison of FFD and CFD simulations with experimental data for a mixed convective flow in an empty room.

Although the FFD is not as accurate as the CFD, it can provide more detailed information other multizone or zonal models. In addition, all the information can be visualized online. Similar to the computer games, our FFD program allows users to interact with the program during the simulation, such as releasing contaminats and changing boundary conditions. Figure 2 are the screen shots of the FFD simulation for the mixed convection case. In velocity field window (Figure 2a), there is a large clockwise circulation due to the inertial momentum of inlet jet. Meanwhile, there were also small recirculations near the corners due to the wall influence. Figure 2(b) illustrates temperature distribution that shows the mixing of the hot air (red) from the floor with the cold air (green) from the jet. Blay's experiment only measured velocity and temperature. Our FFD simulation for concentration started with a uniform distribution of white smoke (species) in the room. Then, the smoke was diluted by the fresh supply air as shown in Figure 2(c). The smoke concentration was low at the flow path and high at the center of the large recirculation, which looks plausible. As a

whole, the FFD gives sufficient flow information for conceptual design and emergency management.



(a) Velocity (b) Temperature

(c) Contaminant

Figure 2 Screen shots of the FFD simulations for the mixed convective flow with homogeneous sources of contaminant

Table 2 summarizes the performance of the FFD and CFD in the four studied cases. We defined relative error of simulation as $|(\phi_{sim}-\phi_{exp})/\phi_{exp}|$, which ϕ_{sim} and ϕ_{exp} are simulation and experimental data, respectively. The performance of the model will be ranked as 4 if its relative errors are less than 10% at the majority (>80%) of measured points. If the error is less than 10% to 30%, the rank will be 3. Accordingly, rank 2 is for error less than 30%-50% and 1 for error larger than 50%. The grade performance

average of the FFD model was 2.25/4.0, which means it can capture the general trend of the flow, but not very accurate. It is not surprised since the FFD was proposed to produce a plausible flow in real-time (Stam 1999). On the other hand, the grade performance average of the CFD with RND k-ε model was 3.75/4.0. That is also the reason why the RNG k-ε model is recommended for indoor airflow simulations by literatures (Chen 1995; Zhang et al. 2007). For more details of the FFD model, one can refer to our previous paper (Zuo and Chen 2009b).

Table 2 The grade performance average of the FFD and CFD model for indoor airflows

Model	Channel Flow	Forced Convection	Natural Convection	Mixed Convection	Averaged
FFD	2	3	2	2	2.25
CFD	4	4	3	4	3.75

By sacrificing some accuracy through the numerical scheme, the FFD can gain significant improvements on computing speed. Figure 3 compares the computing time required by the FFD and CFD with the same numerical settings. The time step size was 0.1s. The computing time for the two models linearly varied with grid number. However, the time required by the FFD was only 2% of that by the CFD.

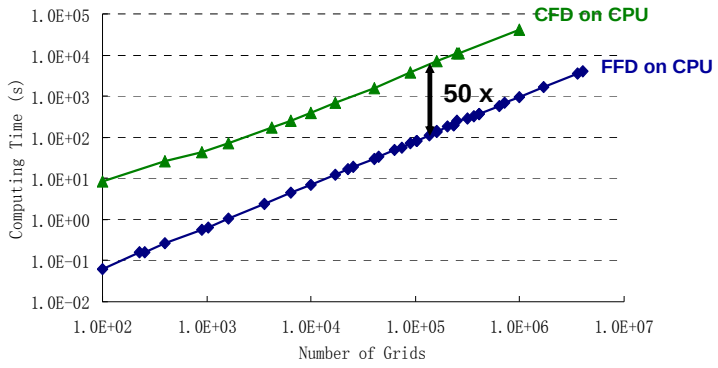


Figure 3 Comparison of the computing time used by the FFD and CFD.

The accuracy and speed of the FFD model can be further improved. For instance, the FFD has significant numerical diffusion due to the linear interpolation used in the

semi-Lagrangian solver for the advection equation. For simplicity, here is shown the one dimensional form of the linear interpolation:

$$\phi(x) = \phi_i + (x - x_i)\frac{\phi_{i+1} - \phi_i}{\Delta x}, \tag{9}$$

where Δx is mesh size, x is between x_i and x_{i+1}, ϕ_i and ϕ_{i+1} are ϕ at x_i and x_{i+1}, respectively.

