2017-06-30

# Coupled Simulation of Indoor Airflow, HVAC, Control, and Building Envelope

Wei Tian
*University of Miami*, tianwei19891123@gmail.com

UNIVERSITY OF MIAMI

# COUPLED SIMULATION OF INDOOR AIRFLOW, HVAC, CONTROL, AND BUILDING ENVELOPE

By

Wei Tian

A DISSERTATION

Submitted to the Faculty
of the University of Miami
in partial fulfillment of the requirements for
the degree of Doctor of Philosophy

Coral Gables, Florida

August 2017

UNIVERSITY OF MIAMI

A dissertation submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

# COUPLED SIMULATION OF INDOOR AIRFLOW, HVAC, CONTROL, AND BUILDING ENVELOPE

Wei Tian

Approved:

| | |
|---|---|
| Wangda Zuo, Ph.D.<br>Assistant Professor of Civil, Architectural, and Environmental Engineering | Gang Wang, Ph.D.<br>Assistant Professor of Civil, Architectural, and Environmental Engineering |
| Matthew Jacob Trussoni, Ph.D.<br>Assistant Professor in Practice of Civil, Architectural, and Environmental Engineering | Guillermo Prado, Ph.D.<br>Dean of the Graduate School |

Michael Wetter, Ph.D.
Computational Staff Scientist
Lawrence Berkeley National Laboratory
Berkeley, California

TIAN, WEI (Ph.D., Civil Engineering)

Coupled Simulation of Indoor Airflow, HVAC, Control, and Building Envelope (August 2017)

Abstract of a dissertation at the University of Miami.

Dissertation supervised by Professor Wangda Zuo.
No. of pages in text (174)

Nowadays people spend 90% of the time in indoor. To provide a comfortable and healthy environment for occupants, buildings consumes 40% of the total energy in the world. Due to the inappropriate design of the indoor environment, the problems related to bad indoor air quality caused over $20 billion loss in the US. Then it raises a question on how to improve the indoor environment and decrease the energy consumption in the buildings. One of the strategies available is to utilize the stratified airflow distributions such as mixed mode ventilation. Previously, the coupled simulation between building energy simulation program and computational fluid dynamics (CFD) models was used to study energy and comfort performance of those systems while putting the control dynamics aside.

This research develops the coupled simulation model to study the dynamic systems of non-uniform airflows, HVAC, control, and building envelopes. Fast fluid dynamics (FFD) is chosen to simulate non-uniform airflows since FFD is computationally fast than FFD in simulating the non-uniform airflows. Modelica language, which is equation-based and object-oriented, is used to model HVAC, control, and building envelopes. Then, the coupled simulation model is further ameliorated by adding the

multizone models to expand the application scope of the model from a single zone to a building with multi zones.

To further improve the model for design optimization study, this research improves the computation speed of FFD by parallelizing it using open computing language (OpenCL). We systematically evaluated on the feasibility of using OpenCL to accelerate the airflow simulation using FFD as an example. Though FFD programmed in OpenCL running on different graphics processing units (GPU) may generate different results due to different interpretation of *IEEE-754* standards, the difference is minor to some extents that are negligible in airflow simulation. Running FFD in parallel on a, up to 1139 times speedup is achieved, which is promising to dramatically reduce the time cost for design optimization of the dynamic systems. Regarding the operation optimization, it would be preferable to increase the computation speed of non-uniform airflow simulations by using reduced order models (ROM). We proposed to use in situ adaptive tabulation (ISAT), which differentiates from other conventional ROMS in that it can call a full-scale FFD simulation when the prediction is deemed not accurate. This is critical in the optimization. ISAT executes a FFD simulation if interpolation is deemed inaccurate. In this study, ISAT is trained by using FFD running in parallel on a GPU and once well trained ISAT can answer query points both inside and close to training domain using retrieve actions within a time less than 0.001s for each query. This shows that ISAT can be used to further improve the coupled simulation model to realize operation optimization, such as model predictive control using a non-gradient based optimization.

ACKNOWLEDGEMENT

I would like to first thank Dr. Wangda Zuo, my advisor at the University of Miami, for guiding me throughout the whole Ph.D. research. The process is not always full of joy and I am grateful to him for his patience and mentoring when things cannot work out as expected. I deeply appreciate his generosity of sharing with me his core values. Thanks for everything he has done to make the time at the university so memorable.

I am grateful to have Dr. Michael Wetter from the Lawrence Berkeley National Laboratory as a committee member. In the past few years, I am so lucky to learn lots of research methodologies from him, a world-renowned expert. His advice and comments are critical to the completion of the research.

I want to express my gratitude to Dr. Gang Wang for providing critical comments on the dissertation from engineering perspectives. I have been very fortunate to work with Dr. Wang as teaching assistant for few semesters, and his dedication and commitment to teaching and researching always inspire me.

I want to express my appreciation to Dr. Matthew Jacob Trussoni who serves as a member in my dissertation committee. His inputs are always the keys to broaden the research scope and stimulus to spark new visions.

I would like to acknowledge Dr. Michael D. Sohn from the Lawrence Berkeley National Laboratory and Dr. Stephen B. Pope from the Cornell University for the helpful discussions. My hearted thanks also go to Dr. Mingang Jin for his help in coding and Dr. Liangzhu Wang from the Concordia University for providing me the experiment details.

I am so blessed to know all the amazing people in the Sustainable Building Systems Lab. Among them are Dr. Sen Huang, Reymundo J. Miranda, Dan Li, Thomas A. Sevilla, Qiujian Wang, Yunyang Ye, Yangyang Fu, Guang Zhou, Xing Lu, Xu Han, Danlin Hou, and Jing Wang.

I am indebted to my parents, who probably are the best ones that I can ever ask for. Thanks for always getting my back when I screw up my life badly. The dissertation is to you.

## TABLE OF CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

# Chapter 1 Introduction

## *1.1 State of the Problem*

On average, Americans spend 90% of their time indoors (Kats 2003). Therefore, in order to maintain thermal comfort using HAVC systems, buildings consume about 41% of total energy in the US (Department of Energy 2011). However, the current indoor environment is far from satisfactory. The estimated loss of productivity due to the poor indoor environment is up to 160 billion dollars in the US (Fisk 2000). Thus, it is critical to improve the indoor environment while decreasing the energy consumption.

Ventilation with stratified air distribution may provide better building energy efficiency and indoor air quality (Yuan et al. 1999). Conventional building performance simulation (BPS) programs can rarely handle the non-uniform airflow distribution as they typically adopt multizone models (Axley 2007), which assume that air is well mixed in a zone. To resolve this limitation, some pioneering efforts have been made to couple the computational fluid dynamics (CFD) models with the BPS program (Beausoleil-Morrison 2000; Zhai 2003; Griffith and Chen 2004; Djunaedy et al. 2005b).

However, with the prevalence of those advanced ventilation techniques, current approaches cannot be fully satisfying in terms of the ability to represent the system dynamics and computation speed for some applications, such as design optimization of energy efficiency buildings, model-based control in system operation, fire smokes and hazardous contaminants control when emergency happens

### 1.1.1 Application Requirements

Design Optimization of Building with Advanced Ventilation Techniques

Sustainable building is receiving increasing attention under the context of the energy crisis and global warming, as it is an efficient way to save energy in building section which consumes over two-thirds of the total energy use in the world (Li, Yang, et al. 2013). One of the strategies of designing sustainable building is to use the passive ventilation techniques. In the conventional energy efficient building design, the ventilation with stratified air distribution may provide better building energy efficiency and indoor air quality (Yuan et al. 1999). On the HVAC side, the model needs to be capable of studying the dynamics of the system over a wide range of time period in order to better design the control system. For a longer time period such as one hour, the dynamics of the system needs to be studied in order to adjust system component capacity to save energy. For a shorter time period such as one minute, the dynamics of the system need to be obtained to avoid short-cycling and possible failure of the system. On indoor environment side, the model is expected to provide the non-homogenous airflow and temperature distribution to be fed into the system control. To optimize the design, the model should be essentially fast in computation speed, since usually hundreds of iterations are needed to find the optimal controls.

Design Optimization of Building with Advanced HVAC Techniques

Nowadays lots of prototypes in HVAC systems are emerging and accessible for the design of sustainable buildings. For example, Li, Yang, et al. (2013) reviewed the renewable energy technologies in the zero energy buildings such as photovoltaic,

wind turbines, solar thermal system, heat pumps. Zhao et al. (2011) reported that compared to a conventional HVAC system, the air-conditioning system involving temperature and humidity independent control can achieve 9% energy saving. One of the key components is the liquid desiccant fresh air handling unit. Also, to save energy for the HVAC system, a complex control strategy may be utilized such as chillers stage control (Huang et al. 2016), condensing water temperature set point control (Huang et al. 2017). Apparently, the designer should be able to assemble the complex models and control systems from basic components in a timely manner without infringing on the stability and accuracy. Also, the model should be computationally efficient to emulate the HVAC system in different time scales. For example, the control system may be varying in a time constant of milliseconds while the thermal condition of the wall heat transfer may vary in hours. If applying the small time step size to the whole simulation, the time cost of running the model is not acceptable for the designers.

Operation Optimization of Energy Efficiency Building

With the increasing of computation power, the model predictive control (MPC) has gained more and more awareness nowadays to assist the HVAC operation in buildings for more energy savings (Afram and Janabi-Sharifi 2014). Even with modern computers, it is still very difficult to realize a full-scale simulation in MPC due to the limited time windows for each online optimization. Instead, lots of studies try to use data-driven statistical models or simplified physical models to expedite the simulation with some sacrifice in accuracy (Hazyuk et al. 2012; Oldewurtel et al.

2010). However, those models can hardly keep the fidelity of system and hence difficult to reflect the dynamics of the system. Compared to the design optimization, in the operation phase, time cost can be a serious constraint for computation models, as the operation conditions can be changed frequently. The previous study shows that compared to HVAC system model, the indoor environment model, which is CFD, can be dramatically more time-consuming, especially for large space (Zhai et al. 2002a). Then, the challenge is that how to significantly improve the computation speed of CFD such that hundreds of simulations can be completed within an hour or so.

Fire smokes and hazardous contaminants control

Building safety is a critical issue as people spend over 90% of their time in buildings (Kats 2003). One of the two major threats to building safety is the fire and hazardous species such as liquefied chlorine. United States Fire Administration (Administration 2013) reported 3240 life losses and 11.5 billion dollars as a result of the fire in buildings in 2013. In the USA, trains transport hazardous liquefied gas, such as natural gas, petroleum gas, and chlorine (Hepner and Finco 1995; Havens and Spicer 2005; Scargiali et al. 2005; Luketa-Hanlin et al. 2007; Cormier et al. 2009). An accidental spill of a railcar near an urban area can lead to hazardous exposures and severe harm to health (Van Sickle et al. 2009). Understanding the indoor distribution of the gas resulted from an outdoor spill can improve emergency-response and sheltering-in-place concepts of operation. To assist operators to control the HAVC system and evacuate the occupants, we should build the models that can provide faster-than-real-time simulations of non-uniform indoor airflow and species transport,

as well as the operation of HVAC systems. This further pushes the requirement of an informative and computationally fast airflow simulation model.

### *1.1.2 Limitations of Current Approaches*

The conventional BPS can hardly fulfill the requirement identified in the previous chapter. Some programs such as EnergyPlus (EnergyPlus 2012), DOE-2 (Birdsall et al. 1985), are developed for whole year energy simulation and ideal control algorithm is implemented. Thus, it is hard for them to simulate the dynamics of the systems with complex control strategies (Wetter 2009a).

For some other programs such as TRNSYS (Klein et al. 1976), Matlab/Simulink (Riederer 2005), though it is possible to simulate the dynamics and control of HVAC system, they are usually time demanding in creating the new model for the HVAC system. Wetter and Haugstetter (2006) reported that writing a new model as a TRNSYS TYPE costs a significant amount of time by comparing the BuildOpt (Wetter 2005) and Modelica (Fritzson 1998) buildings library development time. Furthermore, TRNSYS cannot capture the dynamics in a short time scale as it typically uses constant time step size, and instability may occur during that period (Kim et al. 2013b).

More importantly, for indoor airflow simulation BPS often employs multizone models based on the well-mixed assumption (Gu 2007). Apparently, multizone models are not suitable for simulating ventilation systems with non-uniform air

distributions such as strong momentum effect, non-uniform temperature distribution (Wang and Chen 2007a).

To address the limitation imposed by non-uniform airflow simulation, coupled simulations between BPS and CFD tools were proposed to study the energy performance for buildings with stratified air distributions. For instance, Zhai et al. (2002a) found that there was a considerably large difference in predicting the cooling load for an auto racing complex between a standard BPS using the multizone airflow network model and a coupled BPS and CFD. Zhang et al. (2013) studied the performance of natural ventilation by coupling BPS and CFD. Fan et al. (2012) investigated the performance of energy recovery ventilator in a real office with coupled simulation of BPS and CFD. Daoud (2008) demonstrated the dynamic coupling between TRNSYS and CFD program FLUENT by using the air flow across a converging-diverging nozzle as an example.

However, conventional CFD tools are too slow to perform unsteady simulations for a room in real-time or faster-than-real-time manner. Instead, most coupled BPS-CFD coupling studies usually perform only a few steady-state CFD simulations to compute the indoor environment, assuming the indoor environment does not change over a long period. This strategy is appropriate for estimating building energy performance with limited computing time. However, using only a few steady-state CFD simulations is not appropriate for the design and optimization of an

HVAC control for a stratified indoor environment as it does not account for the dynamics of the feedback control.

## 1.2 Objectives

The objectives of this research are three-fold:

1. To develop a coupled model that can simulate the dynamic interaction between the non-uniform airflow, HVAC, building envelope and feedback control for a single zone and a building with multi zones.
2. To further improve the coupled model for the design optimization of a dynamic system consisting of non-uniform indoor airflow, HVAC, building envelope and feedback control.
3. To further improve the coupled model for the operation optimization of a dynamic system consisting of non-uniform indoor airflow, HVAC, building envelope and feedback control.

## 1.3 Methodology

This research proposes to develop the coupled simulation model by applying Modelica for HVAC modeling and sequential Fast Fluid Dynamics (FFD) model for indoor environment simulation in a single zone. Afterward, the model is further improved by adding the multizone models to extend the applications scope to a building with multi zones. Regarding the design optimization, this research proposes to parallelize FFD for indoor environment simulation, which can be significantly speeded up on multi-core devices on a personal computer, such as a powerful graphics processing unit (GPU). To realize the model-based optimization at operation

phase, a reduced order model (ROM) called in situ adaptive tabulation (ISAT) algorithm is used.

### 1.3.1 Modelica

To develop the model for the coupled simulation, we selected the Modelica modeling language (Fritzson 1998) to model the building envelope, HVAC system, and feedback control. Modelica is an equation-based, object-oriented modeling language targeting for the multi-domain dynamic systems. The development of building energy and control systems can be based on the Modelica *Buildings* library (Wetter et al. 2014), which is an open-source, freely available Modelica library. The *Buildings* library has been utilized for the design and performance evaluation of various building energy and control systems (Kim et al. 2013a; Ansuini et al. 2012; Zuo and Wetter 2011; Huang and Zuo 2014).

Compared to conventional BPS, Modelica has some key features that make it stand out to be selected to fulfill the identified needs (Wetter 2009b):

- Capability to simulate dynamic system performance over wide range of period; By creating the model using the governing equations (partial differential equations, ordinary differential equations, algebraic equations) without the need to apply simplifications, the model is kept in high fidelity during the model constructing process. Moreover, it is not necessary to set the time step size as usually done in BPS, Modelica solver can capture all

the dynamics in different time constants using advanced solving techniques.

- Capability to construct and manage complex model; Modelica uses a hierarchy structure that allows building a complex model from assembling multiple basic models. Since Modelica is object-oriented, users can instantiate different applications using the same model, which can reduce significantly time cost for development. Moreover, Modelica has clearly defined interfaces to outside and no nested solver, models are highly extendable and multi-domain models can be combined to construct difficult system models. Finally, Modelica allows equation based non-casual modeling, the modeling process can be easier than BPS which typically uses casual modeling.
- Capability to use advanced numerical solver and perform optimization; After converting the model into an equation set, the numerical solver employing advanced solving techniques can be used to solve those equations such as symbolic manipulating, implicit integration algorithms with adaptive step sizes. Since all the equations are solved simultaneously and a tolerance can be applied, the accuracy is ensured and optimization can then be performed.

### *1.3.2 Fast Fluid Dynamics*

For the non-uniform indoor environment simulation, we choose the FFD program (Zuo and Chen 2009c; Jin et al. 2012c). FFD solves the same Navier-Stokes

equation and other governing equations as CFD does. However, by employing different numerical algorithms and sacrificing some accuracy, FFD has been shown to be around 50 times faster than its counterpart CFD (Zuo and Chen 2009). In addition, parallel FFD simulations on a NVIDIA GPU, which is realized by using CUDA (NVIDIA 2007), achieves another 30 times speedup (Zuo and Chen 2010a). Consequently, this results in a total speedup of 1,500 times over CFD (Zuo and Chen 2010a). The FFD program has been validated and used to study various airflows inside and around buildings (Zuo and Chen 2010c, 2010b; Jin, Zuo, et al. 2013; Jin, Chen, et al. 2013; Jin et al. 2012a; Zuo and Chen 2007).

To conclude, as an intermediate model between the multizone model and the CFD model, FFD realizes a good balance in accuracy and speed, which makes it preferable for simulation of building system dynamics. If further accelerating the FFD simulation using parallel computing techniques, FFD is promising to address the design optimization of the non-uniform indoor environment with HVAC systems. For operation optimization (model based control) which usually requires instantaneous prediction of the airflow, it is necessary to use ROMs that are trained by outputs from FFD simulations ran offline (Kolokotsa et al. 2009; Hazyuk et al. 2012).

### *1.3.3 In Situ Adaptive Tabulation*

A common approach for generating ROM is to use a regression model with a limited number of inputs in order to construct the data-driven ROMs based on pre-calculated CFD results (Chen and Kooi 1988). However, they can rarely reflect the

dynamics of a full order CFD model. On the other hand, ROMs can be built by using the Principal Orthogonal Decomposition method to extract important features (snapshots) of the flow and then project them to a Linear Time Invariant system (Li, Su, et al. 2013). Such ROMs can partially maintain the dynamics of the full order CFD model. Although it can be time-consuming to run various CFD simulations to generate training data, the trained ROMs can compute the solution almost instantaneously by either interpolating or extrapolating using an existing data set. However, conventional ROMs can only perform well when the inputs are within or near the training domain. Consequently, if the inputs are too far outside the training domain, the ROMs may resolve them without any guaranteed accuracy (Stockwell and Peterson 2002).

Obviously, it is too expensive to train a ROM for a domain which includes all the possible inputs of the application. Therefore, to overcome this drawback of conventional ROMs, we propose to a fast and self-learning indoor airflow simulation method. The idea is that we will train the ROM within a domain in which the system is most likely to operate. If the trained ROM cannot project the solutions accurately, a full-scale CFD simulation will be executed. The newly generated data from the CFD simulation will then be used to enlarge the training domain for the ROM.

To realize the proposed fast and self-learning airflow simulation method, we selected an in situ adaptive tabulation (ISAT) algorithm and the FFD model. ISAT is a general function approximation method. ISAT was originally proposed to speed up

combustion simulations (Pope 1997). It stores key simulation data in a data table and linearly interpolates the solutions from the table if the inputs are within the region where the interpolation accuracy is guaranteed. Otherwise, it executes a full-scale simulation to obtain the solution.

## 1.4 Dissertation Outline

This dissertation reports the research to develop coupled simulation models based on HVAC and non-uniform airflow models to simulate dynamics of HVAC systems and indoor environment for design and operation optimization. It is divided into seven chapters. Chapter 1 is a state of the problem for the system dynamics and indoor environment modeling. Chapter 2 is an overview of the coupled simulation between BPS and CFD. The outline of other work is sketched in Figure 1-1.

Chapter 3 introduces a coupled simulation model between Modelica and sequential FFD. Chapter 4 introduces the improvement of the coupled simulation model by adding multizone models. To improve the computation speed of non-uniform airflow simulation for design optimization, study on parallel FFD programmed in OpenCL is presented in Chapter 5. To further improve the computation speed of non-uniform airflow simulation for operation optimization, a ROM ISAT which is coupled with the parallel FFD is introduced in Chapter 6. Chapter 7 introduces conclusion and future research.

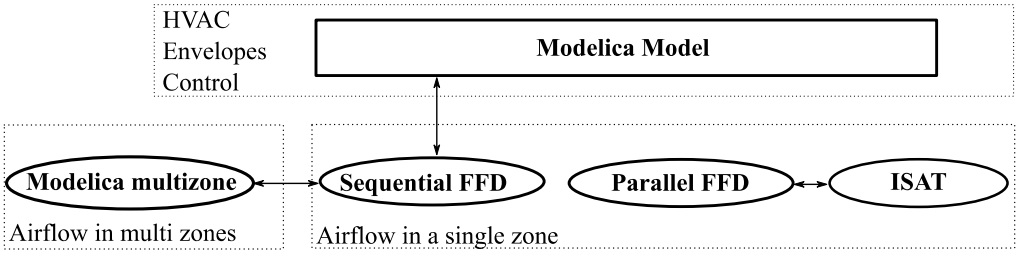



*Figure 1-1 Structure of the dissertation*

# Chapter 2 Literature Review on Coupling Building Energy and Non-Uniform Airflow Simulation

## *2.1 Background*

Due to the assumptions associated with simple multizone airflow models that are employed in typical BPS, most BPS program may not provide satisfactory results for some applications that involve inhomogeneous airflow distribution. For instance, EnergyPlus, a common BPS software, may not perform well in predicting localized comfort, the contaminate distribution within an occupant zone, or the control of HVAC systems (Crawley et al. 2008). In addition, multizone airflow models are often not suitable to simulate non-uniform airflow distribution within a zone. Moreover, it is challenging for a BPS software to predict the correct convective heat transfer coefficient (CHTC). Most BPSs estimate the CHTC with empirical formulas assuming the room air in the room is instantaneously mixed. As shown by Lomas (1996), the difference between the estimated annual heating energy using four different empirical formulas could differ by as much as 27%. These potential limitations of BPSs may be resolved by using computational fluid dynamics (CFD) models.

On the other hand, CFD also has its own technical limitations. CFD usually uses idealized static boundary conditions, such as fixed supply airflow rate and temperature, fixed wall temperature or heat flux through the wall. However, the actual boundary conditions are changing with weather condition and operation schedule of HVAC system that must be obtained from a BES (Djunaedy et al. 2003).

As a result, it is possible to couple the BPS and the CFD to provide missing information when a single program fails to do. This report reviews literature dealing with the coupled simulation between CFD with BPS.

## 2.2 Exchanged Data

This section is to summarize methods that have been reported to exchange information, data, and model predictions between BPS and CFD, and discuss the benefits of the resulting simulation performance.

The information commonly transferred from a BPS to a CFD include interior surface temperature of walls (Zhai et al. 2002b; Zhang et al. 2012a), boundary conditions at openings, including outdoor airflow velocity, pressure and temperature (Ohba and Lun 2010; Wang and Wong 2008), heat flux or air-conditioning load (Zhai et al. 2002b; Fan and Ito 2012), supply airflow rate, temperature, humidity or pressure at supply diffuser (Fan and Ito 2012; Ascione et al. 2012), and temperature or pressure at the outlet (Fan and Ito 2012; Ascione et al. 2012). The possible information that may be transferred from the CFD to BPS include: Indoor temperature distribution (Zhai et al. 2002b; Ascione et al. 2012), CHTC (Zhang et al. 2012b; Rong et al. 2011), flow rate through the openings, such as window, doors, outlet of the atrium (Zhang et al. 2012a; Wang and Wong 2008; Pan et al. 2010), and concentration of pollutants (Goldsworthy 2012; Wang and Chen 2008).

The selection of exchanged data can also impact the convergence, stability and computing speed of the coupled simulation. As concluded by Zhai and Chen (2005),

sending the enclosure interior surface temperature from BPS to the CFD and return the CHTC and indoor air temperature gradients to the BPS can improve the accuracy of a simulation capability dramatically.

## 2.3 Data Synchronization Strategies

We divide the strategies for data synchronization between CFD and BPS into three major categories and describe them in subsection 2.3.1 to 2.3.3.

### 2.3.1 Static Synchronization Strategy

In the static coupling strategy, the CFD exchanges data with the BPS for only a few times, whether manually or automatically, in two methods: one-way and two-way exchange.

- One-way static synchronization

The BPS transfers data to the CFD at some specific time. Figure 2-1 shows the workflow of one-way static synchronization. The rectangle represents CFD and the ellipse represents the BPS. The BPS runs continuously while the CFD runs and receives the data only at a few moments. Since the CFD needs significantly more computing time than the BPS, the one-way static synchronization can greatly reduce the total simulation time of a coupled simulation (Zhai et al. 2002b). This method can be used when two programs are somewhat insensitive to the timing and the amount of exchanged data.

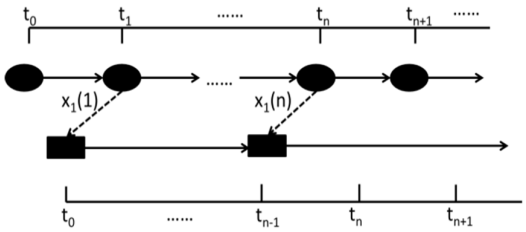



*Figure 2-1 One-way static synchronization*

- Two-way static synchronization

The major difference between the one-way static synchronization and the two-way static synchronization (Figure 2-2) is that the latter also sends the output of CFD to the BPS (Zhai et al. 2002b). The "feed-back/feed-forward" exchange is conducted in order to update the CHTC in the BPS if it is significantly different from that calculated by CFD.

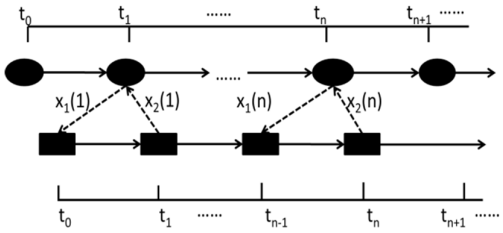



*Figure 2-2 Two-way static synchronization*

### 2.3.2 *Dynamic Synchronization Strategy*

In dynamic synchronization strategies, two programs exchange data with each other at each predefined time step which indicates the communication frequency. Dynamic synchronization strategy exchanges data more frequently than the static ones so that the data synchronization typically has to been implemented

automatically. We can further divide the dynamic synchronization into quasi-dynamic synchronization and fully dynamic synchronization.

- Quasi-dynamic synchronization strategy

Quasi-dynamic synchronization (Zhai et al. 2002b) is also called loose coupling (Trcka et al. 2007) or "ping pong" coupling (Trcka and Hensen 2006). It requires the two programs to conduct one exchange of data at every time step. The Quasi-dynamic strategy has two sub-categories: loose quasi-dynamic strategy and cross quasi-dynamic strategy (Trčka et al. 2009). In the loose quasi-dynamic strategy (Figure 2-3), the BPS transfers data to the CFD which in turn runs the former time step and returns its output to the BPS. In the cross quasi-dynamic strategy (Figure 2-4), the BPS and the CFD run simultaneously and exchange data at the end of each time step. The recovered data are used for the next time step.

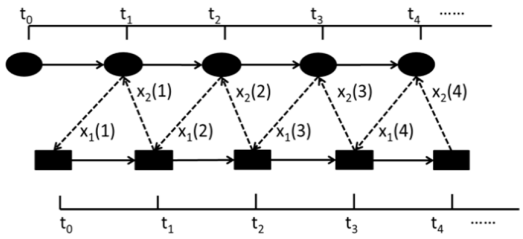



*Figure 2-3 Loose quasi-dynamic synchronization*

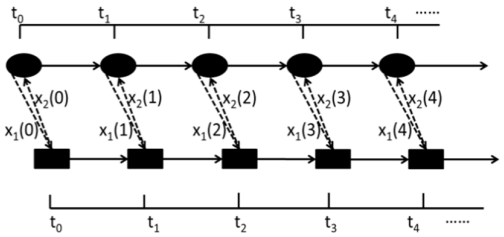



*Figure 2-4 Cross quasi-dynamic synchronization*

- Fully dynamic synchronization strategy

Fully dynamic synchronization strategy (Zhai et al. 2002b) is also called onion coupling (Trcka and Hensen 2006) or strong coupling (Trčka et al. 2009). It requires two programs to conduct multiple data exchanges within every time-step until convergent solutions are achieved for both simulations (Figure 2-5). Compared to the quasi-dynamic synchronization, the fully dynamic synchronization is more accurate but significantly slower, which can be a severe limitation for many applications (Zhai and Chen 2003).

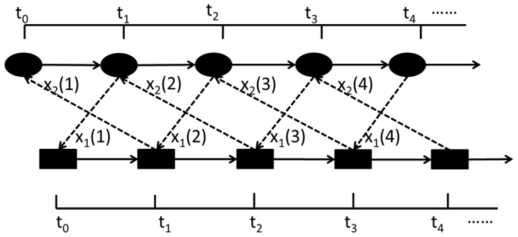



*Figure 2-5 Fully dynamic synchronization*

### 2.3.3 Bin Synchronization Strategy

The bin synchronization strategy (Zhai and Chen 2005) is also called virtual dynamic strategy (Zhai et al. 2002b). It integrates indirectly the BPS with the CFD through an intermediary, such as a database (Zhai et al. 2002b), a neural network

(Rong et al. 2011), or a fitted formula (Zhang, Hiyama, et al. 2013; Hiyama and Kato 2011; Pan et al. 2010). The intermediary is established based on pre-computed CFD simulations in typical scenarios (Figure 2-6). For some typical scenarios the CFD will iterate with the BPS, and the results will be transferred to a database, with which the BPS then is coupled to simulate the building energy consumption. With no CFD run during the coupling, the bin synchronization strategy is computationally faster than the dynamic synchronization. However, the accuracy of the bin synchronization strategy may significantly drop if the flow conditions are outside the range of pre-computed CFD simulations (Zhai et al. 2002b).

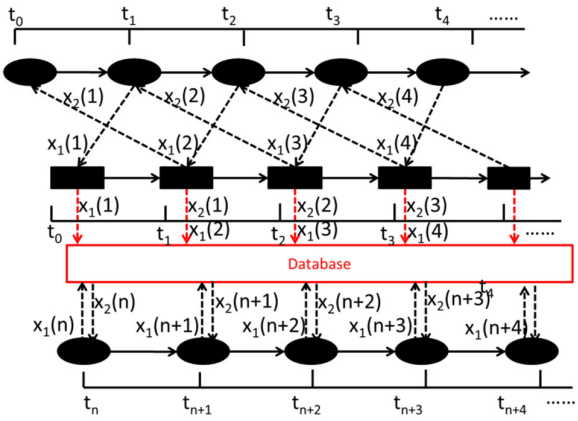



*Figure 2-6 Bin synchronization*

## 2.4 Software Architecture

The software architecture focuses on how to implement the software coupling. In general, there are two methods: internal coupling and external coupling.

### 2.4.1 *Internal Coupling*

The internal coupling approach adds physical models of BPS, such as the envelope heat transfer model and the envelope radiation model, into the CFD. As a result, the new integrated CFD program solves all the governing equations simultaneously until it reaches a converged solution. There are two major limitations in the internal coupling approach (Djunaedy et al. 2003):

- Solving heat transfer through the envelope by the CFD is computationally demanding due to the different time scales in the physical process. The heat transfer through the envelope may take hours while the airflow changes happen in a few seconds.
- It is difficult to obtain a converged result due to the difference in the stiffness of fluid and solid equations (Chen et al. 1995).

### 2.4.2 *External Coupling*

Two methods for external coupling are reported in the literature. The first one is called a discontinuity mechanism, which is defined as "exchanging data between two programs sequentially, where a model preprocessor transforms the output of one program into the input for a slave program after the master program completes its simulation" (Djunaedy et al. 2005a). The other one is called a continuity mechanism, by which two programs are called separately and run in parallel (Trcka and Hensen 2006). The external coupling has at least four advantages as to the internal coupling (Djunaedy et al. 2005a):

- It is much faster than the internal coupling (Djunaedy et al. 2004).
- It can take the advantage of the state-of-art technology in either program as there is no need to rewrite the code.
- The program can be optimized individually in order to solve some specific problems.

To implement data exchange by external coupling, researchers have developed different methods, such as using a self-developed interface for direct coupling, a data exchange platform, and a standard interface. The main function of the self-developed code is to transform the output from one program into a recognizable pattern for the other program (Liping and Hien 2007; Fan and Ito 2012).

The data exchange platform allows programs to exchange data with other programs after connected to the platform. Trčka et al. (2009) presented several data exchange platforms for co-simulation of building performance. A popular approach, the Building Controls Virtual Test Bed (BCVTB) (Wetter 2010) is one of the more advanced methods for building performance simulation. However, it does not provide any links to commercial CFD software. It is necessary to develop a code to bridge the CFD and BCVTB, such as FLOW+ which is used to connect FLUENT to BCVTB (Zhang et al. 2012b).

Two standard interfaces are developed for coupling CFD and the BPS: Functional Mockup Interface (Blochwitz et al. 2011) and building product model based on International Standard Organization standard as used and demonstrated in the literature (Lydon et al. 2005).

## 2.5 Use Cases

The coupled simulation of BPS and CFD can be used for various applications. It was used to evaluate the performance of the advanced indoor ventilation method, such as personalized ventilation, and natural ventilation (Zhang et al. 2012b). It was adopted for the design of the advanced air-conditioning methods, such as underfloor heating with a top return (Fan and Ito 2012) and local thermal environment control (Steeman 2009). Moreover, it was applied in the study of the thermal performance of double-skin facades (Zeng et al. 2012), double-skin wooden roof (Villi et al. 2009), and membrane (Devulder et al. 2007) where flow within and around complex geometry is involved. Furthermore, Goldsworthy (2012) used it to investigate mechanical ventilation for smoke control.

## 2.6 Conclusion

By coupling CFD and BPS, we can obtain more complementary and accurate information about the indoor environment and building energy system performance than a single program running on its own. The exchange data can be synchronized by mainly three ways: static coupling, dynamic coupling, and bin coupling. These strategies vary in time cost on implementation, simulation speed, and accuracy. To implement the data exchange between the two programs, internal and external coupling are feasible.

The coupled simulation can achieve satisfactory simulation for some applications. However, the coupled simulation still needs to be improved for the design and performance evaluation of indoor environment control. For instance, how

to control the HVAC system to prevent the spread of smoke when there is a fire emergency in buildings. This requires a fast computing speed for indoor environment simulation and dynamic simulation capability for HVAC control. The fast fluid dynamics (Zuo and Chen 2009c) model is about 50 times faster than the CFD and can provide sufficient results for the smoke control. The Modelica models are dynamic and well-suitable for HVAC control (Wetter et al. 2013). Coupling the FFD and Modelica may enable a fast and dynamic simulation for the indoor environment control.

# Chapter 3 Coupling Indoor Airflow, HVAC, Control and Building Envelope Heat Transfer in the Modelica Buildings Library

## 3.1 Background

To address the simulation of the dynamic systems of HVAC, control, building envelopes and non-uniform airflows in a single zone, it is necessary to couple the Modelica model with the sequential FFD model. This chapter is structured as follows: The next section introduces the mathematical algorithms for data exchange in the coupled simulation between the FFD program and the Modelica *Buildings* library. We present the implementation of the FFD programs and Modelica models. Next, the accuracy is quantitatively evaluated using a case of non-isothermal flow with stratified distribution and qualitatively studied using the other two cases with feedback control. We also measured the computing time for all three cases.

## 3.2 Mathematical Algorithms for Data Exchange

### 3.2.1 Data Synchronization

To exchange the data between FFD and Modelica during the coupled simulation, we used a data synchronization strategy with a zero-order hold of the respective input signals. The zero-order hold means that the program holds the received data constant until the next synchronization time step. To reduce the computing time, the data exchange is performed only once for every synchronization. This synchronization strategy is semantically equivalent to the one used by the Building Controls Virtual Test Bed (BCVTB) (Wetter 2011). The BCVTB is a

middleware used to facilitate the data exchange between two programs while we, on the other hand, applied direct data exchanges to reduce the overhead of passing information through the middleware.

Figure 3-1 illustrates our data synchronization strategy. At time step $t_n$, FFD sends data $x_1(t_n)$ to Modelica and Modelica sends data $x_2(t_n)$ to FFD. The $x_1(t_n)$ and $x_2(t_n)$ are then kept constant in each program that receives the data until the next synchronization point. Each program may use smaller time step sizes ( $\Delta t_{ffd}$ or $\Delta t_{mod}$) for its own integration between synchronization points. It is possible that $\Delta t_{mod}$ will vary during the simulation since it is determined by an adaptive time step integration algorithm.

The above data synchronization strategy was implemented in the FFD program and the Modelica *Buildings* library using a master-slave method. The Modelica is the master of the coupled simulation and FFD is the slave. Modelica defines the coupled simulation period and the next synchronization point. It also launches and terminates the FFD simulation.

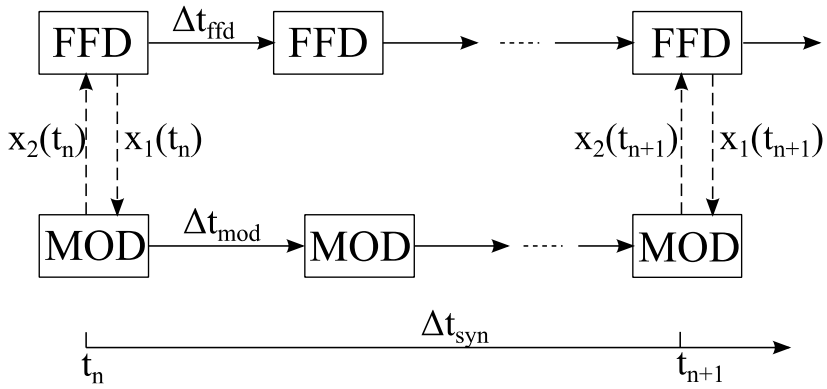



*Figure 3-1. Data synchronization between FFD and Modelica.*

### 3.2.2 Exchanged Physical Quantities

This section describes the exchanged physical quantities between the FFD program and the Modelica models. Compared to coupling the CFD and conventional building energy simulation programs, a major challenge of coupling FFD and Modelica models is that in Modelica, flow directions in the HVAC system can reverse based on the computed pressure difference. Therefore, an air inlet in FFD may become an outlet if the room pressure is higher than the supply air duct pressure, and vice-versa. Thus, the FFD program has to be able to change boundary conditions for the inlet and outlet during the simulation. This is achieved by two steps: First, the FFD program checks the newly received mass flow rates at all inlets and outlets at the synchronization point. Second, the FFD sets the "inlet" boundary condition for those having positive mass flow rates and the "outlet" for those with negative values. The new boundary conditions are then applied to the FFD simulation until next synchronization point. The following part presents the detailed implementation.

#### 3.2.2.1 Fluid Ports

In the Modelica *Buildings* library, the fluid flow into and out of models is modeled using fluid ports. These fluid ports include variables for pressure, mass flow rate, enthalpy, mass fractions (such as water vapor), and optional trace substances (such as carbon dioxide) that are carried by the mass flow. The fluid ports in the Modelica models correspond to the inlet and outlet boundaries in FFD. The Modelica fluid port implementation allows the direction of the mass flow to reverse in order to satisfy the pressure and flow equations. Therefore, in the FFD program, air inlets or outlets need to be dynamically assigned according to the direction of the flow during the coupled simulation.

For the inlet fluid port, the Modelica model defines inlet boundary conditions for FFD. At the time of the data exchange, $t_n$, FFD converts the averaged mass airflow rate at the inlet received from Modelica to the inlet velocity $u_{in}(t_n)$. FFD assumes a uniform velocity distribution on the inlet surface. Hence

$$u_{in}(t_n) = \frac{1}{\rho S_{in} \Delta t} \int_{t_{n-1}}^{t_n} \dot{m}_{in}(t) dt, \tag{1}$$

where $\rho$ is the fluid density, $S_{in}$ is the inlet surface area, and $\Delta t = t_n - t_{n-1}$ is the time interval between two data exchanges. In addition, Modelica sends FFD the temperature, concentration of species, and trace substances at the inlet by using their corresponding time-averaged quantities at the Modelica fluid ports.

For the outlet fluid port, FFD defines the boundary conditions for the Modelica models. The FFD computes a time averaged mass flow rate as

$$\dot{m}_{out}(t_n) = \frac{\rho}{\Delta t}\int_{t_{n-1}}^{t_n}\int_{S_{out}} u_n(s,t)ds\,dt, \tag{2}$$

where $u_n(s,t)$ is the velocity normal to the mesh surface *s* at the outlet and $S_{out}$ is the total outlet area. The time averaged air temperature at the outlet $T_{out}(t_n)$ is computed as

$$T_{out}(t_n) = \frac{\rho}{\dot{m}_{out}(t_n)\Delta t}\int_{t_{n-1}}^{t_n}\int_{S_{out}} u_n(s,t)T(s,t)ds\,dt, \tag{3}$$

where $T(s,t)$ is the air temperature on the mesh surface. Other scalar variables, such as mass fraction and trace substances concentration, are calculated similarly.

#### 3.2.2.2 Walls and Windows

For the FFD simulation, thermal boundary conditions of solid surfaces, such as surfaces of walls and windows, can either be a given temperature or a given heat flux. In our implementation, if Modelica provides to FFD the time-averaged temperature of a solid surface $T_{sur}(t_n)$ as

$$T_{sur}(t_n) = \frac{1}{\Delta t}\int_{t_{n-1}}^{t_n} T(t)dt, \tag{4}$$

FFD will compute the surface heat flux $\dot{q}_{sur}(s,t)$ and provide Modelica the heat flow rate $\dot{Q}_{sur}(t_n)$ as

$$\dot{Q}_{sur}(t_n) = \frac{1}{\Delta t}\int_{t_{n-1}}^{t_n}\int_{S_{sur}} \dot{q}_{sur}(s,t)\, dsdt. \quad (5)$$

Alternatively, if Modelica computes the time-averaged heat flow rate $\dot{Q}_{sur}(t_n)$ as

$$\dot{Q}_{sur}(t_n) = \frac{1}{\Delta t}\int_{t_{n-1}}^{t_n} \dot{Q}_{sur}(t)dt, \quad (6)$$

FFD will convert it to a heat flux $\dot{q}_{sur}(t_n)$ using

$$\dot{q}_{sur}(t_n) = \frac{\dot{Q}_{sur}(t_n)}{S_{sur}}. \quad (7)$$

In addition, FFD computes the time-and-surface-averaged temperature $T_{sur}(t_n)$ as

$$T_{sur}(t_n) = \frac{1}{\Delta t}\int_{t_{n-1}}^{t_n}\frac{1}{S_{sur}}\int_{S_{sur}} T(s,t)\, dsdt. \quad (8)$$

### 3.2.2.3 Sources

For internal heat sources, our current implementation assumes that the heat flow rate $\dot{Q}_{sou}(t_n)$ that is injected into the space to be uniformly distributed. Hence, the heat flow rate in FFD is

$$\dot{q}_{sou}(t_n) = \frac{\dot{Q}_{sou}(t_n)}{V}, \quad (9)$$

where V is the volume of the room air.

Furthermore, if a heat source needs to be modeled at a certain location, such as for computing the plume caused by a person, we can use one or several surfaces and prescribe their temperature or heat flux as described in the previous section.

#### 3.2.2.4 Sensors

FFD allows users to add "virtual sensor" to get the information such as temperature, flow velocity and contaminant concentration at any user defined location (e.g. the room center) or space (e.g. the room or the occupied zone). The value can be instantaneous or time-averaged. By default, the standard FFD sensor output provides the time and volume averaged room air temperature to Modelica as

$$T_{room}(t_n) = \frac{1}{V_{room}\Delta t}\int_{t_{n-1}}^{t_n}\int_{V_{room}} T(V,t)\, dV\, dt. \tag{10}$$

Users can also add their own sensors by adding codes to the FFD program. For instance, FFD can send to Modelica the average temperature of the occupied zone defined as

$$T_{occ}(t_n) = \frac{1}{V_{occ}\Delta t}\int_{t_{n-1}}^{t_n}\int_{V_{occ}} T(V,t) dV\, dt, \tag{11}$$

where $\boldsymbol{V_{occ}}$ is the volume of a user-defined occupied zone.

## *3.3 Implementation*

### *3.3.1 Implementation in the Modelica Buildings Library*

The Modelica *Buildings* library version 1.6 couples the well-mixed indoor environment and the HVAC system through the connection of fluid ports and/or heat

ports of the room model and HVAC component models. The room model named *Rooms.MixedAir* simulates the indoor environment with the assumption of completely mixed air. This model can have any number of constructions and surfaces that participate in the heat exchange through convection, conduction, infrared radiation and solar radiation. The model *Rooms.MixedAir* and its window model have been validated (Nouidui, Phalak, et al. 2012; Nouidui, Wetter, et al. 2012). Based on the existing *Rooms.MixedAir* model, we introduced the new *Rooms.CFD* model to compute the room air using coupled simulation with CFD/FFD. The term "CFD" is used in the related Modelica model names because most of the implementation in the Modelica models can be also used for coupled simulation with other CFD programs.

As shown in Figure 3-2, the model icons of the *Rooms.MixedAir* and *Rooms.CFD* models are similar. This model similarity allows users to easily switch the two room models for different modeling purposes. For instance, *Rooms.MixedAir* can be used during a preliminary design to reduce the computing time. Subsequently, during a detailed design, one can replace the *Rooms.MixedAir* model by *Rooms.CFD* to increase accuracy. It is worth to note that there are two differences in the model icons between *Rooms.MixedAir* and *Rooms.CFD*. One is that *Rooms.CFD* does not have the input for the shading control signal because a movable shade would require the CFD/FFD program to change the surface area of the boundaries for the shaded and unshaded window during the simulation which is not implemented in FFD. Thus, modifications will be needed to use the current *Rooms.CFD* model to simulate the

room airflow with shading control. The other is that *Rooms.CFD* has extra outputs *yCFD* for output sensor data as discussed in section 2.2.4.