To reduce the numerical diffusion, some researchers (Fedkiw et al. 2001; Song et al. 2005) tried high order interpolations in the semi-Lagrangian solver. However, none of those approaches was satisfactory (Zuo and Chen 2010b). On one hand, the low order interpolation may introduce numerical diffusion, but can stabilize simulation. A high order interpolation can reduce numerical diffusion, but may lead to numerical dispersion.

To obtain a stable interpolation with low numerical diffusion, we may combine different schemes to obtain a hybrid method. For instance, if the profile monotonously increase or decrease, a high order scheme can be applied to obtain better accuracy. Otherwise, a less accurate but more stable low order scheme may be taken to damp the oscillations.

In this study, we proposed a hybrid scheme by using the first order and third order interpolations. Assuming a uniform grid distribution, the one-dimensional formula of the hybrid interpolation is as follows:

$$\phi(x) = \begin{cases} \phi_i + (x - x_i)\dfrac{\phi_{i+1} - \phi_{i-1}}{2\Delta x} + \dfrac{(x - x_i)^2}{2}\dfrac{\phi_{i-1} + \phi_{i+1} - 2\phi_i}{(\Delta x)^2}, & (\phi_{i+1} - \phi_i)(\phi_{i+1} - \phi_{i-1}) > 0 \\ \phi_i + (x - x_i)\dfrac{\phi_{i+1} - \phi_i}{\Delta x}, & (\phi_{i+1} - \phi_i)(\phi_{i+1} - \phi_{i-1}) \le 0 \end{cases} \tag{10}$$

This study simulate two simple flows to evaluate the linear and hybrid interpolations. One is the transportation of a one-dimensional triangular wave in inviscid fluid. The initial condition of the flow is

$$\phi(x,0) = \begin{cases} x, & 0 \le x \le 0.5 \\ 0, & otherwise \end{cases} \tag{11}$$

Neumann boundary conditions are applied on both sides of the domain:

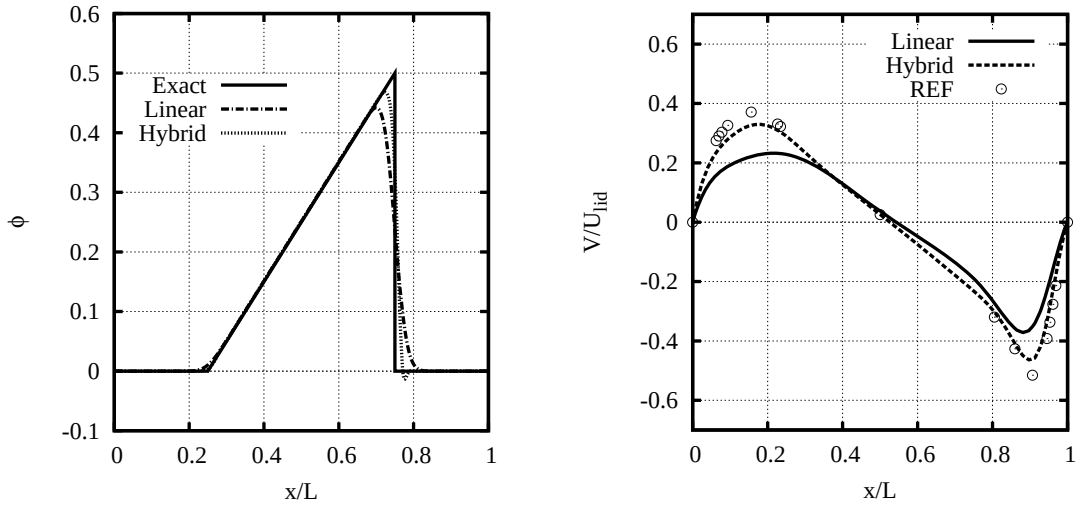
$$\frac{\partial \phi}{\partial x} = 0. \tag{12}$$

If the wave is traveling from left to right at a velocity 1 m/s, then exact solution is

$$\phi(x,t) = \phi(x - t, 0). \tag{13}$$

A one-dimensional uniform mesh with 200 grids was applied for the triangle wave case. The other case is a lid-driven cavity flow, which is well defined in literatures (Erturk et al. 2005; Ghia et al. 1982; Shankar and Deshpande 2000). A uniform mesh with 65 x 65 was used for this flow.

Figure 4 compares the FFD results with the linear and hybrid interpolations for both flows. The FFD with the hybrid interpolation can compute the flow profiles much better than that with the linear interpolation.