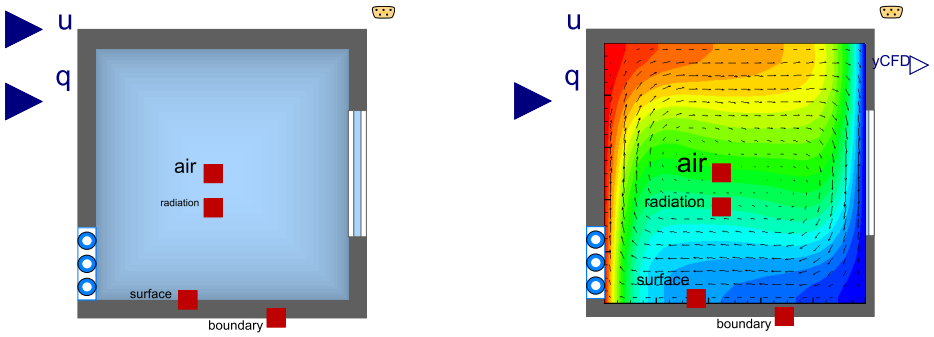
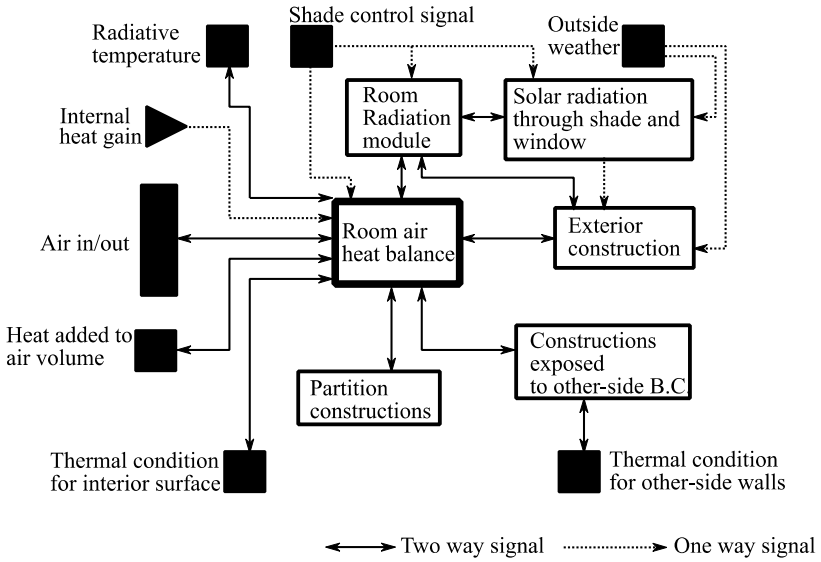
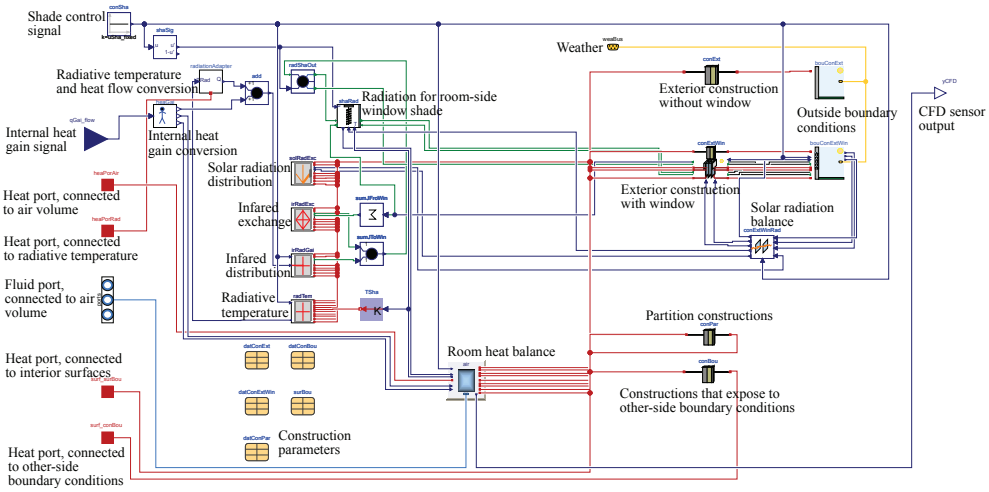



*Figure 3-2 Icons of the two room models (a)* Rooms.MixedAir *and (b)* Rooms.CFD.

Figure 3-3 shows the schematic and Modelica implementation of the model Rooms.*CFD* that is extended from *Rooms.BaseClasses.RoomHeatMassBalance*. The *RoomHeatMassBalance* model is largely based on the room model described by Wetter et al. (2011) However, it was redesigned to be a based model for both *Rooms.MixedAir* and *Rooms.CFD*. The major modification is to model the radiative heat balance in the *RoomHeatMassBalance* while computing the convective heat balance using the mixed air model or the CFD/FFD model, as they use different approaches to calculate the convective heat flow rate.




(a) Schematic




(b) Diagram of the Modelica model

*Figure 3-3 Schematic and diagram of the Modelica model* Rooms.CFD

The key component of the *Rooms.CFD* model is the model

*Rooms.BaseClasses.CFDAirHeatMassBalance* that calculates the heat and mass

balance of the air using CFD/FFD. It provides an interface between the causal modeling of CFD/FFD and the acausal modeling of Modelica. As shown in Figure 3-4, the co-simulation data exchange is managed by a block called *cfd*. To generate inputs and process outputs from the block *cfd*, there is one block named *fluInt* at the bottom center that interfaces the fluid ports. There are also nine blocks on the right that are the interfaces to the heat ports.

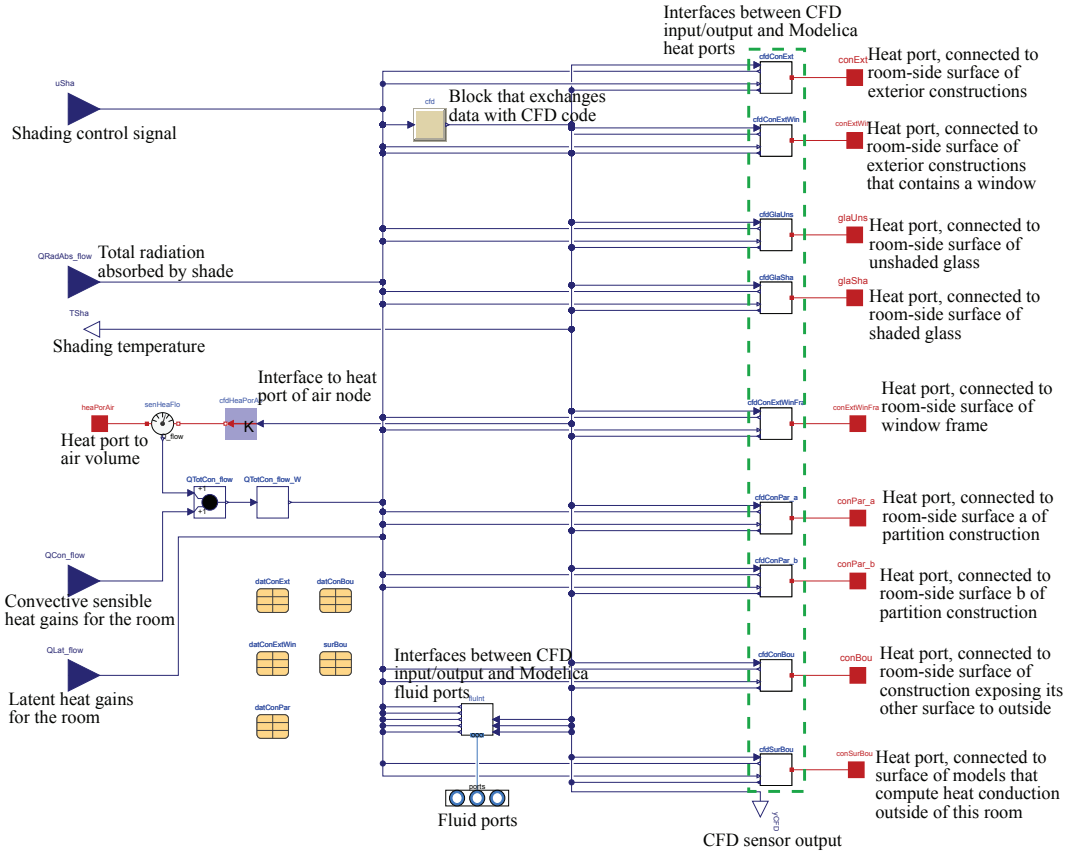



*Figure 3-4 Diagram of Modelica model* Rooms.BaseClasses.CFDAirHeatMassBalance.

### 3.3.2 Implementation in FFD

We also revised the FFD code in order to perform the coupled simulation with Modelica. The key change is to revise the boundary conditions at FFD so that they can change according to the direction of the air flow rate sent by Modelica. As mentioned before, fluid ports in Modelica allow the flow to change direction any time during the simulation. Since the Modelica model defines the inlet and outlet boundary conditions for FFD, it is possible that an inlet will become an outlet or vice versa during the simulation. This is realized by implementing a dynamic flow boundary definition in FFD. Immediately after each data synchronization, the FFD program will reset the inlet and outlet boundary conditions according to the signs of the mass flow rates as received from Modelica. The new boundary conditions will then be used for the FFD simulation until the next data synchronization.

To conduct the coupled simulation, Modelica calls C functions that initiate the FFD simulation, synchronize data during the simulation and terminate the FFD simulation at the end of the coupled simulation. The FFD program is compiled to a dynamically linked library (.dll on Windows or .so on Linux). This library will be loaded by the compiled Modelica code to access the C functions.

## 3.4 Case Study

Our previous paper (Zuo et al. 2014) introduced some preliminary work and validated the implementation by simulating simple airflow in an empty room without HVAC equipment and feedback control. This study further improved the implementation and evaluated the performance of the coupled simulation by using

more realistic flow conditions and adding an HVAC system and its control. We first quantitatively validated the coupled simulation by modeling ventilation in a space with high air exchange rate and heat load (such as in an aircraft cabin). Then we studied a feedback control for space heating with an idealized HVAC input. After that, we reduced the air exchange rate and heat load and replaced the idealized HVAC inputs with a constant air volume system to mimic the feedback control of space cooling in an office. To compare the difference in performance, we also simulated the same cases using the standalone Modelica simulation with *Rooms.MixedAir* model.

### 3.4.1 Non-Isothermal Flow with Stratified Distribution

This case simulates the ventilation for space (2.44 m × 2.44 m × 2.44 m) with a heated rectangular box (1.22 m × 1.22 m × 1.22 m) inside and its center is located at X = 1.22m, Y = 1.22 m, and Z = 0.61m (Figure 3-5). The heated box is to mimic a heat source, like occupants. The experiment (Wang and Chen 2009) was designed to study the airflow inside an aircraft cabin with a high internal heat load (about 700 W) and a high air flow exchange rate (ACH = 28.3). The inlet is located on the west wall with a height of 0.03 m and the outlet on the east wall with a height of 0.08 m. The velocity and temperature of the inlet flow are 0.455 m/s and 22.2 $^{o}$C, respectively. The temperature is 25.8 $^{o}$C on the ceiling, 26.9 $^{o}$C on the floor and 27.4 $^{o}$C on other walls. The temperature on the surface of the box is 36.7 $^{o}$C. The flow structure is complex because the internal obstacle and the airflow are under the strong interaction of inertia force and buoyance force. The detailed description and experimental data are available in (Wang and Chen 2009).

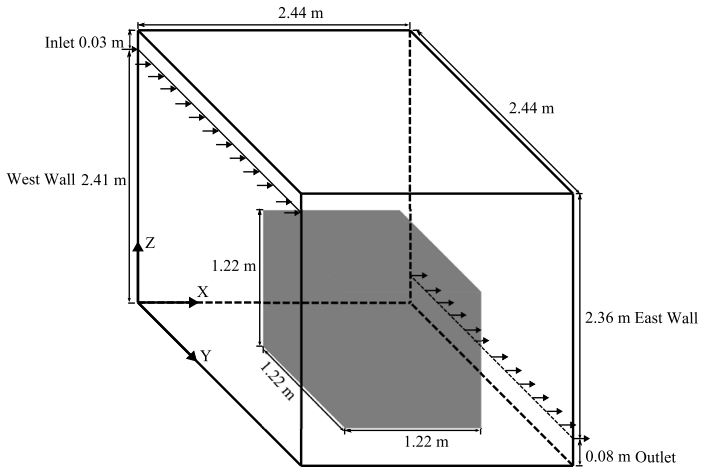



*Figure 3-5 Schematic of the non-isothermal flow with stratified distribution in an empty room with a box.*

Figure 3-6 shows the diagram of the Modelica models. In Modelica, we define the wall temperatures and the mass flow rate and temperature of the air into the room. The radiative, convective and latent heat gains were set to zero. For the FFD simulation, a non-uniform 20 × 20 × 20 mesh was used and the time step size was 0.1 s. The initial temperature of the room is set to be 22.2 $^{o}$C. The data between the two programs was synchronized every 4 s. The same settings of the FFD and data synchronization were applied to all the three cases in the paper.

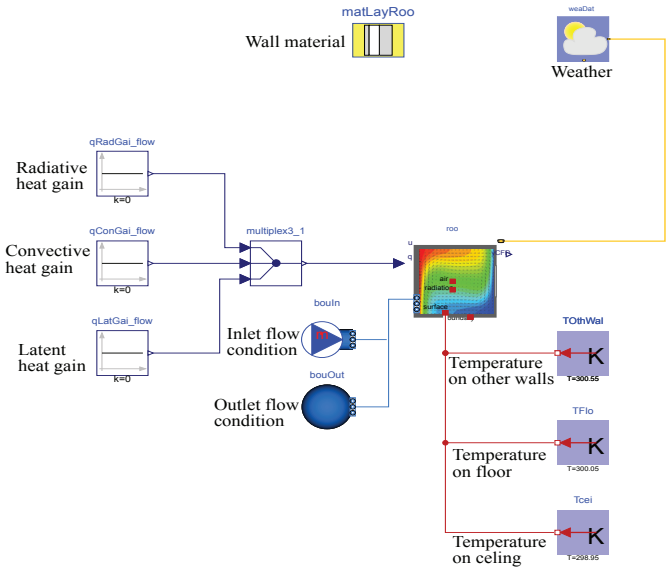



*Figure 3-6 Diagram of Modelica models for the non-isothermal flow with stratified distribution case.*

The detailed experimental data from the floor to the ceiling at ten locations were available (Figure 3-7). Here we showed the comparison at four locations that were at the front, top, back and side of the box. Figure 3-8 compares the velocity profiles normalized by a maximum velocity of $U_{\max}$ = 1.5 m/s. In general, the FFD prediction agrees with the experimental data. The relatively large discrepancy in prediction at point 5 is due to the complex flow structure that was also discovered in a previous study (Jin et al. 2012a). It is noteworthy that the studied flow is unstable flow with high turbulence intensity. Even state-of-the-art CFD models could not precisely capture all the flow details (Wang and Chen 2009). Since the FFD is a simplified CFD model, it is not expected to have higher accuracy than state-of-the-art CFD models.

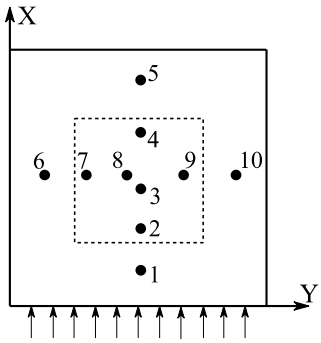
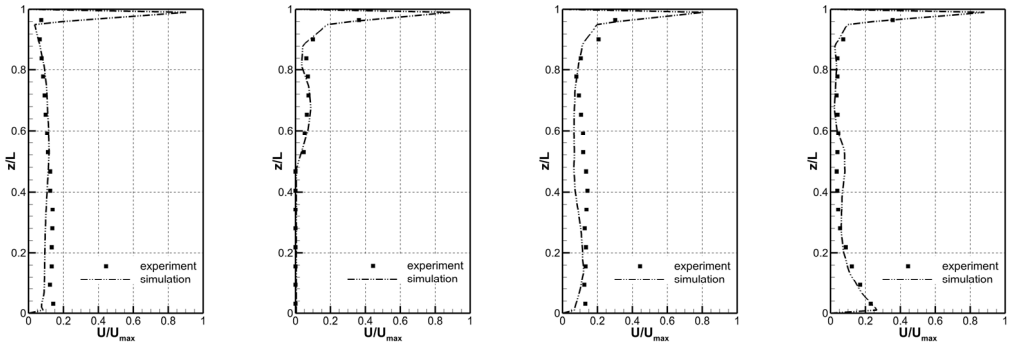



*Figure 3-7 the distribution of ten locations with experimental data available.*




*Point 1* *Point 3* *Point 5* *Point 6*

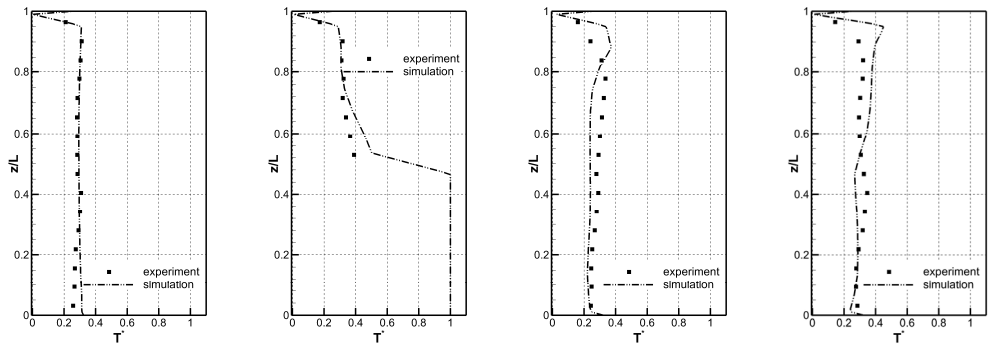
*Figure 3-8 Comparison of normalized velocity profiles calculated by the Modelica-FFD coupled simulation with the experiment data by (Wang and Chen 2009).*

Figure 3-9 compares the temperature profiles calculated by the coupled simulation and the experimental data. The temperature was normalized as

$$T^* = \frac{T - T_{min}}{T_{max} - T_{min}}, \tag{12}$$

where $T_{min}$ =22.2 ºC and $T_{max}$ = 36.7 ºC. FFD made a good prediction for point 1 which is in the front of the box. Due to the impact of box, FFD did not calculate the

temperature precisely for the locations behind (point 5) and around the obstacle (point 6). However, it still captured the stratification of the temperature along the line from the floor to the ceiling.




*Point 1* *Point 3* *Point 5* *Point 6*

*Figure 3-9 Comparison of normalized temperature profiles calculated by the Modelica-FFD coupled simulation with the experimental data by (Wang and Chen 2009).*

Figure 3-10 shows the side view of velocity vectors and temperature contours on the cross-section at Y = 1.22 m computed by FFD. The cold air was injected from the upper-left corner and a circulation was formed between the box and the east wall after the inlet air hit the east wall and then constrained by the box. The thermal plume rose up due to the impact of both buoyance force and air circulation. We put five virtual temperature sensors (s1 to s5) at different locations. The temperatures were then extracted from the FFD simulation and sent to Modelica during the coupled simulation.

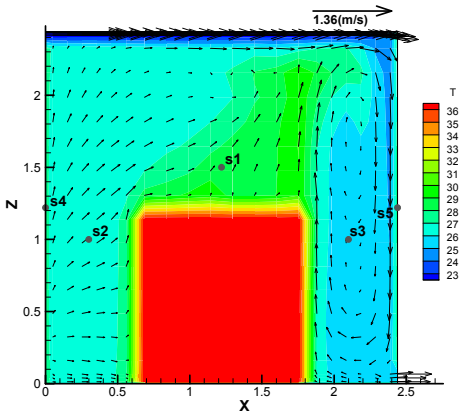
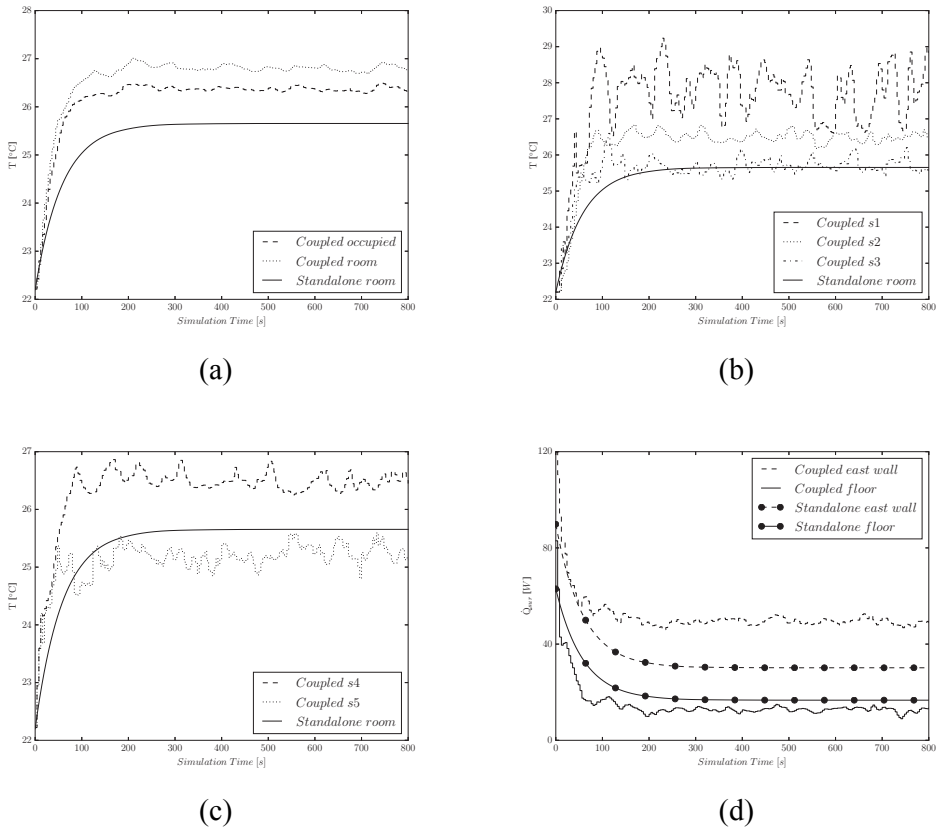



*Figure 3-10 Velocity vectors and temperature contour on a cross-section at Y = 1.22 m for the non-isothermal flow with stratified distribution case.*




(a) (b)

(c) (d)

*Figure 3-11 Comparison of the coupled FFD-Modelica simulation and the standalone Modelica simulation for the non-isothermal flow with stratified distribution case*

Figure 3-11 shows time series from the coupled FFD-Modelica simulation using the *Rooms.CFD* model and the standalone Modelica simulation using the *Rooms.MixedAir* model. Figure 3-11(a) compares the room temperatures predicted by both simulations and the temperature of the occupied zone ($Z \leq 1.22$ m) by the coupled FFD-Modelica simulation. Ignoring the temperature stratification in the standalone simulation led to the predicted room temperature approximately 1.5 °C lower than the one by the coupled FFD-Modelica simulation. In addition, the FFD-Modelica simulation computed an occupied zone temperature that was lower than the averaged room temperature. Due to the thermal plume generated by the heated box, the temperature at s1 (1.22 m, 1.22 m, 1.5 m), which was above the heated box, was the highest among the temperatures obtained from all five locations Figure 3-11b). Because the box blocked the access of cool air for locations s2 (0 m, 1.22 m, 1.22 m) and s4 (0.3 m, 1.22 m, 1.0 m), they also had higher temperatures than the room temperature computed by the standalone simulation using *Rooms.MixedAir* model. Locating on the path of supplied cool air, s5 (2.44 m, 1.22 m, 1.22 m) had the lowest temperature among the 5 locations (Figure 3-11c). Due to the dynamic characteristics of the flow, all temperatures showed in the coupled FFD-Modelica simulation were oscillating during the entire simulation although they were fully developed and their time-averaged value was almost constant. This reflects the actual flow condition in the reality. On the other side, the temperature predicted by the standalone simulation reached steady state with a constant value over time after about 300 s.

Figure 3-11(d) compares the heat flows from the wall to the air. In the coupled FFD-Modelica simulation the heat flow rates were calculated by FFD and passed to the Modelica model *Rooms.CFD*. Both the coupled and standalone simulations showed the declining heat flow rates over time because the room air was heated from a low initial temperature of 22.2 °C. When the room airflow fully developed, the heat flow rates became almost constant. The coupled FFD-Modelica simulation showed a higher heat flow rate between the east wall and air than the standalone simulation because the coupled simulation captured a layer of cool air near the east wall which the standalone simulation was not capable of predicting. Similarly, the coupled simulation also calculated a smaller temperature difference between the floor and air, which then led to a smaller heat flow rate compared to the standalone simulation.

The agreement in prediction between the coupled FFD-Modelica simulation and experimental data shows that the coupled simulation is able to predict the three-dimensional distribution of indoor airflow and the difference in the heat exchange with different parts of the building envelope. The next step is to evaluate the coupled simulation using HVAC system with feedback control.

### *3.4.2 Feedback Control for Space Heating*

In order to illustrate the capability of the coupled FFD-Modelica simulation for ventilation control, we simulated a case with air temperature control for a space under ventilation and heating in an idealized way. Note that his case is not intended to simulate a typical building during the heating season. The study was based on the

previous case with an additional PI control for adding heat uniformly to space. The control object was to maintain the room temperature at 30 °C. The implementation of the Modelica model is shown in Figure 3-12.

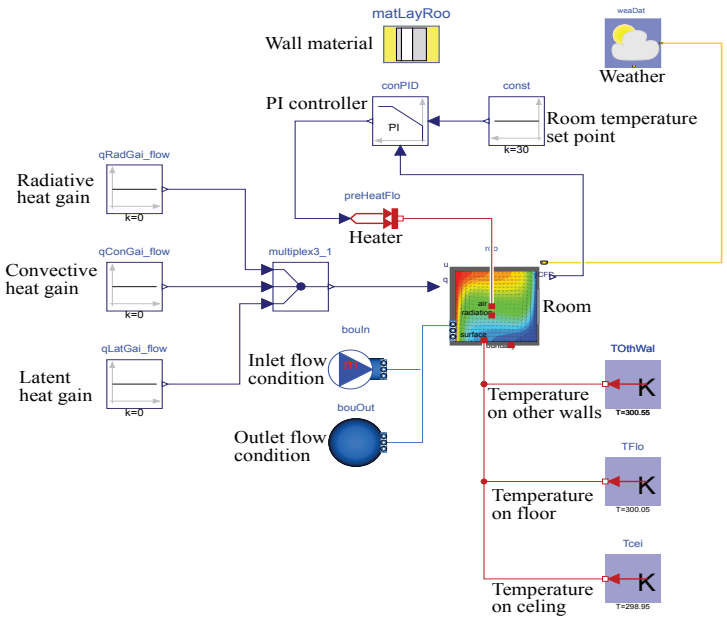



*Figure 3-12 Diagram of Modelica models for space heating.*

Figure 3-13 shows a non-uniform temperature distribution in the room computed by FFD. The temperature of the room air at the lower part (occupied zone) was lower than the upper part. The hottest spots occurred above the heated box. There was also a cold air layer near the ceiling and east wall formed by the cold inlet air.

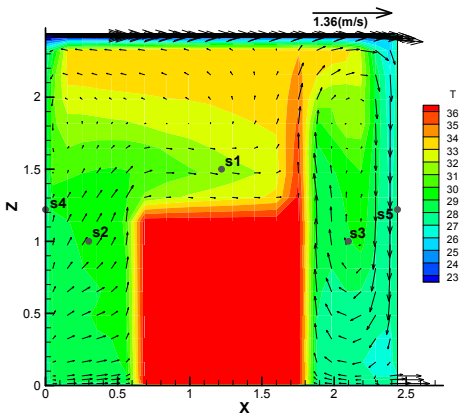
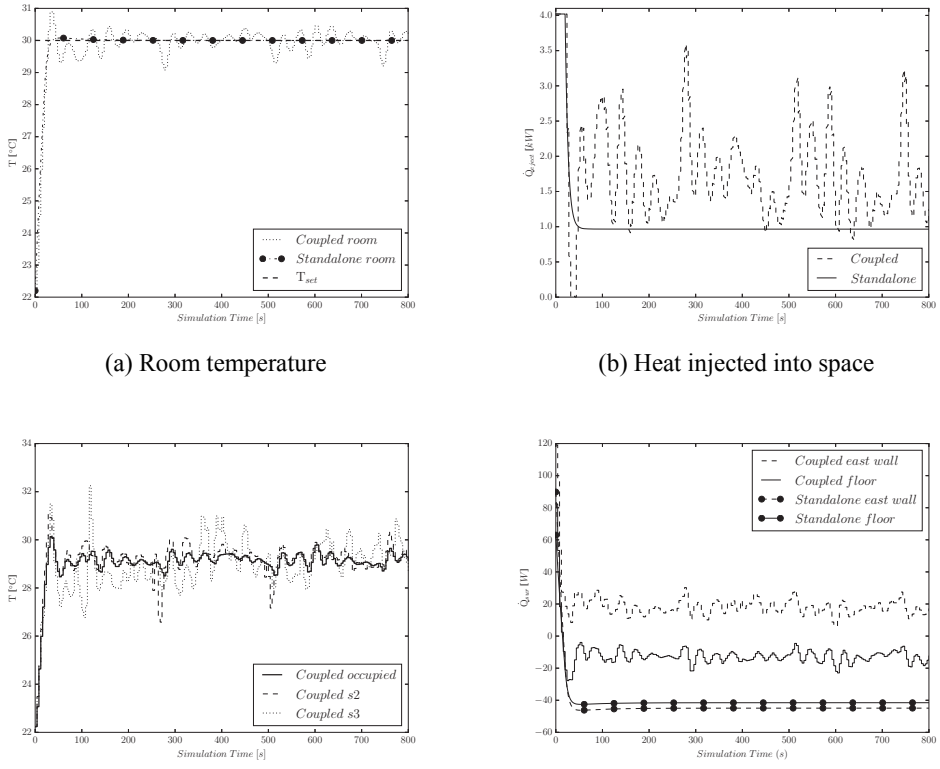



*Figure 3-13 Velocity vectors and temperature contour on a cross-section at Y = 1.22m for space heating).*

As shown in Figure 3-14(a), the room temperature was maintained at 30 ºC in the standalone simulation using *Rooms.MixedAir* model. The coupled FFD-Modelica simulation calculated a transient flow pattern and the controlled temperature was oscillating within ±1 ºC around the set point during the entire simulation. As a response, it was constantly adjusting the injected heat flow in order to maintain the room temperature (Figure 3-14b). There are three possible causes for the oscillations in temperature and heat injections: First, the airflow is dynamic in nature as discussed in the previous case. Second, synchronizing the sensor data from FFD and the control action from Modelica for every 6 s introduces a delay that may cause instability. Third, the controller is unstable for this process. The coupled simulation predicted temperatures of the occupied zone, s2 and s3 locations (Figure 3-14c) slightly lower than the average room temperature of 30 ºC, which realistically represented the temperature stratification in the space.

The standalone simulation predicted a uniform temperature of 30 ºC that was higher than the temperatures of the east wall (27.4 ºC) and the floor (26.9 ºC). Consequently, it predicted that the heat flow direction was from the air to the wall, as shown in Figure 3-14(d). In comparison, the coupled FFD-Modelica simulation computed a layer of cold air along the east wall due to the cold inlet air. Therefore, it predicted that the direction of heat flow through the east wall is from the wall to the air, which is opposite to the prediction of the standalone simulation.




(a) Room temperature

(b) Heat injected into space

(c) Temperature at other locations

(d) Heat flow between wall and air

*Figure 3-14 Comparison of the coupled FFD-Modelica simulation and the standalone Modelica simulation for the space heating.*

### *3.4.3 Feedback Control of Space Cooling*

The previous case of space heating control used an idealized HVAC system. In this case, we connected a constant air volume air conditioning system to the room for space cooling. In order to make the flow condition close to an office room, the supply air flow rate was reduced to 5.6 ACH (0.02 $m^3/s$). The surface temperature of the internal box was decreased to 27.4 $^oC$ to reduce the internal heat load. As shown in Figure 3-15, the warm outdoor air was first cooled by the exhaust air through a heat recovery device. Then it was further cooled by chilled water in a cooling coil. The control objective was to maintain the room temperature at a set point of 26 $^oC$. The chilled water supply was controlled by an on-off controller which behaves accordingly based on the difference between set point and measured room temperature. A deadband of 1 K was applied to avoid short cycling. The simulation was performed for a physical process of 800 s.

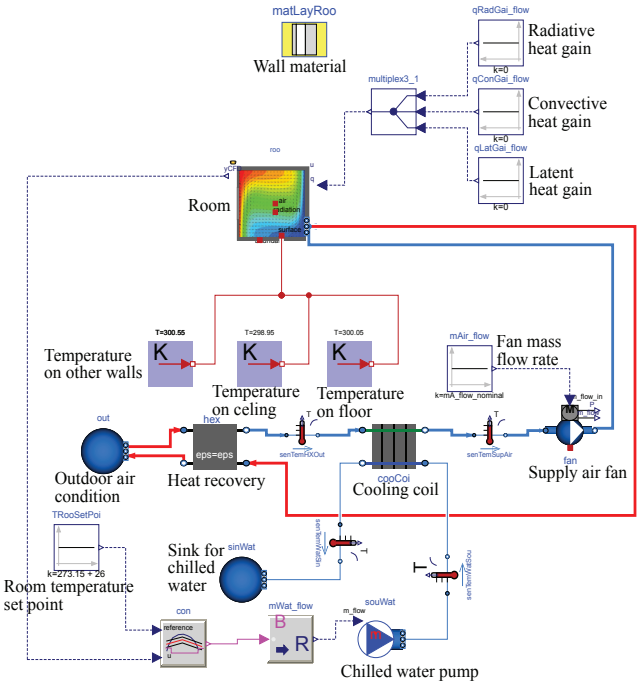



*Figure 3-15 Diagram of Modelica models for space cooling.*

Figure 3-16 shows the results from FFD at 800 s. Cold air was injected into the room through the inlet and sunk into the lower left corner of the room. The upper right corner, since heavily impacted by the thermal plume generated by the heated box and far less affected by the cold airflow air, remained at a higher temperature.

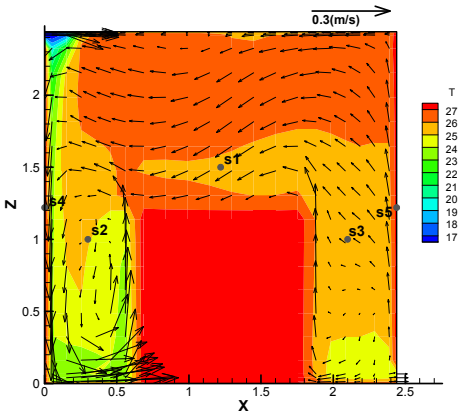
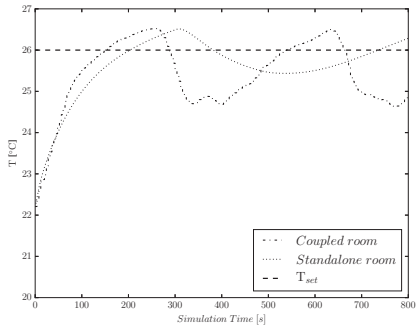



*Figure 3-16 Velocity vectors and temperature contour on a cross-section at Y = 1.22m for space cooling.*

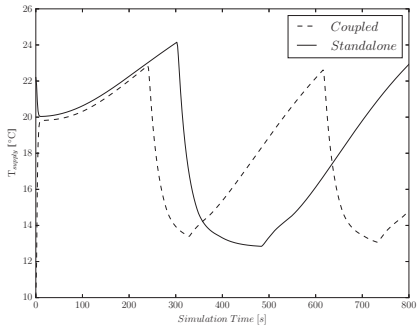
Figure 3-17(a) shows the room temperatures in both simulations. The room temperature calculated by the standalone simulation using the *Rooms.MixedAir* model was oscillating within ±0.5 ºC around the set point. The coupled FFD-Modelica simulation predicted a room temperature varying about -1.5 ºC to +0.5 ºC around the set point. The coupled simulation predicted a quicker rise of the room air temperature than the standalone simulation. As a response, the FFD-Modelica simulation predicted an earlier turning-on for the chilled water valve to cool the supply air than that in the standalone simulation (Figure 3-17 b and c).

By getting the spatial air temperature distribution for the near wall region, the coupled simulation can better estimate the heat flow rates between the air and wall than the stand-alone simulation (Figure 3-17a). Based on a uniform room air temperature, the standalone simulation predicted the difference between heat flow rates on the east wall and floor is only due to the difference in the surface

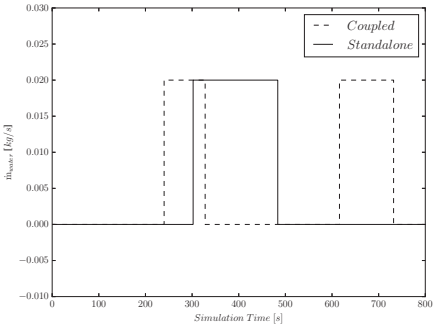
temperatures. On the other side, the FFD-Modelica coupled simulation considered that the air temperature near the east wall was actually higher than the one near the floor, so the difference in heat flow rates through these two surfaces was smaller than the one predicted by the standalone simulation.



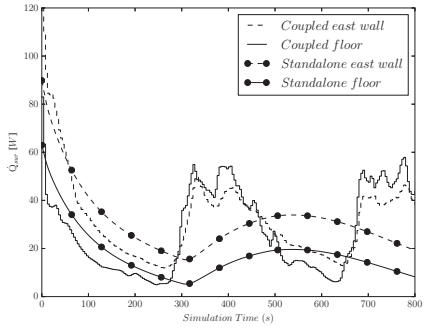

(a) Room temperature




(b) Supply air temperature




(c) Chilled water flow rate




(d) Heat flow rate between wall and air

*Figure 3-17 Temperature at sensor location of the room, supply air temperature and chilled water flow rate for the space cooling.*

### 3.4.4 Computing Time

The Modelica models were simulated using a Modelica simulation environment called Dymola 2015 FD01 (www.dynasim.se). The Radau solver with a

tolerance of $10^{-6}$ in Dymola was applied in all simulations. The simulation was performed using a workstation with an Intel Xeon Processor E5-1603 with a four-core CPU at 2.8 GHz. The results showed that the coupled FFD-Modelica simulation using the *Rooms.CFD* model provided detailed flow information with a significant cost on computing time (about 430 s for each of the three cases) compared to about 0.2 s required by the standalone simulation using the *Rooms.MixedAir* model. However, the coupled FFD-Modelica simulation was still faster than the real time since the simulated physical process was 800 s.

## 3.5 Conclusion

In this chapter, we implemented and validated a coupled simulation between the FFD and the Modelica Buildings library for the dynamic simulation of building ventilation system with stratified air distributions. The coupled simulation was implemented as a new *Rooms.CFD* model in the Modelica *Buildings* library. For the non-isothermal flow with stratified distribution case, the coupled simulation could calculate the air velocity and temperature distribution close to the experimental data. For the cases of space cooling and space heating with feedback control, the coupled FFD-Modelica simulation could also realistically represent the dynamic non-uniform air distributions in the room that were not captured by the standalone Modelica simulation using the *Rooms.MixedAir* model. This difference resulted in different heat exchange rates with building envelopes and different control actions between the coupled and standalone simulations. For the three cases, the coupled FFD-Modelica simulation using the *Rooms.CFD* model was faster than the real time, but

significantly slower than the standalone Modelica simulation using the *Rooms.MixedAir* model. Thus, it is recommended to use the standalone Modelica simulation in the preliminary design/evaluation and the FFD-Modelica simulation for the specific room in the detailed design/evaluation if spatial distributions of the air properties are of interest.

The oscillations in indoor air temperature can be attributed to the following factors: frequency of data synchronization in the simulation, which can be a reflection of sensor sampling frequency in a real system; the placement of temperature sensors, and the tuning of control parameters. With the capability of dynamic simulation of stratified air distribution and HVAC system, our tool provides a more realistic environment for control engineers and researchers to study the control of stratified ventilation systems.

Given that design optimization required hundreds of iteration of simulation to seek the optimal, the computation speed is still not fast enough (based on previous results, it is projected 12 hours' time cost for a design day simulation). Since airflow simulation takes up most of the time, performing the FFD simulation in parallel using GPU or multicore CPUs could dramatically reduce the simulation time. We cover the parallelization of FFD simulation in chapter Chapter 5.

# Chapter 4 Coupling Fast Fluid Dynamics and Multizone Airflow Models in Modelica *Buildings* Library to Simulate Dynamics of HVAC Systems

## *4.1 Background*

Chapter 3 introduced a coupled simulation that allowed for the dynamic interaction between the stratified airflow, HVAC, control and building envelope models in a single zone (Zuo et al. 2014; Zuo et al. 2016). In that work, FFD model (Zuo and Chen 2009c) was used to simulate the stratified airflow distribution within a room. However, in the work of Zuo et al. (Zuo et al. 2014; Zuo et al. 2016), FFD used a constant pressure that was independent of the pressure of the Modelica model. Instead of calculating the mass flow rate according to the pressure difference between two rooms, the FFD used in the *Room.CFD* model must rely on Modelica to provide the inlet mass flow rate. As a result, the model can only be used as a standalone tool, and cannot be linked to the multizone models in the library to simulate the airflow distribution within a large building. Similarly, it is not feasible to connect *Room.CFD* and *Room.MixedAir* for the study of the thermal environment, envelopes and system control in a building that typically contains many rooms. These limitations make the coupled simulation model introduced in chapter Chapter 3 nearly impossible to be applied to a building with multi zones.

To provide a timely prediction of airflow in large buildings consisting of multiple zones, or rooms, researchers proposed to apply only the CFD model to a few zones with stratified airflow while also utilizing the multizone models for the rest of

the building. Gao (2002) developed methodologies to couple CONTAM (Walton 1994) with a CFD program called MIT-CFD. Three different data synchronization strategies (virtual coupling, quasi-dynamics coupling, and dynamic coupling) were proposed. Case studies had been carried out to use the developed model to study common indoor airflow types such as forced convection, natural convection and contaminant transportation in a building (Yuan 2003; Tan and Glicksman 2005). Wang and Chen (2007b) further validated the dynamic coupling strategy by using experiments in which stratified airflows were involved including the non-uniform distribution of momentum, temperature, and contaminant (Wang and Chen 2007b; Srebric et al. 2008).

While significant, previous work focused strictly on the airflow movement and left out modeling of HVAC systems and controls. It is critical to add the HVAC modeling of the coupled multizone and CFD models for airflow to seek a holistic solution for building ventilation control.

To integrate the coupled airflow simulation of CFD-multizone with the models for HVAC and control, this paper reports our research in the coupling of three models based on the Modelica *Buildings* library and CFD. Although the coupling scheme can employ any CFD program, again this paper uses FFD as an example since it is about 50 times faster than CFD (Zuo and Chen 2009c). For ease of writing, in the rest of the paper, we use FFD only, whereas the reader should bear in mind that FFD can be replaced by conventional CFD programs. We first introduce the

implementation of data synchronization between FFD and multizone models. Then we verify the implementation using two case studies involving an isothermal flow and a non-isothermal flow and compare the results with experimental data. Afterward, we study another three cases with more complicated/realistic building configurations. Finally, we discuss further research needs on the coupled simulation model.

## 4.2 Research Methodology

### 4.2.1 Mathematical Description of FFD and Multizone Models

#### 4.2.1.1 FFD Model

FFD solves the Navier-Stokes equations:

$$\frac{\partial \boldsymbol{U}_i}{\partial t} = -\boldsymbol{U}_j \frac{\partial \boldsymbol{U}_i}{\partial \boldsymbol{x}_j} + \nu \frac{\partial^2 \boldsymbol{U}_i}{\partial \boldsymbol{x}_j \partial \boldsymbol{x}_j} - \frac{1}{\rho}\frac{\partial P}{\partial \boldsymbol{x}_i} + \boldsymbol{F}_i \tag{13}$$

where $\boldsymbol{U}_i$ and $\boldsymbol{U}_j$ are the velocity component in $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ directions, respectively, $\nu$ is the kinematic viscosity, $\rho$ is the fluid density, $P$ is the pressure, $t$ is the time, and $\boldsymbol{F}_i$ is the source term, such as the buoyancy force. FFD splits the Navier-Stokes equation into the following three equations:

$$\frac{\partial \boldsymbol{U}_i}{\partial t} = -\boldsymbol{U}_j \frac{\partial \boldsymbol{U}_i}{\partial \boldsymbol{x}_j} \tag{14}$$

$$\frac{\partial \boldsymbol{U}_i}{\partial t} = \nu \frac{\partial^2 \boldsymbol{U}_i}{\partial \boldsymbol{x}_j^2} + \boldsymbol{F}_i \tag{15}$$

$$\frac{\partial \boldsymbol{U}_i}{\partial t} = -\frac{1}{\rho}\frac{\partial P}{\partial \boldsymbol{x}_i} \tag{16}$$

FFD first solves the advection equation (2) using a semi-Lagrangian method (Courant et al. 1952). It then solves the diffusion equation (3) with an implicit scheme. Finally, it solves the pressure equation (4) together with the continuity equation

$$\frac{\partial \boldsymbol{U}_i}{\partial \boldsymbol{x}_i} = 0 \tag{17}$$

using a projection-correction method (Chorin 1967). FFD applies a similar algorithm to solve the conservation equations of energy and species. In the literature, both the sequential FFD programs (Zuo and Chen 2009c; Jin et al. 2012b) and parallel FFD programs (Zuo and Chen 2010a; Yang 2013; Tian et al. 2017) are available.