(a) (b)

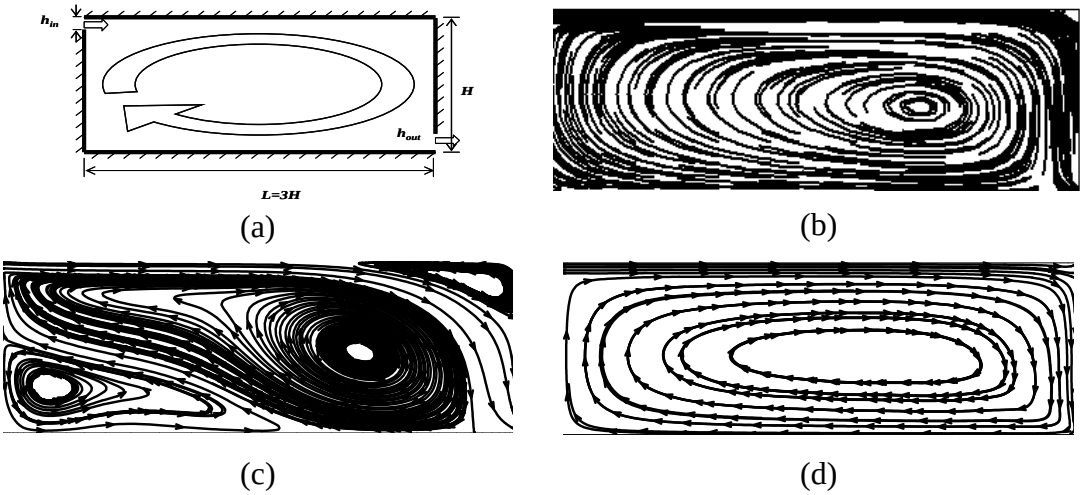
Figure 4 Comparison of computed flow profiles by the FFD with the linear and hybrid interpolation methods. (a) Transport of a tri-angle wave in inviscid fluid at t = 0.25s; (b) Vertical velocity at the center in a square lid-driven cavity at Re = 1000. The data was from Ghia et al (1982).

Besides the numerical diffusion, our studies also showed that the FFD simulation results did not always satisfy mass conservation. One possible reason is that the pressure-correction step in the FFD only corrects the velocities once. Thus, the corrected velocity may not satisfy the mass conservation. On the contrary, the pressure-correction in SIMPLE algorithm (Patankar 1980) requires multiple corrections to achieve a converged solution. We have tried multiple pressure-corrections in the FFD, but have not seen obvious improvements. In addition, the simulation speed dramatically slows down due to extra computing costs on the trial-correction loop. Although the reason for mass unbalance is not clear yet, we can still take a simple approach to force a mass balance over a domain:

$$U_{perp}^{new}\Big|_{out} = \left[1+\alpha\left(\frac{M_{in}}{M_{out}}-1\right)\right]U_{perp}\Big|_{out}, \qquad (14)$$

where M_{in} and M_{out} are total mass flows into and out of the domain, respectively, and α is a correction coefficient. $U_{perp}\big|_{out}$ and $U^{new}_{perp}\big|_{out}$ are the velocities perpendicular to the outlet surface before and after the correction. To make the numerical simulation stable, the recommended value of α is 0.7.

The benefits of applying mass correction (14) can be seen from an example of a forced convection flow in an empty room (Nielsen 1990). As shown in Figure 5(a), the air was injected from the inlet at the left-upper corner and was exhausted from the outlet at the right-lower corner. The CFD results (Zuo and Chen 2009b) obtained with the RNG k-ε model (Yakhot and Orszag 1986) agreed with the experimental data (Nielsen 1990). Thus, they were used as reference here since no streamlines available in experimental data. Figure 5 (c) and (d) are streamlines predicted by the FFD without and with the mass correction. Obviously, the streamlines by the FFD with mass correction (Figure 5d) are much better than those by the FFD without the mass correction (Figure 5c). It is worth to mention that the FFD without mass correction computed similar streamlines as the CFD with *k-ω* SST model did (Rong and Nielsen 2008). It should be an coincidence since FFD and CFD with *k-ω* SST model quite different.



(a) (b)

(c) (d)

Figure 5 Comparison of streamlines for forced convection. (a) Schematic view; (b) CFD with RNG k-ε model; (c) FFD without mass correction; (d) FFD with mass correction.

Besides the accuracy, one can accelerate the speed of the FFD program through optimization of the numerical scheme. For instance, by solving the advection equation first, one can eliminate an extra projection function. This effort can reduce the computing time by 23% (Zuo 2010). In addition, the computing time can be further reduced by optimizing the implementation of the FFD program, such as storing the

coefficient matrices during the computing of matrix equations instead of calculating them every time. This effort can save another 30% time on computing (Zuo 2010). As a whole, the optimization in numerical scheme and model implementation can save around 50% of the computing time.

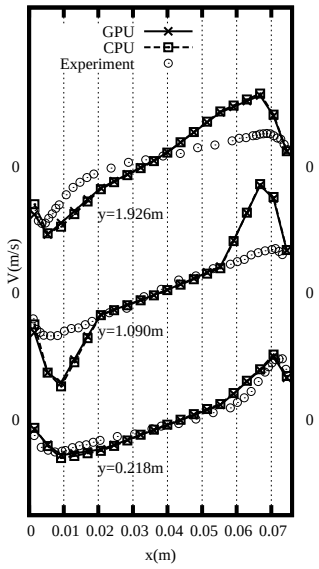
GRAPHICS PROCESSING UNIT

The second approach is to run the FFD in parallel 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. To run flow simulation on GPU was difficult a few years ago because it required special programming skills. With new general purpose GPU programming languages, such as Stream by AMD (2010), CUDA by NVIDIA (2007), Ct by Intel (2010), and Brook by Stanford (Stanford University Graphics Lab 2010), it is possible to expand GPU applications from visualization to 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 2009a, 2010a).

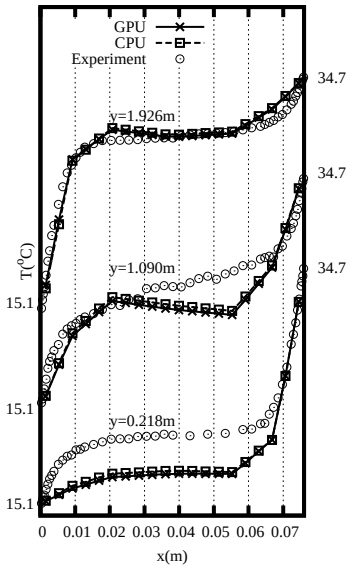
The knowledge of general parallel programming on multiple CPUs can be applied for the GPU programming, although some details may be different due to the specific structure of GPU hardware. Our investigation adopted the CUDA language (NVIDIA 2007) on a NVIDIA GeForce 8800 GPU. CUDA treats CPU as “host” 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.

To evaluate the GPU performance for FFD simulations, FFD simulations with same numerical settings were performed on both CPU and GPU. Figure 6 compares the results for a natural convection flow in a tall cavity. The lines are computed velocity and temperature profiles at various heights across the cavity and the symbols are experimental data (Betts and Bokhari 2000). Although discrepancies existed

between the simulated results and the experimental data due to the defect of the FFD model, the FFD simulations on CPU and GPU yielded the same results. Thus, one can trust the GPU as computing hardware as much as CPU.



(a) Vertical Velocity



(b) Temperature

Figure 6 Comparison of the velocity and temperature profiles predicted by the FFD model on CPU and GPU with the experimental data

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. It is worthy to notice the simulations on CPU and GPU were done in single precision since the GPU in our study did not support double precision.

As a whole, the FFD on GPU can be 500 to 1500 times faster than the CFD on CPU. In other words, if a CFD simulation on CPU needs 24 hours, the FFD on GPU can provide the same mount of information within one minute. With this speed, the FFD on GPU can do a real-time simulation at with half million grids and Δt = 0.1s, which can be sufficient for conceptual design of a small building.

Our current GPU program was to evaluate the possibility of flow simulation on GPU so the code was not optimized yet. As shown in Table 3, even at the best performance, our GPU code utilized only 3% of the computing power offered by the GPU. Thus, there is a great potential to accelerate the simulation speed through optimization. For instance, instead of writing own code to solve matrix equations, one can use more efficient solvers in CUDA library which became available recently.

Table 3 Comparison of the FFD code performance with the peak performance of GPU		
Peak Performance of GeForce 8800 GPU	**FFD Program on GPU**	
	Best Performance	**Worst Performance**
367 GFLOPS	~10 GFLOPS	~4 GFLOPS

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 paper discussed a new technique for informative and fast simulations of air distributions in buildings. This investigation used an FFD model to provide the same detailed information of air distribution as a CFD model. Although the accuracy of the FFD model was not as good as the CFD, the FFD was 50 times faster than the CFD.

The accuracy of the FFD model has been improved through reducing numerical diffusion with a hybrid interpolation method in semi-Lagrangian solver and through enforcing the mass conservation with a correction function. The computing speed of the FFD model can be also further accelerated by modifying the time-splitting method and by optimizing the FFD program.