#### 4.2.1.2 Multizone Model

A typical multizone model uses a power law relation to express flow as a function through an orifice. In the power law equation, the mass flow rate $\dot{m}_{ij}$ represents the flow from zone $i$ to zone $j$ (Dols and Walton 2002):

$$\dot{m}_{ij} = C_d A\sqrt{2\rho}\Delta P^m \tag{18}$$

where $C_d$ is the discharge coefficient normally ranging between 0.6 to 0.75; $A$ is the area size of the opening; $\rho$ is the density of the air; $m$ is constant, which is 0.5 for large openings. $\Delta P$ is the pressure difference which is the aggregate sum of the total

pressure difference $|P_i - P_j|$, and pressure difference as a result of wind $\Delta P_w$, and pressure difference due to density and elevation difference $\Delta P_t$ (Wang and Chen 2007b).

Since Modelica is an equation-based, object-oriented modeling language (Fritzson 1998), the sign of $\dot{m}_{ij}$ can be automatically determined based on the pressure in two zones. Thus, we can write the mass conservation for zone $i$ as:

$$\frac{dm_i}{dt} = \sum_{j=1}^{n} \dot{m}_{ij} + F_i \tag{19}$$

where $t$ is time; $n$ is number of surrounding neighbors to zone $j$; $m_i$ is the air mass in the zone $i$; $F_i$ is the air mass source in the zone $i$. Once the boundary conditions (e.g. ambient pressure) are applied, the pressure at each zone and mass flow rate between neighboring zones can be determined uniquely.

### *4.2.2 Coupling Strategies between FFD and Multizone Model*

As defined by Zhai et al. (2002a), there are mainly three categories in coupling two building simulation programs: static coupling, dynamic coupling, and quasi-dynamic coupling. Considering that we would need to couple the HVAC and control simulation later, we chose the quasi-dynamic coupling strategy (Figure 4-1) which provides a suitable balance between computing time and capturing system dynamics. A quasi-dynamic coupling requires that FFD and multizone models exchange

information consisting of $\boldsymbol{x_1}(t_n)$ *and* $\boldsymbol{x_2}(t_n)$ mutually at data synchronization time points $t_n$. After data synchronization is completed, the two programs will run separately for one synchronization time step $\Delta t_{syn}$ until the next data synchronization point. FFD uses a constant time step size $\Delta t_{ffd}$ while the multizone model, implemented in Modelica, adopts a variable time step size that is automatically determined in the implicit integration algorithm to solve the equations efficiently (Wetter and Haugstetter 2006).

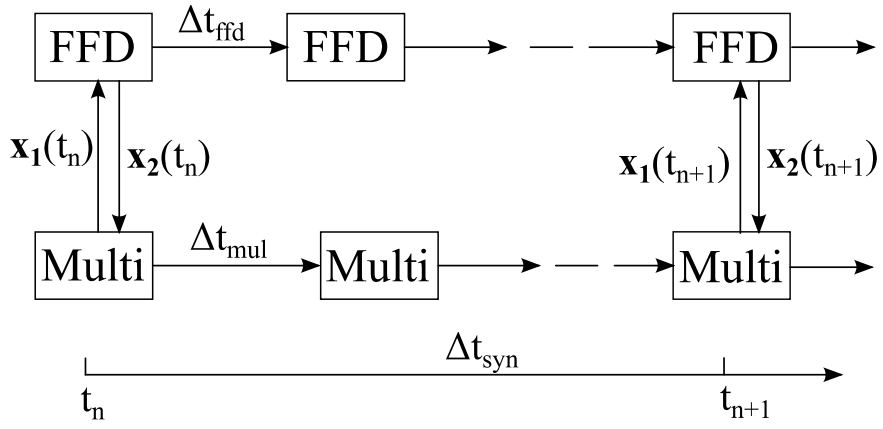



*Figure 4-1 Quasi-dynamic coupling between FFD and multizone model*

The exchanged data $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ between the two programs can vary according to boundary conditions applied to FFD. If the inlet flow rate for FFD is known prior to multizone model calculation, the velocity boundary condition is applied to FFD.

Otherwise, the pressure boundary condition is applied. In the following sections, we will introduce our implementations for different boundary conditions.

#### 4.2.2.1 Mass Flow Rate Boundary Condition

In some cases, such as rooms with a mechanical ventilation system, the inlet mass flow rate $\dot{m}_{inlet}$ at the inlet can be specified. As shown in Figure 4-2, as one inlet is facing towards an outlet, a strong momentum effect takes place at the outlet in *Zone 1*. Thus *Zone 1* is a candidate for simulation using FFD to express the unevenly distributed mass flows at the outlets (Wang and Chen 2007b). *Zone 2* and *Zone 3* are then simulated by the multizone models. Regarding the exchange data, the flow rate $(\dot{m}_{in\_1}, \dot{m}_{in\_2}, \dots, \dot{m}_{in\_i})$ and temperature $(\varphi_{in\_1}, \varphi_{in\_2}, \dots, \varphi_{in\_i})$ at inlets are fed to FFD and FFD gives the flow rate $(\dot{m}_{out\_1}, \dot{m}_{out\_2}, \dots, \dot{m}_{out_j})$ and temperature $(\varphi_{out\_1}, \varphi_{out\_2}, \dots, \varphi_{out\_j})$ at outlets back to the multizone model, as follows:

$$\boldsymbol{x_1} = \{\dot{m}_{in\_1}, \dot{m}_{in\_2}, \dots, \dot{m}_{in\_i}, \varphi_{in\_1}, \varphi_{in\_2}, \dots, \varphi_{in\_i}\} \tag{20}$$

$$\boldsymbol{x_2} = \{\dot{m}_{out\_1}, \dot{m}_{out\_2}, \dots, \dot{m}_{out_j}, \varphi_{out\_1}, \varphi_{out\_2}, \dots, \varphi_{out\_j}\} \tag{21}$$

where $i$ and $j$ are the numbers of the inlet and outlet in FFD, respectively; $\dot{m}$ is the mass flow rate; $\varphi$ is the scalar variable such as temperature and trace substance concentration.

Receiving $\boldsymbol{x_1}$, at the inlet, FFD calculates the velocity at inlets based on the mass flow rates $(\varphi_{in\_1}, \varphi_{in\_2}, \dots, \varphi_{in\_i})$ and applies the velocity and temperature as

Dirichlet boundary conditions for inlets. Finally, for the outlet and wall, FFD applies Neumann boundary conditions. Once completing the simulation for one data synchronization time step, FFD feeds the mass flow rate at the outlets ($\dot{m}_{out\_1}, \dot{m}_{out\_2}, \dots, \dot{m}_{out_j}$) to calculate the pressure ($P_2, P_3$) at the other zones and the mass flow rate ($\dot{m}_{20}, \dot{m}_{30}$) at the other openings based on Equation (18).

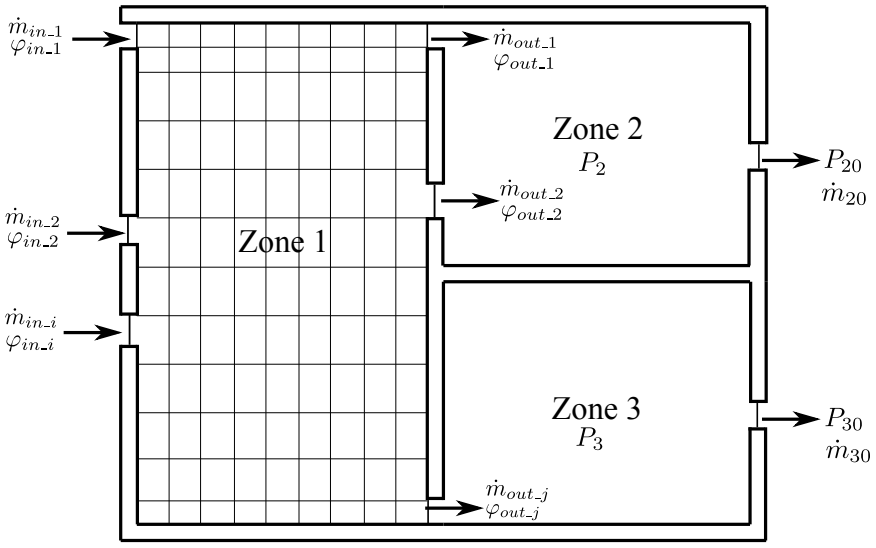



*Figure 4-2 Sketch of the case where a velocity boundary condition is applied*

### 4.2.2.2 Total Pressure Boundary Condition

On other occasions such as wind-driven natural ventilation, the mass flow rate at the inlet cannot be directly obtained. Instead, the total pressure can be derived from measuring at the upstream point. In contrast, as shown in Figure 4-3, despite that the mass flow rate at the inlet of *Zone 1* may be known, the mass flow rate $\dot{m}_{12}$ at inlet of *Zone 3*, which is simulated using FFD, is unknown prior to the calculation of

multizone model. *Zone 1*, *Zone 2* and *Zone i* are simulated by the multizone model. As show in Equations (22) and (23), the multizone model gives the pressures $(P_1, P_2, \dots, P_i)$ and other scalar variables $(\varphi_{M\_1}, \varphi_{M\_2}, \dots, \varphi_{M\_i})$ at all openings to FFD. FFD will use the sign of the mass flow rates calculated by Modelica to determine if the openings are inlets or outlets. Then after that, FFD will only use the temperatures at the inlet for calculation, as for outlets it applies a Neumann (zero-gradient) boundary condition for temperature. Upon completing the calculation for one data synchronization time step, FFD transfers the mass flow rates $(\dot{m}_1, \dot{m}_2, \dots, \dot{m}_i)$ and scalar variables $(\varphi_{F\_1}, \varphi_{F\_2}, \dots, \varphi_{F\_i})$ at all elements of the multizone model. In the equations below, $i$ represents the number of openings in the CFD simulation.

$$\boldsymbol{x_1} = \{P_1, P_2, \dots, P_i, \varphi_{M\_1}, \varphi_{M\_2}, \dots, \varphi_{M\_i}\} \tag{22}$$

$$\boldsymbol{x_2} = \{\dot{m}_1, \dot{m}_2, \dots, \dot{m}_i, \varphi_{F\_1}, \varphi_{F\_2}, \dots, \varphi_{F\_i}\} \tag{23}$$

Wang and Chen (2005) showed that by exchanging total pressure and mass flow rate between multizone model and CFD, it is feasible to achieve converged results between coupled models.

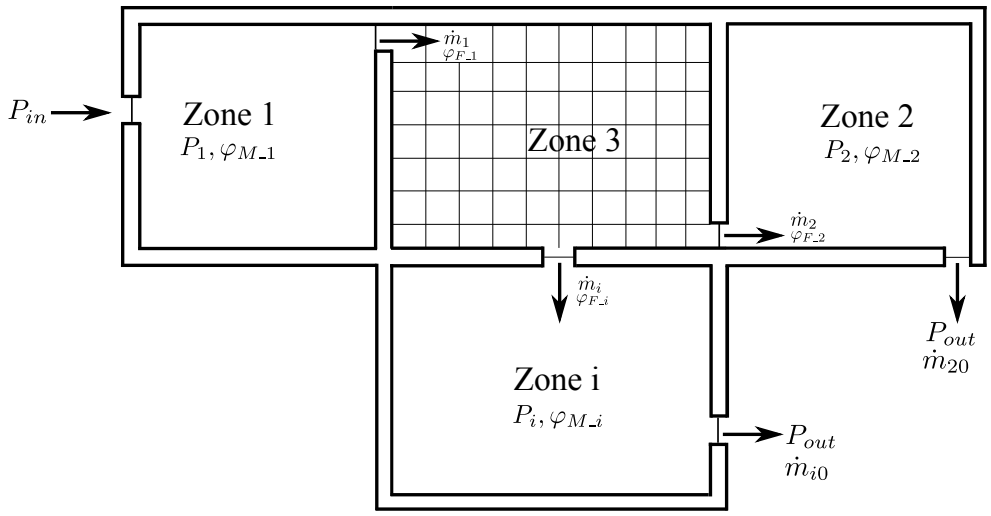



*Figure 4-3 Sketch of the case where a total pressure boundary condition is applied*

After receiving the $\boldsymbol{x_1}$ from the multizone model, FFD will calculate the mass flow rate and values of scalar variables at all openings. In lieu of velocity as a boundary condition for inlets, static pressures are used together with the temperatures at inlets for FFD to determine the mass flow rates $(\dot{m}_1, \dot{m}_2, \dots, \dot{m}_i)$ and temperature $(\varphi_{F\_1}, \varphi_{F\_2}, \dots, \varphi_{F\_i})$ at all openings. The details of how FFD applies pressure boundary conditions to openings will be introduced in section 3.2. Finally, FFD sends that information as $\boldsymbol{x_2}$ back to the multizone models. The multizone models will then determine the total pressure $(P_1, P_2, P_i)$ and temperature $(\varphi_{M\_1}, \varphi_{M\_2}, \dots, \varphi_{M\_i})$ at other zones and the mass flow rates $(\dot{m}_{20}, \dot{m}_{i0})$ at other openings in the systems.

## 4.3 Implementation of the coupled FFD and Multizone Models

This chapter introduces the implementation of coupling strategies in section 4.2. Previously, Zuo et al. (2016) presented a coupling framework between Modelica and CFD (using the FFD program as an example) to study the heat transfer through building envelopes, HVAC operation and control, and airflows in buildings. FFD was compiled as a dynamic linker and called by Modelica once the simulation was being executed. The coupling framework named as *Rooms.CFD* was later publicly released along with the Modelica *Buildings* library. The implementations in this paper are dependent on that work.

### 4.3.1 Implementation with Mass Flow Rate Boundary Condition for FFD

Figure 4-4 shows the schematic of the first coupling strategy which applies mass flow rate as a boundary condition to FFD. The icon named *CFD Zone* is an instance of *Rooms.CFD*, which interfaces between Modelica and FFD. On one hand, the model *CFD Zone* calculates the mass flow rates at all fluid ports, which are *Port 1* and *Ambient,* in this case, using a mixing volume assumption within the *Rooms. BaseClasses.CFDFluidInterface*. The information (flow rate, pressure, etc) will be given to *Rooms.BaseClassess.CFDExchange* to feed into FFD. Note that the outlet *Ambient* is connected to *CFD Zone* through a *Resistance*, as this is to guarantee that the equation sets governing the mixing volume are fully closed when the number of outlets exceeds two. Receiving the mass flow rates, FFD will first assign “inlet ” and

"outlet" tags to openings based on the sign of mass flow rate. Afterward, it can assign the corresponding boundary conditions. Similarly, the *Heat port* is connected to the *CFD Zone* to provide the thermal boundary conditions of the walls to FFD.

On the other hand, FFD sends the exchange data quantities back to Modelica through the *CFD Zone*. The mass flow rates and temperature at the outlet fed to a prescribed mass flow rate *Fluid mover*. The *Fluid mover* works as an ideal flow source that can provide any user-provided values of flow rate, temperature, composition and trace substance. The *Fluid mover* is further connected to *Port 2*, which is used as an interface to connect the outside fluid port.

Figure 4-5 illustrates the Modelica implementation of the first coupling strategy. Note that the mass flow rates at the outlet are connected to a prescribed mass flow rate *Fluid mover* through a first order delay, which is used to avoid model failure. Additionally, the sensor information is sent out through an output icon to facilitate the modeling of the control. The FFD implementation is the same as that in the coupling platform presented in the literature (Zuo et al. 2016).

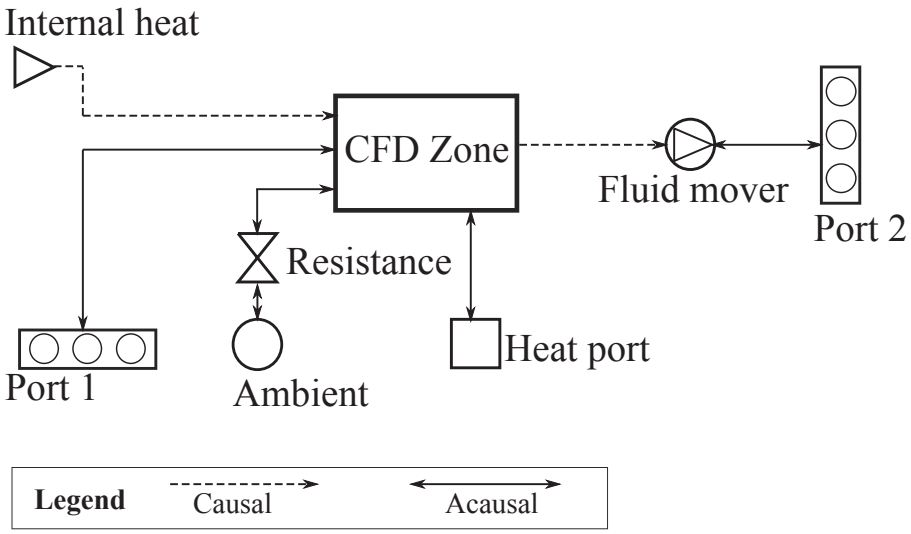



*Figure 4-4 Schematic of coupling strategy in Modelica when velocity boundary condition is applied to FFD*

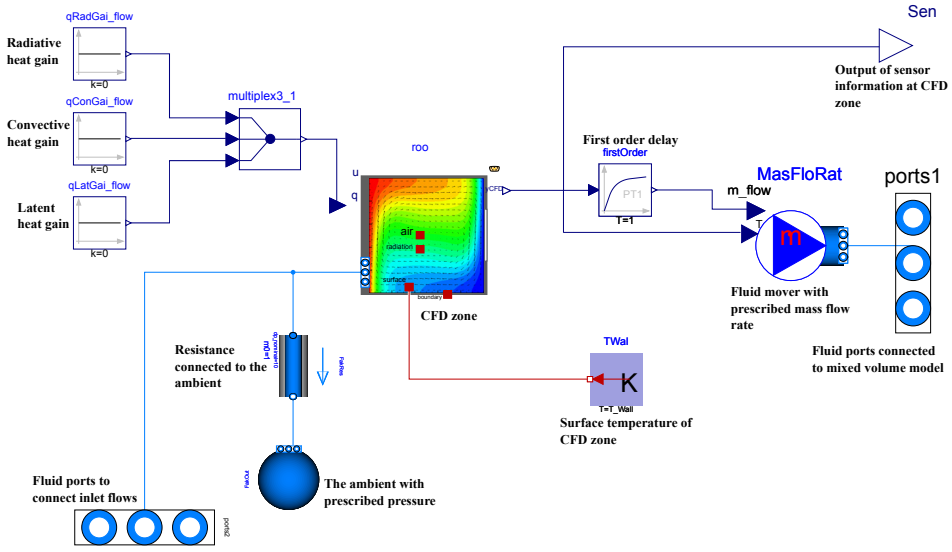



*Figure 4-5 Implementation of coupling strategy in Modelica when a velocity boundary condition is applied to FFD*

### 4.3.2 Implementation with Total Pressure Boundary Condition for FFD

Compared to the previous implementations, the total pressure at the fluid ports is sent to FFD from Modelica and the mass flow rate at all ports are sent from FFD to Modelica. Though largely based on the first implementation, as is illustrated in Figure 4-4, the schematic of the implementation shown in Figure 4-6 differs from the previous implementation in following aspects: first, an additional mixing volume called *Volume 1* is added to receive the pressure (P) and temperature (T) information at the "upstream" zone while *Volume 2*, which previously was *Ambient*, is now used to collect the pressure (P) information at the "downstream" zone. Second, as FFD gives the information of flow rates at all openings back to *CFD Zone*, another port called *Port 2* is created to be connected to the upstream openings (inlet). Similarly, *Port 2* is connected to a *Fluid mover*, which receives information of flow rate and temperature information from *CFD Zone*.

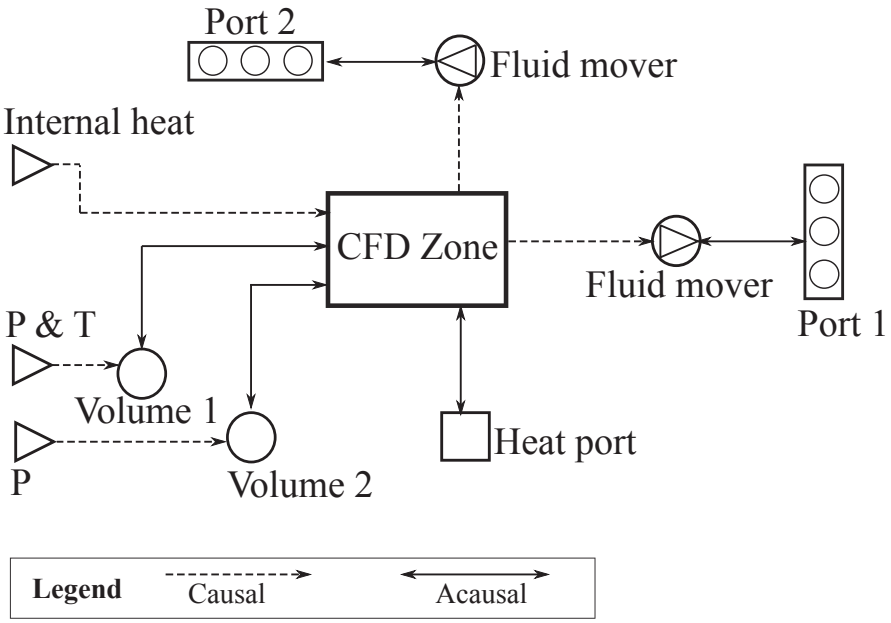



*Figure 4-6 Schematic of coupling strategy in Modelica when total pressure boundary condition is applied to FFD.*

Figure 4-7 shows the Modelica implementation of coupling strategy with total pressure being applied to FFD. Note that to avoid redundancy, vectors are being applied to many models such as mixing volumes, fluid movers, first order delays, etc. Figure 4-8 shows the implementation of total pressure boundary conditions in the FFD program when it is coupled with the multizone model implemented in Modelica. Receiving the pressure at inlets and outlets from Modelica ($P_{in}, P_{out}$), FFD applies the total pressure $P_{out}$ for outlets directly. For inlets, FFD assumes an initial velocity at the inlet at the beginning of the simulation, and calculates the static pressure $P_{static}$ as a boundary condition for the inlet based on the Bernoulli's principle. Afterward, the

FFD algorithm is executed by sequentially solving the advection, diffusion, and projection equation. Consequently, the velocity at the inlet will be updated after each iteration. If the data synchronization point is reached, FFD writes the information to Modelica and receives updated total pressures from Modelica.

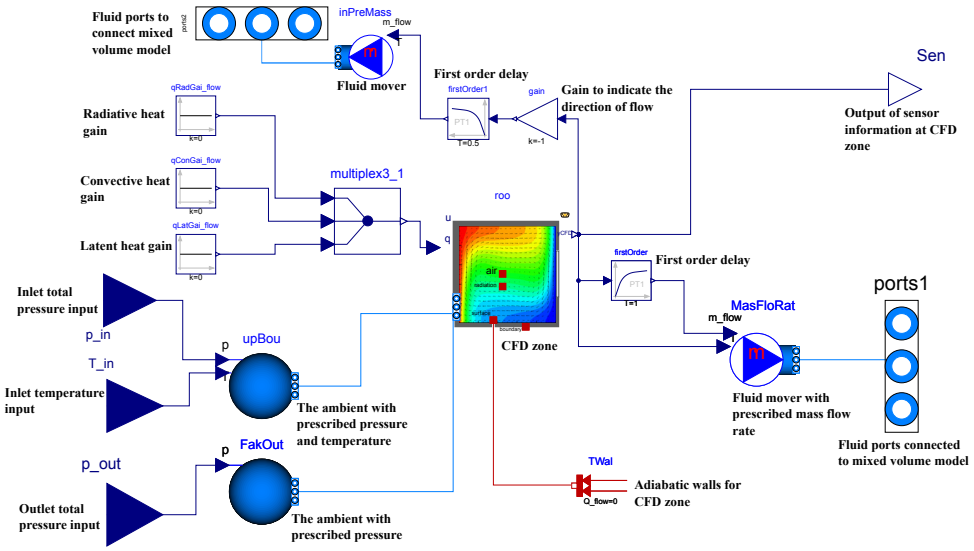



*Figure 4-7 Implementation of coupling strategy in Modelica when total pressure boundary condition is applied to FFD.*

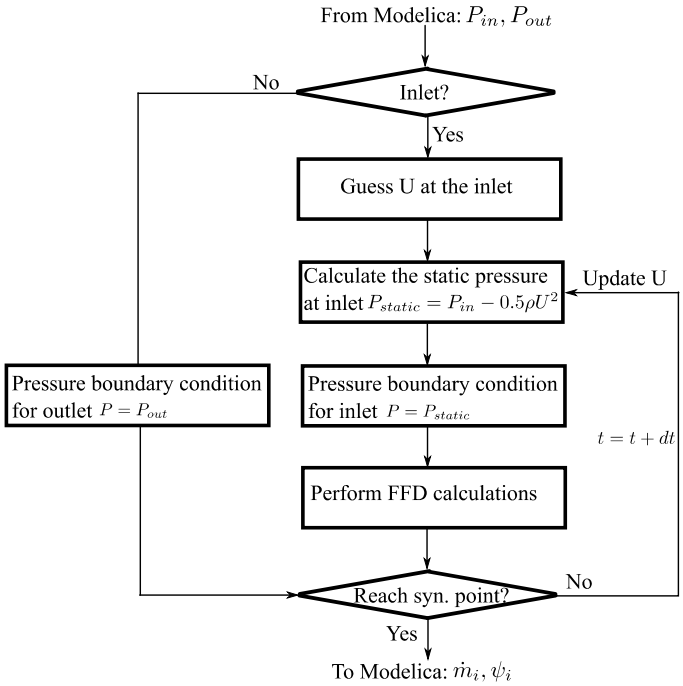



*Figure 4-8 Total pressure implementation in FFD*

## 4.4 Performance Assessment and Validation

The implementations in Modelica of coupling FFD and multizone models were further validated using two experiments by Wang and Chen (2007b). One case is an isothermal flow involving a non-uniform momentum distribution and the other one is a non-isothermal flow involving a non-uniform temperature distribution.

### 4.4.1 Validation 1: Isothermal Flow with Non-Uniform Momentum Distribution

The isothermal flow case is used to validate the first coupling implementation in section 4.3.1. Figure 4-9 shows the sketch of the test room involving the non-uniform momentum distribution since the inlet is directly facing *Opening 1* at *Zone 1*, which is simulated by FFD. The size of *Zone 1* is 4.93 m × 1.83 m × 2.44 m. The inlet is 0.3 m × 0.2 m, and the sizes of *Opening 1* and *Opening 2* are the same (0.4 m × 0.2 m). The flow rate at the inlet varies from 73, 113, 223, 296, and 456 CFM (or 0.033, 0.053, 0.105, 0.14, and 0.215 $m^3/s$) at different scenarios.

In the FFD simulation, a mesh size of 34 × 12 × 18 was used. The time step size was 0.1s for when the flow rates were 0.033 and 0.053 $m^3/s$ and 0.05s for others. To simulate the turbulence effect introduced by the high-velocity jet, we employed a zero-equation model proposed by Chen and Xu (1998). However, we adjusted the $\alpha$ coefficient to 0.00874 in Equation (24), in which $\rho$, $V$, $l$ are density, local mean velocity, and a length scale, respectively.

$$\mu_t = \alpha \rho V l. \tag{24}$$

This is because compared to the CFD, the splitting of Navier-Stokes equations in FFD introduces numerical viscosity (Zuo et al. 2012). Finally, we turned off the energy equation to simulate the case as a forced convection. For other detailed information about the case setup, one can refer to the literature (Wang and Chen 2007b).

The Modelica representation of the case is shown in Figure 4-10. *Zone 1* was simulated by FFD while other zones were studied by multizone model, simply represented by using a *MixingVolume* model from the *Buildings* library. The model named *Zone_1* in Figure 4-10 is the instance of the class introduced in section 2.3.1. The inlet airflow was provided by a prescribed mass flow rate fluid mover model that was connected to the fluid ports of *Zone_1*. The opening on the surfaces of the zones was modeled by the *Orifice*, which was implemented based on the Equation (18). The data synchronization time step was set as 5 s.

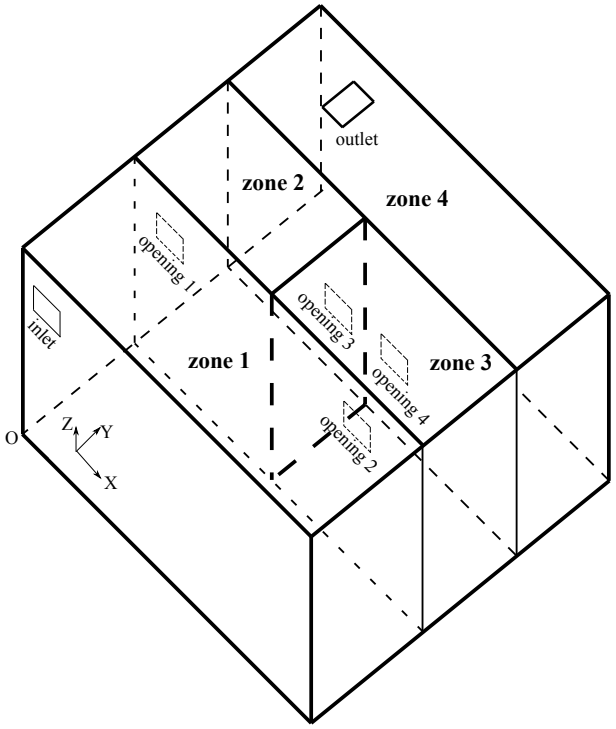



*Figure 4-9 Sketch of the isothermal case for validation*

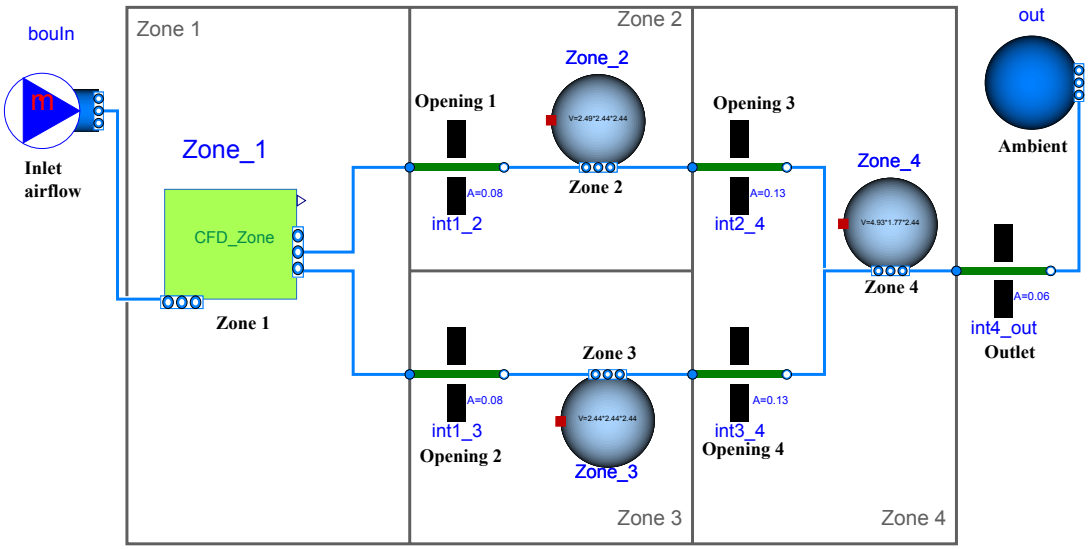



*Figure 4-10 Modelica model of the isothermal case for validation*

Then the model was compiled and ran in Dymola 2016 with Microsft Visual Studio 2013 as the compiler. The Radau 5$^{th}$ order scheme was used as the solver and the tolerance was set as $10^{-6}$. The simulation time was 100 s. Figure 4-11 shows the flow rate ratio of *Opening 1* and *Opening 2* in *Zone 1* under different inlet flow rates. The simulated results show a good agreement with the experiment when the inlet mass flow rate is generally larger. This is due to the fact that the numerical viscosity in FFD is associated with the grid resolution and it becomes relatively smaller compared to the turbulent viscosity when the velocity of inlet jet is high. Interestingly, the results from our coupling model are better than those from Wang and Chen (2007b), which coupled the CFD program with the CONTAM. One of the possible

reasons is that we tuned the coefficients of the zero-equation turbulence model for FFD using one of the five cases.

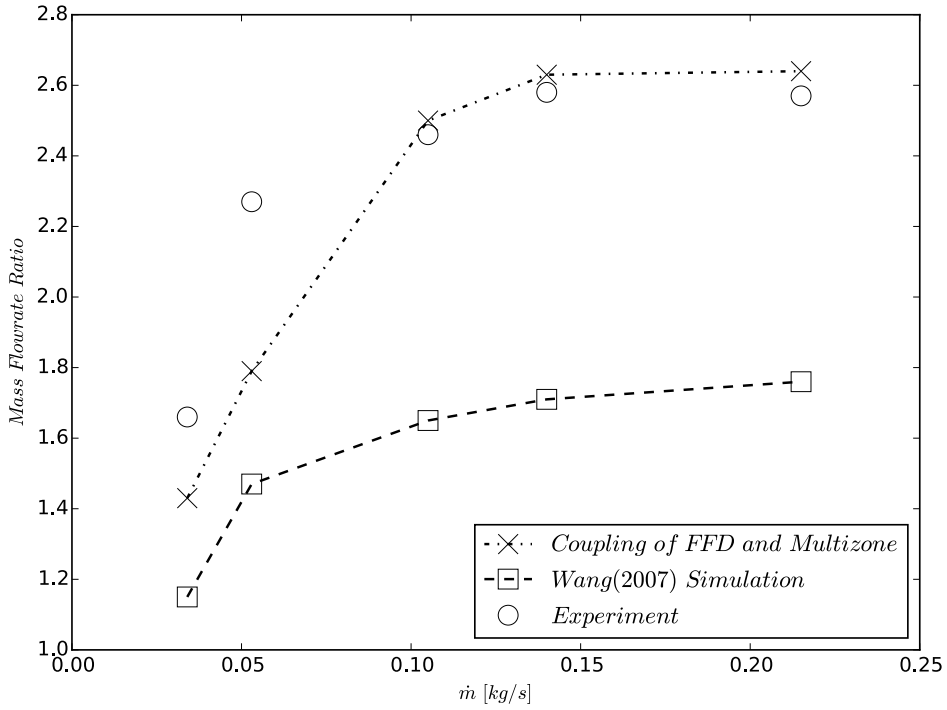



*Figure 4-11 Comparison of mass flow ratios predicted by coupling FFD and multizone, simulation of Wang and Chen (2007b), and experimental data for case 4.1*

Figure 4-12 shows the Modelica results of the mass flow rate at the openings and the pressure in the zones under a flow rate of 0.033 $m^3/s$. The flow rate at *Opening 1* and *Opening 2* were 0.020 and 0.013 kg/s at the time of 30 s, when the flow was estimated to be fully developed. Due to the mass conservation law, *Opening 1* had same airflow rate as *Opening 3* and so did *Opening 2* as *Opening 4*. The pressure at *Zone 2* is slightly higher than *Zone 3* due to the higher air flow rate. Note

that in the first 5 seconds of the simulation the mass flow rate and pressure are zero as zero initial values are applied to the air flow rates of *Opening 1* and *Opening 2*.

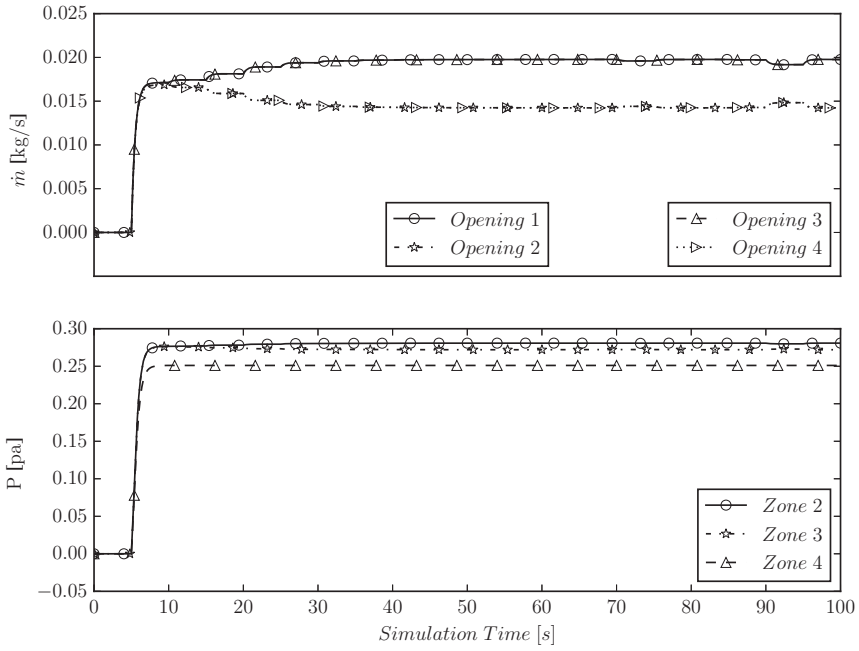



*Figure 4-12 Mass flow rates and room pressures calculated by Modelica models in case 4.1*

### 4.4.2 Validation 2: Non-Isothermal with Non-Uniform Temperature Distribution

The non-isothermal case is used to validate the second coupling implementation in section 4.3.2. Figure 4-13 shows the sketch of the test room that is modified based on the previous case. In the new case, *Zone 2* and *Zone 3* contain two identical blocks symmetrically located near the interior corner. The dimensions of the

blocks are 0.4 m × 0.25 m × 0.5 m. An additional inlet (*Inlet 2*) was also added to *Zone 1* and the size is 0.3 m × 0.2 m which is identical to the *Inlet 1.* Finally, *Zone 2* is 2.49 m × 2.44 m × 2.44 m. Since the block surface temperature at *Zone 2* is higher than the air a non-uniform temperature distribution was formed, thus *Zone 2* was simulated by FFD. Three scenarios were tested, in which the flow rate and temperature at the inlet of *Zone 1* and the block surface temperature in *Zone 2* were varied. The variations are shown in Table 4-1.

*Table 4-1 Boundary conditions for FFD for the non-isothermal case 4.2*

| **Scenario #** | **1** | **2** | **3** |
|---|---|---|---|
| **Flow Rate at *Inlet 1* ($m^3/s$)** | 0.0477 | 0.051 | 0.0514 |
| **Temperature at *Inlet 1* (°C)** | 18.5 | 18.9 | 18.5 |
| **Flow Rate at *Inlet 2* ($m^3/s$)** | 0.0543 | 0.0467 | 0.0533 |
| **Temperature at *Inlet 2* (°C)** | 18.5 | 18.7 | 18.3 |
| **Block Surface Temperature** | 30 | 35 | 46 |

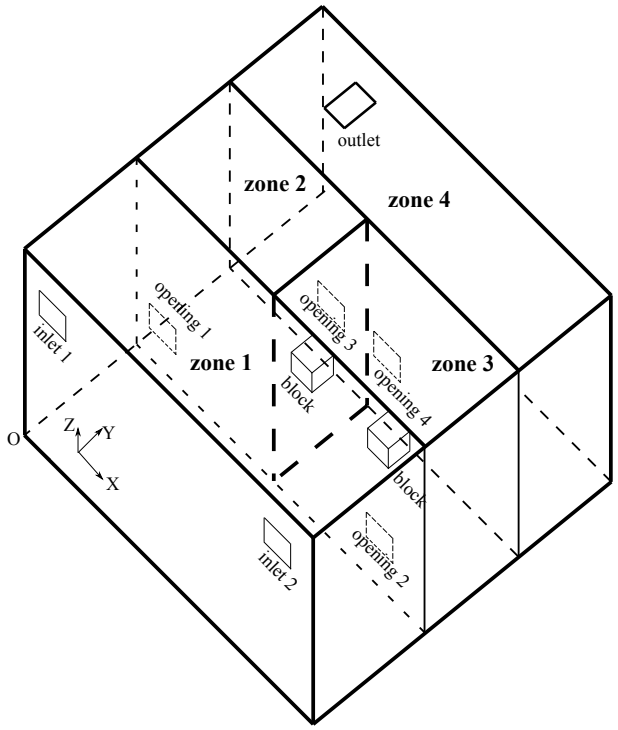



*Figure 4-13 Sketch of the non-isothermal case for validation*

In the FFD simulation, we used a mesh of 24 × 24 × 24. The time step size was 0.05 s. Again, the zero-equation model with the same coefficients as the previous case was employed. We used the Boussinesq assumption to account for the buoyant force caused by the temperature difference.

The Modelica implementation of the non-isothermal validation case is shown in Figure 4-14. The model named *Zone_2* is the instance of the model presented in Figure 4-7. *Zone 1*, *Zone 3*, and *Zone 4* were simulated by the *MixingVolume*. A pressure and a static temperature sensor were put at the upstream of *Zone 2* and also a pressure sensor was fixed at the downstream. The information from those sensors was

provided to FFD as the boundary conditions at the openings. For other parts of the model, they were like the previous case.

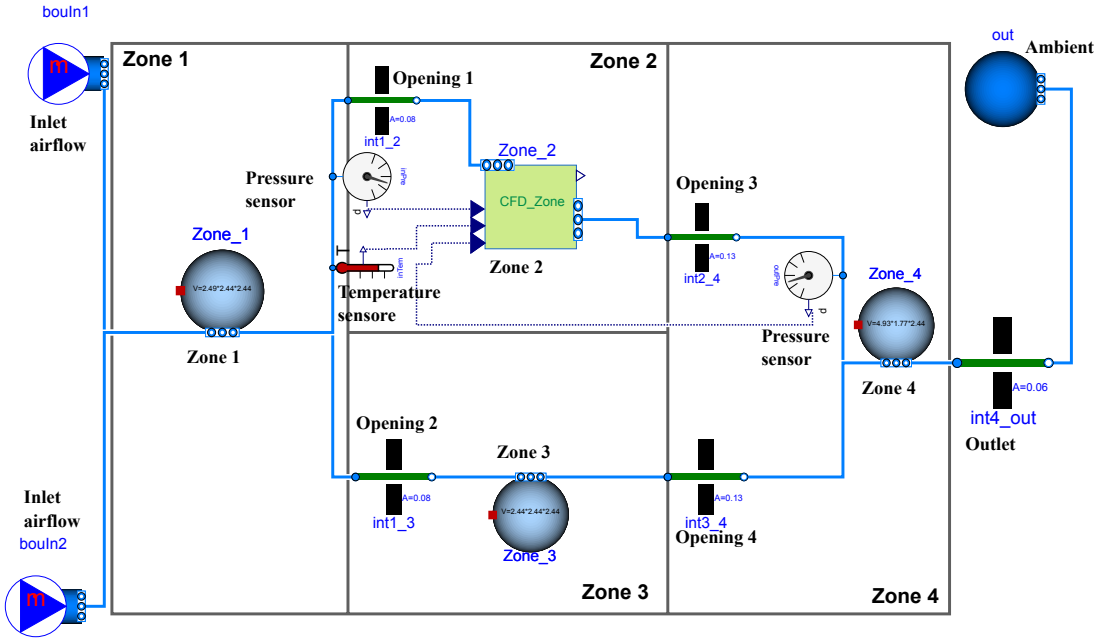



*Figure 4-14 Modelica model of the non-isothermal case for validation case 4.2*

The solver and compiler settings are identical to the previous case. The simulation time is 1000 s. Figure 4-15 shows the flow rate ratio at *Opening 3* and *Opening 4* in three different scenarios. Thanks to a larger pressure near the outlet induced by the stack effect, there was more air flowing out of the outlet in *Zone 2* than *Zone 3*. As the block surface temperature increased from 30 to 46 °C, the magnitude of the stack effect increased, and consequently the flow rate ratio of *Opening 1* compared to *Opening 2* increased accordingly from 1.23 to 1.46. The simulated results were in good accordance with the experimental data, except for the

first scenario. Our results agreed with those from Wang and Chen (2007b), and they attributed the large discrepancy in the first scenario to a measurement error.