The other approach to accelerate FFD simulation is running it on GPUs. The FFD model on a GPU produced the same results as that on a CPU, but the speed is 10 to 30 times faster. As a whole, the FFD on GPU is 500 to 1500 times faster than the CFD on CPU. This speed is sufficient for a real-time flow simulation for a small building with half million grids and Δt = 0.1s. In addition, 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 Δt = 0.1s.

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

Advanced Micro Devices. *ATI Stream Software Development Kit (SDK) v2.01*. 2010 [cited; Available from: http://developer.amd.com/gpu/ATIStreamSDK/Pages/default.aspx.

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.*

Betts, P L and Bokhari, I H. 2000. *Experiments on turbulent natural convection in an enclosed tall cavity.* International Journal of Heat and Fluid Flow, Vol. 21(6), pp 675-683.

Blay, D, Mergui, S, and Niculae, C. 1992. *Confined Turbulent Mixed Convection in the Presence of Horizontal Buoyant Wall Jet.* Fundamentals of Mixed Convection, Vol. 213, pp 65-72.

Chen, Q. 1995. *Comparison of Different K-ε Models for Indoor Air-Flow Computations.* Numerical Heat

Transfer Part B: Fundamentals, Vol. 28(3), pp 353-369.

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

Chen, Q, Zhang, Z, and Zuo, W. 2007. *Computational Fluid Dynamics for Indoor Environment Modeling: Past, Present, and Future*. in *the 6th International Indoor Air Quality, Ventilation and Energy Conservation in Buildings Conference (IAQVEC 2007)*. Sendai, Japan.

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.

Erturk, E, Corke, T C, and Gokcol, C. 2005. *Numerical solutions of 2-D steady incompressible driven cavity flow at high Reynolds numbers.* International Journal for Numerical Methods in Fluids, Vol. 48(7), pp 747-774.

Fedkiw, R, Stam, J, and Jensen, H W. 2001. *Visual simulation of smoke*. in *SIGGRAPH 2001*. Los Angeles, CA, USA.

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.

Ghia, U, Ghia, K N, and Shin, C T. 1982. *High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method.* Journal of Computational Physics, Vol. 48(3), pp 387-411.

Integrated Electronics Corporation. *Ct Technology*. 2010 [cited; Available from: http://software.intel.com/en-us/data-parallel/.

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

Kim, J, Moin, P, and Moser, R. 1987. *Turbulence statistics in fully-developed channel flow at low Reynolds-number.* Journal of Fluid Mechanics, Vol. 177, pp 133-166.

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.

Patankar, S V, 1980. *Numerical heat transfer and fluid flow*. Washington, New York: Hemisphere Pub. Corp. ; McGraw-Hill.

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.

Rong, L and Nielsen, P V. 2008. *Simulation with different turbulence models in an annex 20 room benchmark test using Ansys CFX 11.0*. DCE Technical Report No. 46. Department of Civil Engineering, Aalborg University.

Shankar, P N and Deshpande, M D. 2000. *Fluid mechanics in the driven cavity.* Annual Review of Fluid Mechanics, Vol. 32, pp 93-136.

Song, O-Y, Shin, H, and Ko, H-S. 2005. *Stable but nondissipative water.* ACM Transactions on Graphics, Vol. 24(1), pp 81-97.

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

Stanford University Graphics Lab. *BrookGPU* 2010 [cited; Available from: http://graphics.stanford.edu/projects/brookgpu/.

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.

Yakhot, V and Orszag, S A. 1986. *Renormalization-Group Analysis of Turbulence.* Physical Review Letters, Vol. 57(14), pp 1722-1724.

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.

Zhang, Z, Zhang, W, Zhai, Z, and Chen, Q. 2007. *Evaluation of various turbulence models in predicting airflow and turbulence in enclosed environments by CFD: part-2: comparison with experimental data from literature.* HVAC&R Research, Vol. 13(6), pp 871-886.

Zuo, W 2010. *Advanced Simulations of Air Distributions in Buildings*. Ph.D. thesis, Mechanical Engineering, Purdue University, West Lafayette, IN.

Zuo, W and Chen, Q. 2009a. *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. 2009b. *Real-time or faster-than-real-time simulation of airflow in buildings.* Indoor Air, Vol. 19(1), pp 33-44.

Zuo, W and Chen, Q. 2010a. *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.

Zuo, W and Chen, Q. 2010b. *Improvements on the fast fluid dynamics model for indoor airflow simulation.* in *4th National Conference of IBPSA-USA (SimBuild 2010)*. New York, NY, USA.