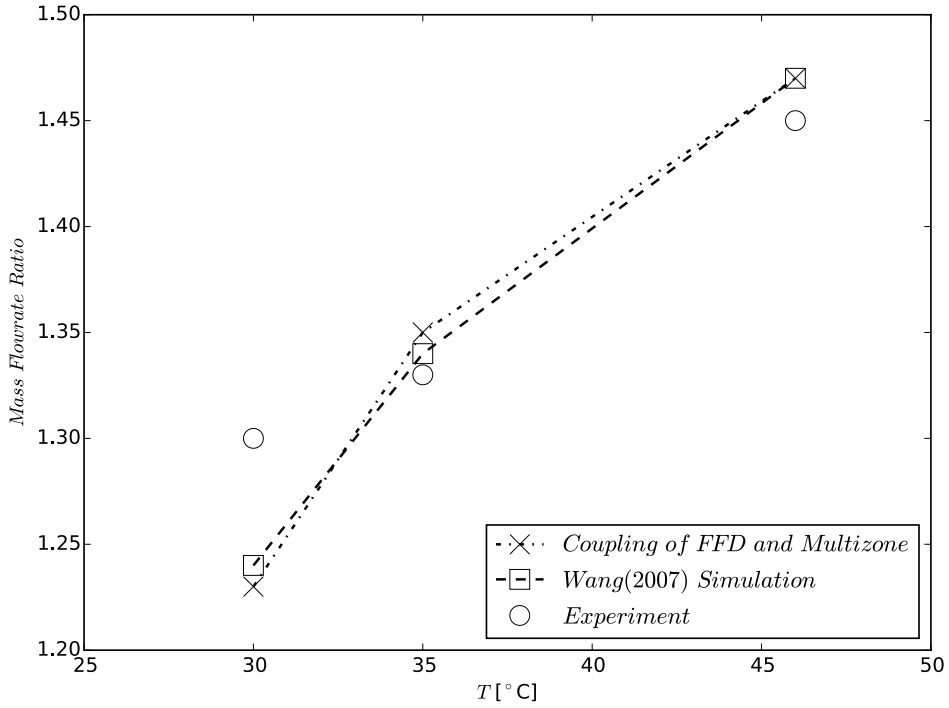



*Figure 4-15 Comparison of mass flow ratios predicted by coupling FFD and multizone, simulation of Wang and Chen (2007b), and experimental data for case 4.2*

Physically, for the whole space, the total pressure at zones and the flow rates at openings were fully coupled. As the initial values were assigned to the flow rates at openings, the pressure at all zones was determined. With the updated pressure information, the flow rate can be obtained. Usually, several iterations were needed to find the correct solution of pressure and flow rates as the calculation tended to be stabilized. Figure 4-15 shows the Modelica results of the flow rate at openings and pressure at zones for the scenario #1. The initial value of zero was given to *Opening 1*

and *Opening 3*, and initial flow rates at *Opening 2* and *Opening 4* were large (0.104 kg/s). They were identical due to the mass conservation. This further formed a large pressure difference of 4.0 Pa between *Zone 1* and *Zone 4*. With a large pressure difference fed to FFD, the mass flow rates at *Opening 1* and *Opening 3* were determined to be 0.092 kg/s. After FFD sent these mass flow rates to the multizone model, the flow rate at *Opening 2* and *Opening 4* decreased from the initial value of 0.104 to 0.012 kg/s and the pressure difference between *Zone 1* and *Zone 3* consequently decreased from 4.0 to 3.6 Pa. With several iterations, the magnitude of the fluctuations of mass flow rate and pressure decreased gradually and the simulation converged to the solution after about 300s. Eventually, the mass flow rates at *Opening 1* and *Opening 2* were 0.05633 and 0.04567 kg/s, which led to a flow rate ratio of 1.23.

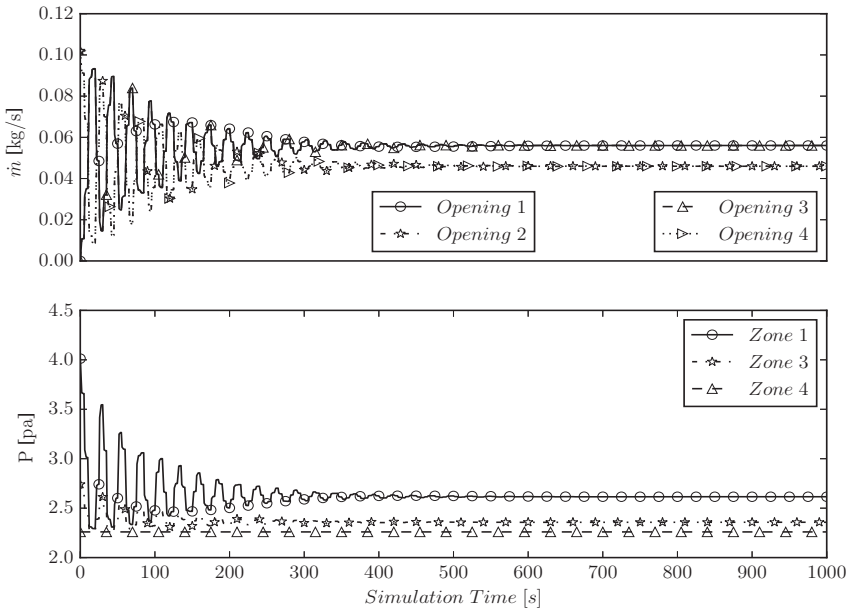



*Figure 4-16 Mass flow rates and room pressures calculated by Modelica models in case 4.2*

## 4.5 Case Studies

In the section three case studies were performed to show that the coupled FFD-multizone model can be used to study the dynamic response of an HVAC system. We started with adding a VAV terminal box to previously validated cases. Afterward, we further increased the complexity of the cases by adding a VAV system to the flow.

### 4.5.1 Flow with Non-Uniform Momentum Coupled to A VAV Terminal Box

Based on the airflow network model in Figure 4-10 in section 4.4.1, this case study added a VAV terminal box to substitute the prescribed fluid movers connected to *Zone 1*, as shown in Figure 4-17. The control objective of the VAV terminal box is to sustain 25 ℃ temperature for occupant zone of *Zone 1*, which is the lower half part (Z≤1.22 m). To increase the efficiency of temperature control, we increased the length of the inlet (in the X direction) by 0.53 m. The surface temperatures for floor and other walls in *Zone 1* are 25 ℃ and 27 ℃, respectively. The initial temperatures of all the zones is 30 ℃.

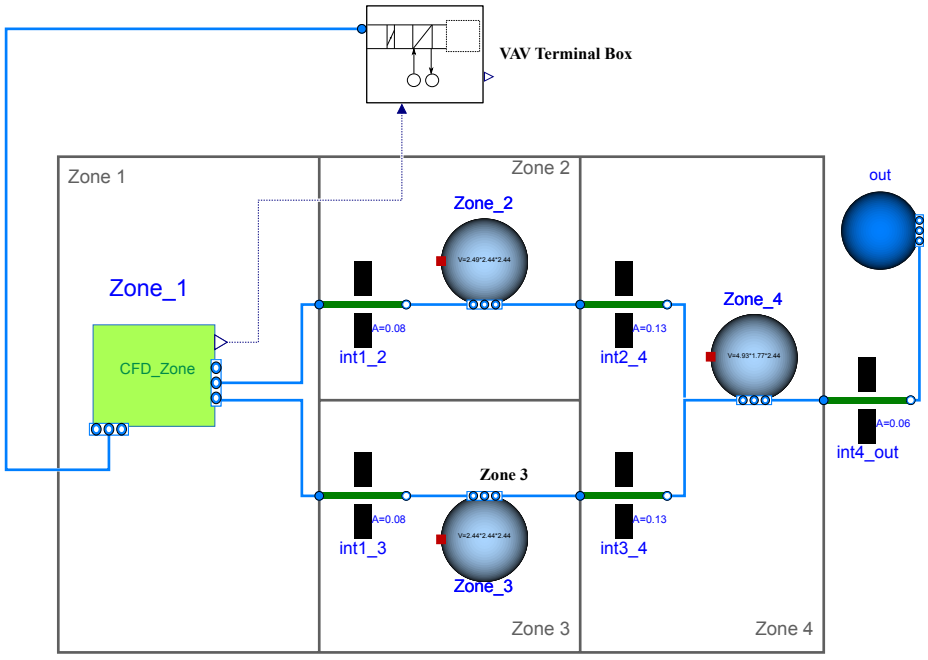



*Figure 4-17 Top level diagram of Modelica models for case 5.1 VAV terminal box for space with non-uniform momentum*

Figure 4-18 illustrates the detailed Modelica model of VAV terminal box based on *Examples.VAVReheat.ThermalZones.VAVBranch* in the Modelica Buildings library. The model consists of a cold air source, a water-air heat exchanger with a valve in the water loop, an adjustable valve in the air loop, a controller, and multiple sensors. Since we isolated the room from a VAV system which serves multiple rooms, we assumed that the pressure difference at terminal box and space outlet was constant. Thus, we set the relative pressure (to the ambient pressure of 101 kPa) and pressure and temperature of the cold air source as 20 Pa and 16 ℃, respectively. The supply water temperature is set to be 50 ℃. The pressure difference at the water source and water sink is 12,000 Pa. Based on the actual room temperature and opening of the valve in the air loop, the controller can adjust the opening ratio of the valve in both air and water loops. If the minimal air flow rate is still too large for the cooling needs, the heat exchanger can heat up the air once the valve in the water loop is turned on.

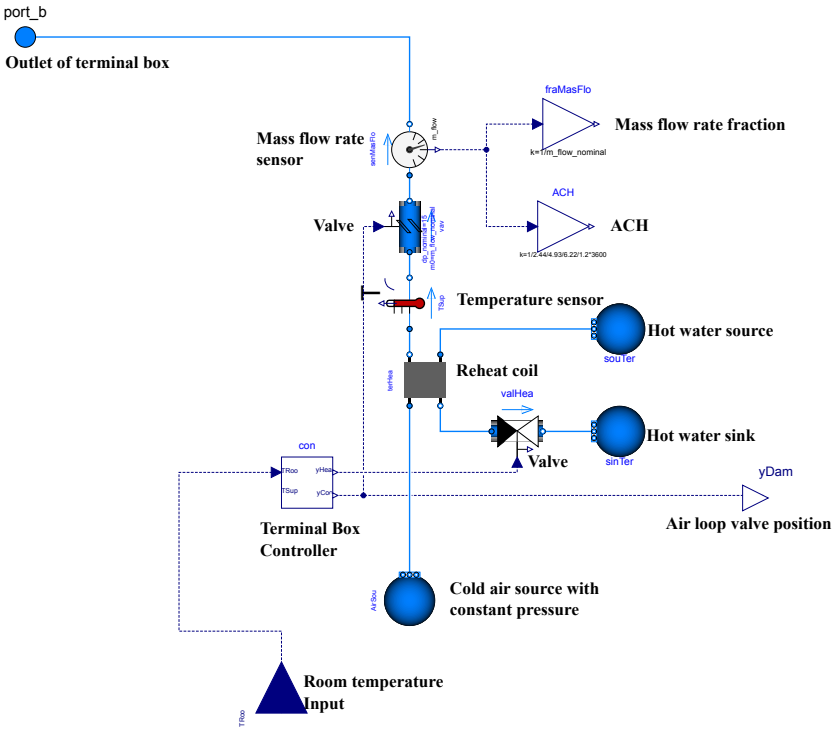



*Figure 4-18 VAV terminal box*

As shown in Figure 4-19, we implemented a pressure-dependent control logic (Liu et al. 2012) as an example based on *Examples.VAVReheat.Controls.RoomVAV*. Note that the control logic is not necessarily the one popularly used nowadays. Our focus here is to use it as an example to demonstrate the capability of the model. The occupant zone temperature signal is first sent to adjust the valve position in the air loop, which is at the lower part of the figure. A PI controller was employed to determine the signal. If the valve opening decreases to 30% (deemed as the lower limit) and the actual room temperature does not reach the set point, the reheat coil will turn on by feeding the opening position signal to the valve in the water loop. The control of the reheat coil is shown in the upper part of the figure. Similarly, another PI

controller was used to determine the signal. To avoid the short cycling of the reheat coil, we added hysteresis to the controller which has a lower bound of 0.3 and a higher bound of 0.4. With hysteresis, when the reheat was turned on and the opening signal to the valve in the air loop was between 0.3 and 0.4, the reheat of air continued.

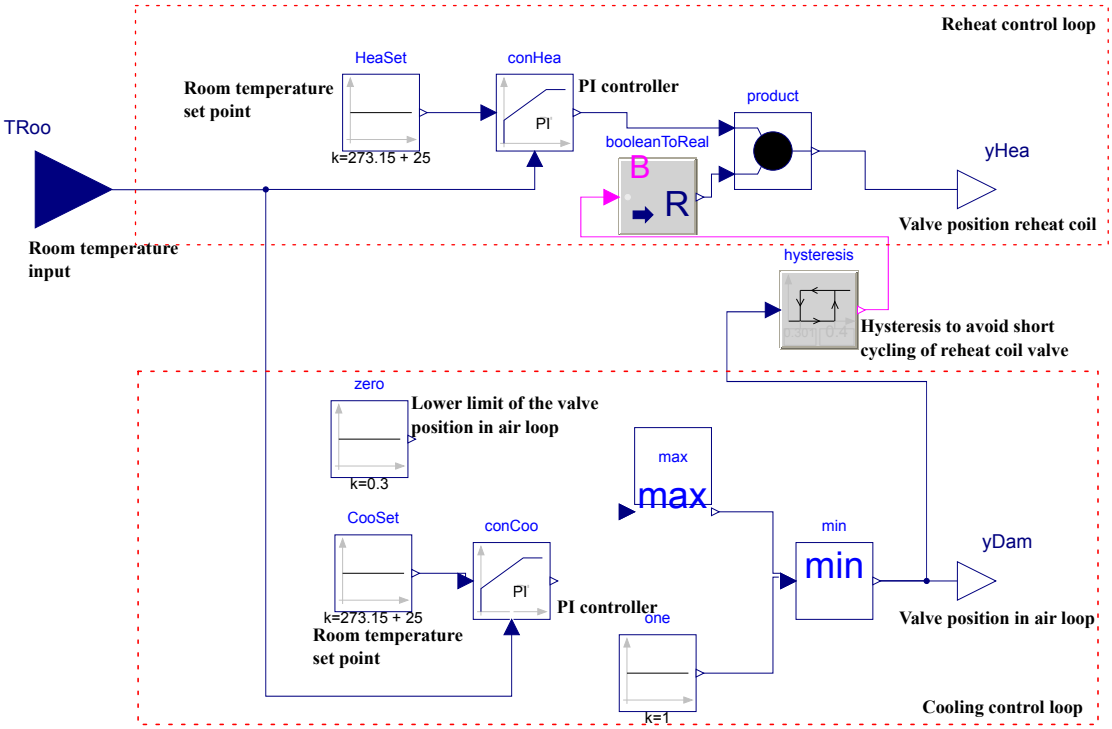



*Figure 4-19 Controller in VAV terminal box*

The case was simulated for 900 seconds using identical settings of solver and tolerance as the previous case. From Figure 4-20a-d, the dynamic response of the VAV terminal box and indoor environment are shown. In the beginning, as shown in Figure 4-20a, the room temperature is initially higher than the set point (25 ℃), the

opening ratio of the valve in the cold air loop decreases from 1.0 to 0.3 as shown in Figure 4-20b. The mass flow rate of the supply air as shown in Figure 4-20c then drops from 0.120 kg/s to 0.044 kg/s. Since the reheat coil does not turn on, the supply air temperature remains constant at 16 ℃, as shown in Figure 4-20d.

At around 60 seconds, when the opening ratio of the valve in the cold air loop reached 30%, and the room temperature was lower than the set point (Figure 4-20 a), the reheat coil is turned on. Then, the room temperature increases. However, it is still lower than the set point over the period from 60-160 s. Here, the opening ratio of the valve in cold air loop remains at the minimum of 30% and the opening of the valve in the reheat coil changes accordingly with the actual room temperature, as shown in Figure 4-20b. Consequently, one can see in Figure 4-20d that the supply air temperature first increases to a maximum of 25.4 ℃ and then gradually drops to 23.0 ℃, along with the change in the valve opening of the reheat coil.

From 160 to 225 seconds, the room temperature is higher than the set point and their difference is decreasing (Figure 4-20a). However, due to the hysteresis embedded in the controller and the opening of the valve in air loop being less than 0.4, the reheat coil remained on with a small opening (Figure 4-20b). Thus, the supply air temperature was higher than 16 ℃ and generally decreased with the valve opening becoming smaller (Figure 4-20d).

After approximately 225 seconds, the room temperature approached the set point (Figure 4-20a). At end of the simulation, the difference between room

temperature and the set point is marginal. Since the room temperature is higher than set point and the opening of the valve in air loop is larger than 0.4, the reheat coil turns off (Figure 4-20b) and supply air temperature remains at 16 ℃ (Figure 4-20d).

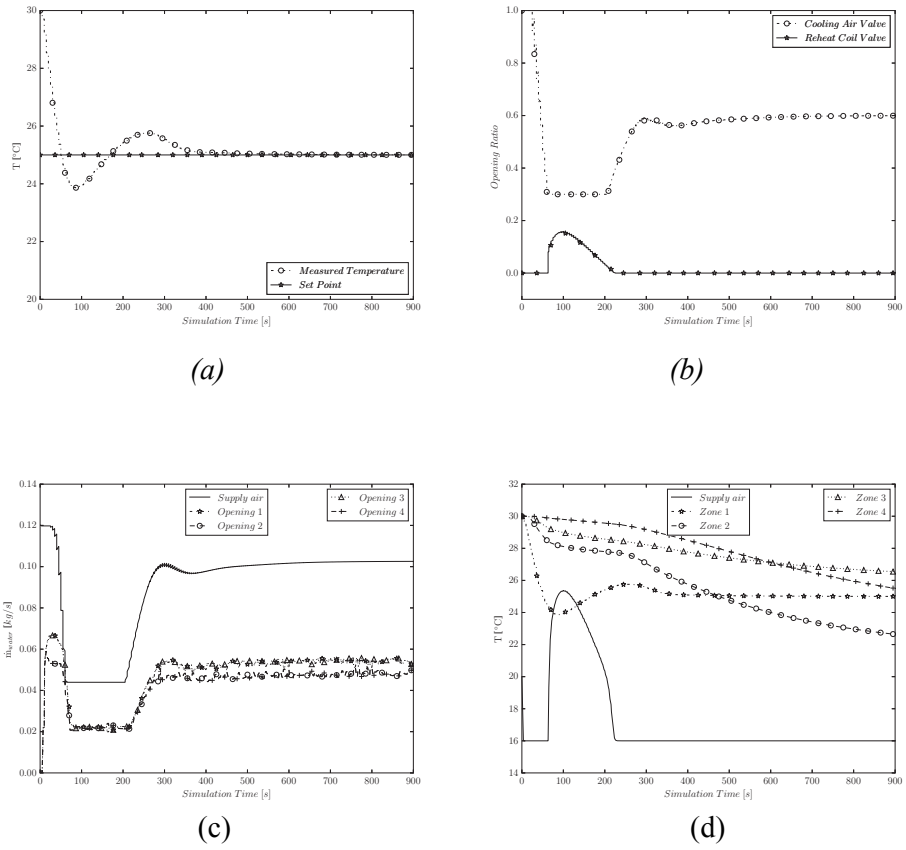



*Figure 4-20 (a) Zone 1 temperature control; (b) Control outputs from VAV terminal box; (c) Mass flow rates at different openings; (d) Zone temperature in the space*

Note that we presented the mass flow rate of the supply air at different openings in the space in Figure 4-20c. We can clearly identify the mass flow rate difference at *Opening 1* and *Opening 2*, which would be ignored if a multizone model were used. Due to the mass conservation law, the mass flow rate at *Opening 1* and *Opening 3* are equal, and the same rule applies to *Opening 2* and *Opening 4*.

### 4.5.2 Flow with Non-Uniform Temperature Coupled to A VAV Terminal Box

Based on the model (Figure 4-14) presented in section 4.4.2, this case study adds a VAV terminal box to substitute the prescribed fluid movers connected to *Zone 1*, as shown in Figure 4-21. The VAV terminal box was set to control the temperature of *Zone 2* as 26 ℃. The surface temperature for the floor and other walls in *Zone 2* are 25 ℃ and 27 ℃, respectively. The initial temperature of Zone 1 and the other spaces are 25 ℃ and 30 ℃, respectively. The VAV terminal box model is identical as the one in section 4.3.

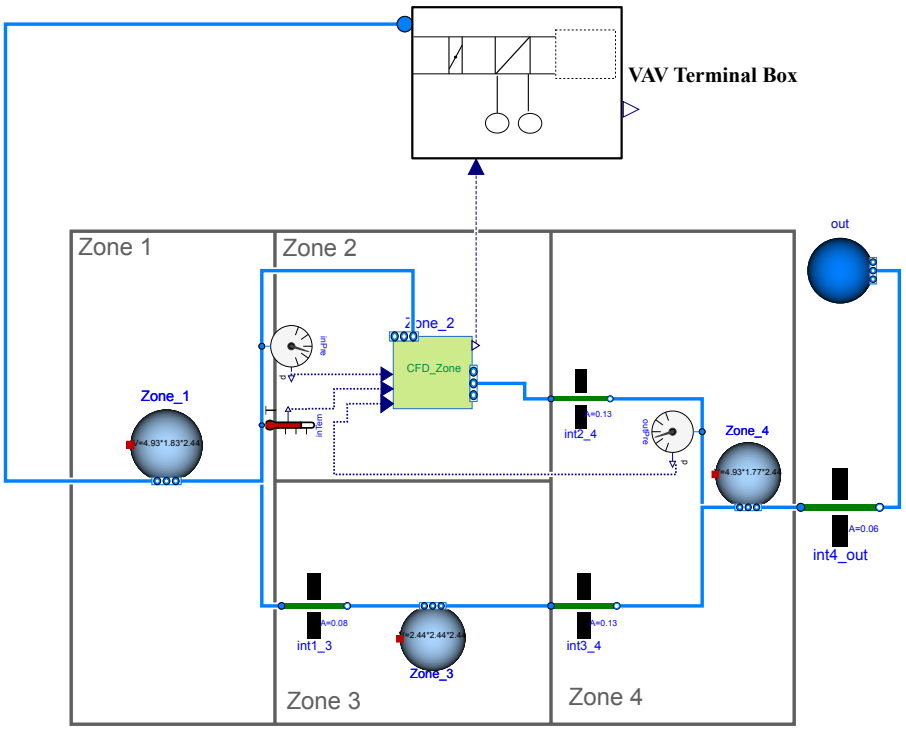



*Figure 4-21 VAV terminal box for space with non-uniform temperature distribution*

The case was simulated for 1800 seconds using identical settings for the solver and tolerance as in the previous case. Figure 4-22a-d show the results of the temperature control for *Zone 2*, control outputs of the VAV terminal box, the mass flow rate at different openings, and temperature in the zones. Since *Zone 2* is connected to *Zone 1* via *Opening 1*, the temperature at *Zone 2* decreases during the first 1000 seconds, in conjunction with the temperature at *Zone 1*, as shown in Figure 4-22d. As *Zone 2* is cooled by the cold air from Zone 1, the opening ratio of the valve in the air loop in the VAV terminal box is gradually turned down from 1.0 to 0.3.

Consequently, the mass flow rate of the supply air is reduced from 0.110 kg/s to 0.044 kg/s, which consequently decreases the flow rate of cold air into *Zone 2* and *Zone 3*. At around 1000 s, thanks to that the actual temperature of *Zone 2* being lower than the set point of 25 ℃ and the opening of the valve in the air loop reaching the minimal value, the VAV terminal box modulates to reheat the supply air, as shown in Figure 4-22b. As a result, the supply air temperature and temperature in *Zone 1* increases. Afterward, the opening ratio of the valve in the air loop increases and the reheat is terminated as it reaches to 0.4. Eventually, *Zone 2* is air-conditioned to the desired temperature of 25 ℃ with high control precision, as shown in Figure 4-22a.

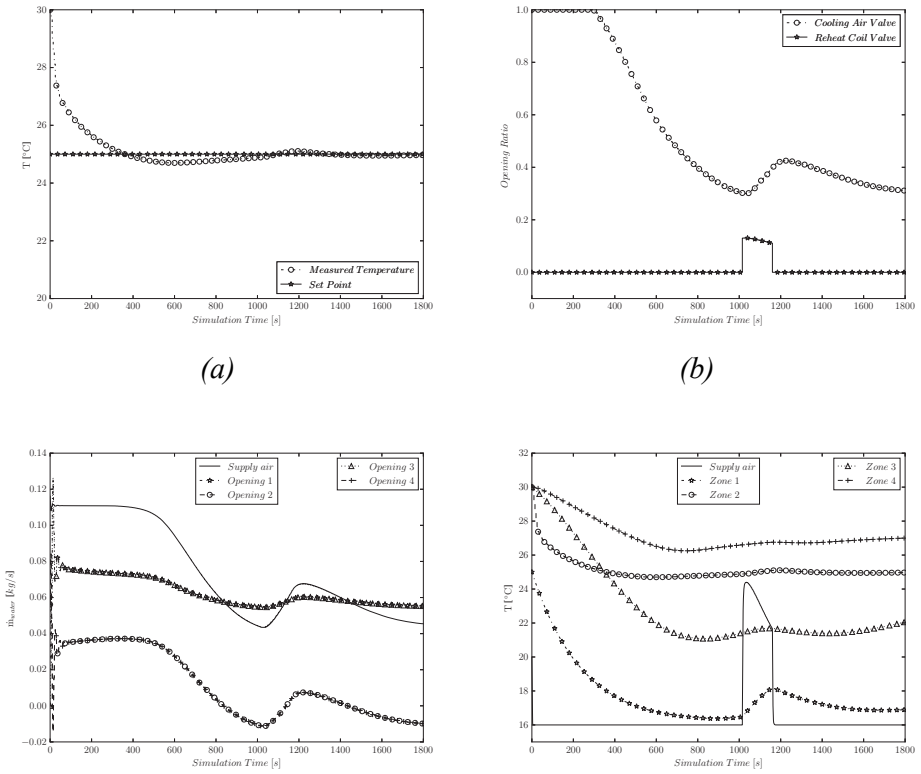
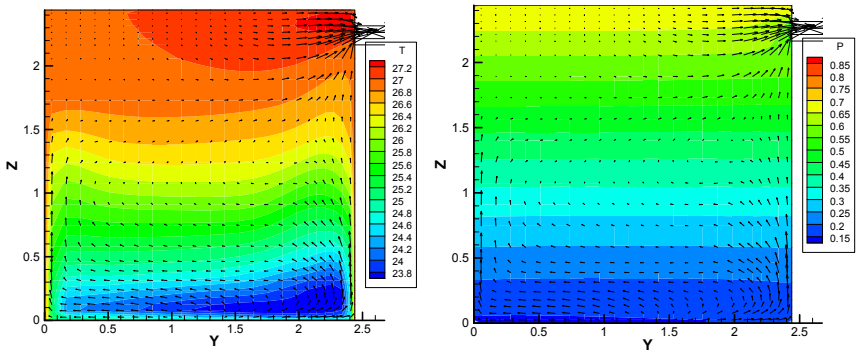



*Figure 4-22 (a) Zone 2 temperature control; (b) Control outputs from VAV terminal box; (c) Mass flow rates at different openings; (d) Zone temperature in the space*

The whole mechanism in this case study is like that in section 4.3 for the dynamic changes of all the variables, except for the mass flow rate. Interestingly, at some time intervals (i.e. from 850 to 1000, from 1400 to 1800 seconds), the supply air mass flow rate is lower than the that in *Opening 1*, which resulted in the reverse flow for *Zone 3*. As such, the pressure at the outlet of Zone 2 is larger than that at the inlet, leaving one to wonder why inflow still exists, in lieu of outflow at the inlet. The reason is that the flow in this case study is jointly determined by the inertia and

buoyancy forces. During those intervals when the opening ratio of the air valve in VAV terminal box is relatively small (<0.4), the total pressure is small which indicates that buoyancy force is dominant over the momentum force. As shown in Figure 4-23a, there is a strong temperature stratification that is induced by the heating box. The generated buoyancy force will form the stratified pressure distribution that increases from the bottom to the top, as shown in Figure 4-23b. As a result, the pressure in the adjacent cells to the outlet on average are 0.57 pa while the total pressure at the outlet is 0.45 pa.




(a) (b)

*Figure 4-23 temperature (a) and pressure (b) distribution at plan of X=2.32 at t=1800 s*

### 4.5.3 Flow with Non-Uniform Temperature Coupled to A VAV System

Based on the airflow network in section 5.2, we connected to each of 4 zones a dedicated VAV terminal box that constitutes a simplified VAV system, as shown in Figure 4-24. The VAV terminal boxes used here are identical to the ones in the previous sections. In this VAV system, the variable-speed fan drives the cold air of 16 ℃ from the air source to the terminal boxes through the pipes and splitters, and the return air gathered from the 4 zones flows back to the air sink. The fan speed is controlled by a PI controller to achieve a 140 Pa pressure difference between the supply and return ducts.

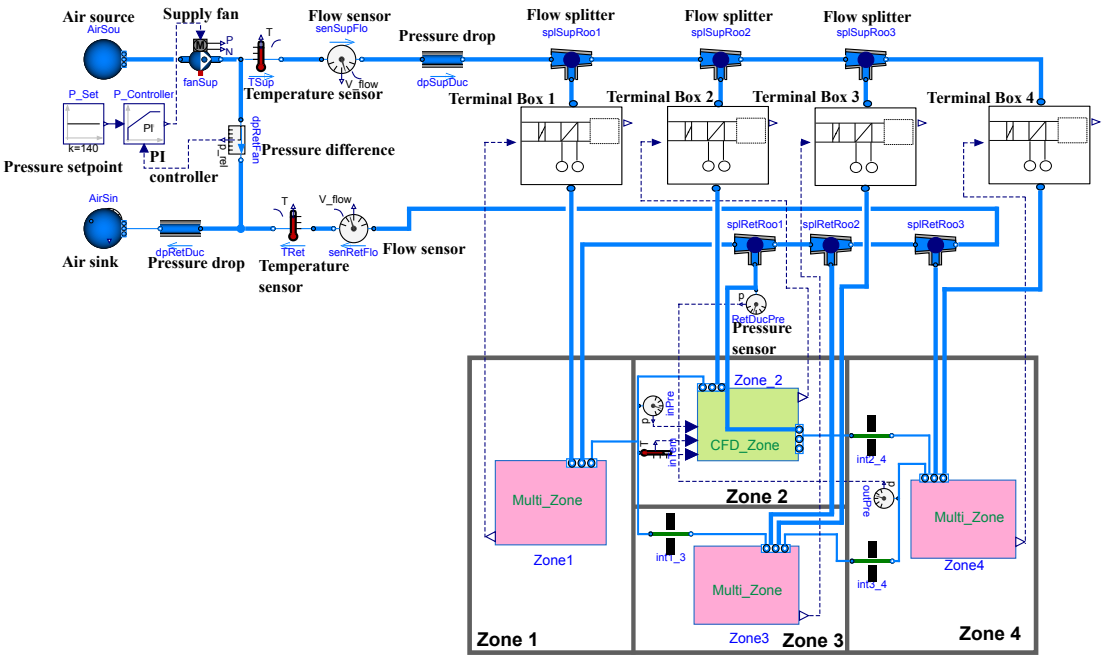



*Figure 4-24 Sketch of VAV system connecting four zones*

The inter-connection between four zones in the space is identical to that in Figure 4-21. Each zone has 2 additional openings to be connected to the inlet and outlet VAV terminal box. For *Zone 2*, the inlet opening and outlet opening, 0.45 m × 0.15 m, is located on the east wall and west wall, respectively. The *CFD_Zone* model is identical to that in section 4.4.2 while the *Multi_Zone* model is shown in Figure 4-25. In the center is the *Room.MixedAir* model that assumes that the air in the room is uniformly mixed. Fluid ports are defined to be connected to the openings of the zones and the VAV terminal box. The room model can calculate the heat transfer between envelopes, two constant temperature models are then set to define the wall surface temperature.

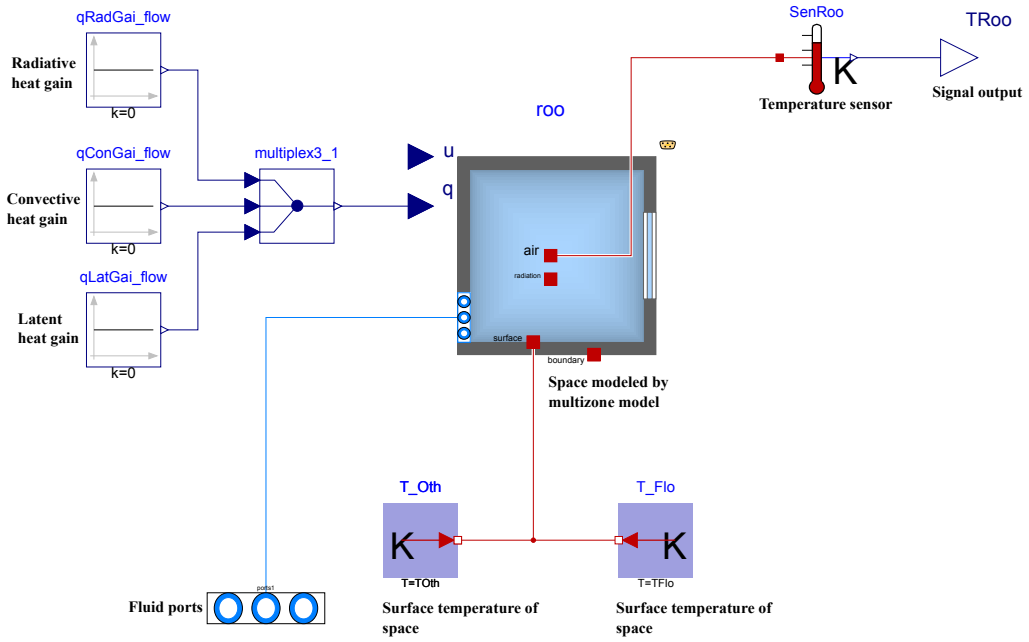


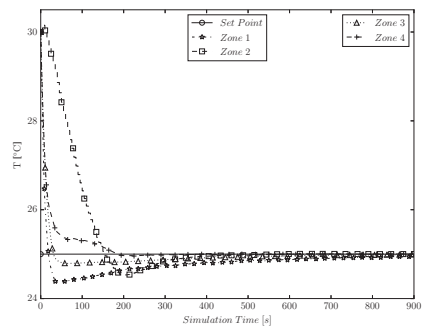

*Figure 4-25 Sketch of multizone model*

To mimic an office area, we defined the nominal mass flow rate for the VAV system as 0.1044 kg/s, leading to about 6 air changes per hour (ACH). The temperatures of the floor and other walls for *Zone 1*, *Zone 3*, and *Zone 4* are 27 ℃ and 25 ℃, respectively. For *Zone 2*, all the walls were set to be adiabatic, as a heat source is located on the floor, which gives out a similar heat gain as other zones. The initial temperature of the air in all zones is 30 ℃ and the target for the VAV terminal box control is to maintain the room temperature at 25 ℃.

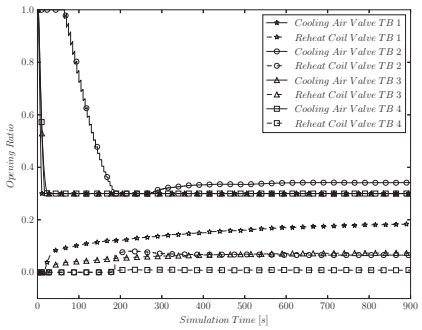
The case was simulated for 900 seconds using identical settings for the solver and tolerance as in the previous case. The dynamics of the VAV system are shown in Figure 4-26 a-e. As the room temperature approaches the set point, the injection of cold air causes the opening of the cooling air valve in the terminal boxes to decrease gradually to 30% prior to 200 seconds, as shown in Figure 4-26 b. Since pressure head for *terminal box 1* is the larger compared to *terminal box 3* and *terminal box 4*, we can find in Figure 4-26 a that the speed for lowering the temperature in *Zone 1* is faster, as more cold air is injected into that zone. The decreasing speed of the valve opening for *Zone 1*, *Zone 3*, and *Zone 4* is considerably faster because the air is assumed to be instantaneously well-mixed. However, for *Zone 2* which is simulated by FFD, the decreasing speed is much slower, as it takes longer time for the stratified air to be mixed and determined. Consequently, the mass flow rate for all the terminal boxes decreases as the resistance in the branch increases significantly due to the tightening of the valve in the terminal boxes.

As the cooling valve opening reaches 30% and the room temperature set point is not met, the reheat in the terminal box is turned on (at 30 s for *Zone 1* and *Zone 3*; at 200 s for *Zone 2* and *Zone 4*). Accordingly, we can see in Figure 4-26 d that the supply air temperature starts increasing instantly after the reheat coil is on. For *Zone 1*, *Zone 3* and *Zone 4*, the opening of the cooling valve in the terminal boxes remains constant at 30% and the opening of the valve in the water loop increases, as the actual temperature is lower than set point. However, for *Zone 2*, the opening of the cooling air valve increases a bit to 34% after reaching the lowest point.

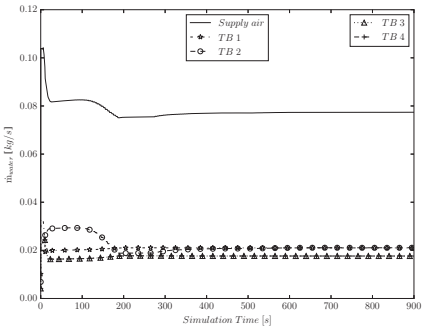
Eventually, as the actual room temperature reaches the set point, the reheat for all terminal boxes remains on, because the heat gain in the zone is relatively small. As shown in Figure 4-26 e, the pressure difference at the supply and return duct reach the set point of 140 Pa from 20 Pa, after decreasing the mass flow rate from 0.104 kg/s to 0.780 kg/s. At the beginning of the simulation when the duct resistance is small (cooling air valve opening is large) the supply mass flow rate of the fan is relatively large, and when the duct resistance increases (cooling air valve opening is small) as simulation progresses, the mass flow rate decreases. This is due to the operational point of the fan being fully coupled with the duct system.



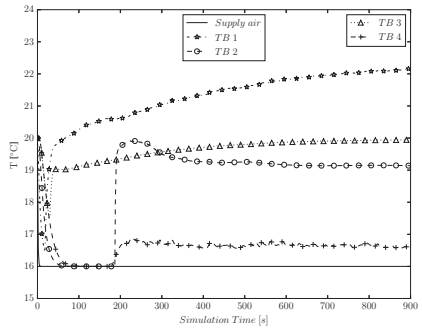

(a)



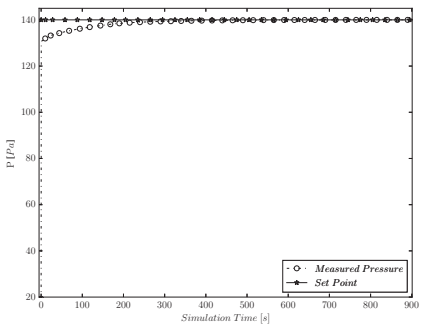

(b)




(c)




(d)




(e)

*Figure 4-26 (a)Temperature in all zones; (b) Control outputs from VAV terminal boxes; (c) Mass flow rates supplied by VAV terminal boxes; (d) Supply temperature by VAV terminal boxes;(e) Pressure difference at supply and return duct*

## 4.6 Conclusion

This chapter presented coupling FFD with multizone models in the Modelica *Buildings* library to study the dynamic response of HVAC systems in large spaces with stratified airflow distributions. The quasi-dynamic data synchronization strategy was used to fulfill the exchange of information for two scenarios. The first one is a simplified scheme in which FFD calculates the mass flow rate at outlets to feed to the multizone models. The other is more sophisticated as the multizone model gives a total pressure at fluid ports to FFD and FFD then returns the mass flow rates to the multizone model.

After implementing the two coupled simulation frameworks, we demonstrated its capability by studying two cases with either non-uniform momentum or non-uniform temperature distributions. The comparison between the simulated results and experimental data showed that the coupled models can capture the physics associated with the non-uniformity of a dynamic system. Further by linking the flows with a VAV terminal box and a VAV system, the coupled simulation models can capture dynamics of the studied system.

# Chapter 5 A Systematic Evaluation of Accelerating Indoor Airflow Simulations Using Cross Platform Parallel Computing

## *5.1 Background*

In order to reduce the computing time for airflow simulations to enable design optimization using coupled simulation models in Chapter 3 and Chapter 4, one can use supercomputers or cloud computing services (Gropp et al. 2001). However, this method is usually costly and may not be readily available. An alternate solution is to use multicore devices widely available in modern personal computers (Zuo and Chen 2009a, 2010a; Corrigan et al. 2011; Gorobets et al. 2013a; Gorobets et al. 2013b; Wang et al. 2011). These devices include GPUs, multi-core CPUs, Digital Signal Processors (DSPs), and other microprocessors. For instance, Zuo and Chen (2009b) sped up the CFD simulation 10-30 times by running it on a NVIDIA GeForce 8800 GTX GPU using CUDA (NVIDIA 2007).

However, CUDA only supports NVIDIA GPUs. A more appealing option is OpenCL, which supports GPUs, CPUs, Digital Signal Processors (DSPs), and other microprocessors from different manufacturers (Khronos 2012). Our literature review showed that there is only one indoor airflow simulation study using OpenCL (Wang et al. 2011). In their study, Wang et al. evaluated one Intel CPU and three NVIDIA GPUs on a Windows operating system using one case study. Despite providing speedup data, they did not provide sufficient validation of the OpenCL code in terms of result accuracy.

Although we would assume consistency in the numerical results across different hardware and operating systems using OpenCL, it remains critical to validate this assumption before adopting OpenCL for the indoor airflow simulation. Likewise, it is also interesting to observe how much speedup one can expect on different hardware. As a result, this study attempted to systematically evaluate the accuracy and speedup of cross-platform computing using OpenCL for indoor airflow simulations. In the investigation, we selected five CPUs and four GPUs differing in types, ages, and manufacturers. After implementing a FFD model using the OpenCL framework, we validated and evaluated both the FFD model and the OpenCL codes using four different cases that cover basic indoor airflows. CFD simulations results were also presented as a comparison. Finally, after analyzing the result consistency, speedup, and overall portability, we provided suggestions on using OpenCL for indoor airflow simulations.

## 5.2 Parallelization of FFD in OpenCL

The OpenCL implementation of FFD is shown in Figure 5-1. The program can be divided into a *host program* and its *kernels*. As stated in the OpenCL specification (Khronos 2012), the *host program* runs sequentially on the *Host* hardware (e.g. CPU) while the *kernels* run in parallel on the *device* hardware (e.g. GPU or other processors of the CPU). The entire implementation is a hybrid of C and OpenCL code. The C code is responsible for the main program structure while the OpenCL code is used to execute the *kernels.* The kernels are created based on the discretization of the governing equations introduced previously. These codes are then compiled in Mac OS

X using Xcode 7.0 (Xcode 2012) and in Windows using Microsoft Visual Studio 2013 professional (Microsoft 2013) together with the AMD APP SDK (AMD 2013).

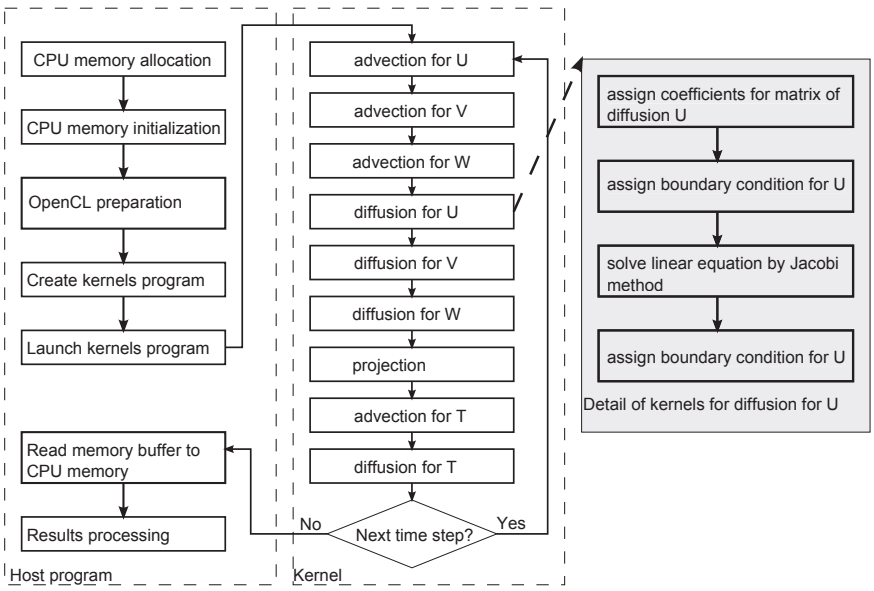



*Figure 5-1 Structure of parallelized FFD using OpenCL*

## 5.3 Numerical Experiment Settings

### 5.3.1 Hardware Device

As summarized in **Error! Reference source not found.**, four GPUs and five CPUs were selected. All five CPUs are manufactured by Intel while the GPUs are from AMD, NVIDIA, and Intel. Note that *CPU 1* and *CPU 5* are the same *Core i7 2620M* CPU installed on a MacBook Pro laptop. We named it as *CPU 1* under Windows 7 (running using Boot Camp) and *CPU 5* under Mac OS X. The peak performances for the hardware can be found online (NVIDIA 2012; AMD 2011, 2014; Intel 2012a, 2012b, 2012d, 2012c). In general, using double-precision floating

point (DPFP) can reduce the round-up errors so that the calculations can be more accurate than those in single-precision floating point (SPFP). However, some devices in this study, e.g. AMD GPUs, did not support the DPFP in OpenCL environment. To carry out the fair comparison, all simulations in this study were performed using SPFP.

*Table 5-1 Technique details of devices used in this study*

| Device | Manufacturer | Model | Year | Base Frequency (MHz) | Peak Performance in SPFP (GFLOPS) | Peak Performance in DPSP (GFLOPS) | Memory Bandwidth (GB/s) | # of Cores | OS |
|---|---|---|---|---|---|---|---|---|---|
| **CPU 1** | Intel | Core i7 2620M | 2011 | 2,700 | N/A | 43 | 21.3 | 2 | Win 7 |
| **CPU 2** | Intel | Xeon E5 1603 | 2012 | 2,800 | N/A | 90 | 31.4 | 4 | Win 7 |
| **CPU 3** | Intel | Core i7 4790 | 2014 | 3,600 | N/A | 230 | 25.6 | 4 | Win 7 |
| **CPU 4** | Intel | Core i5 3210M | 2012 | 2,500 | N/A | 40 | 25.6 | 2 | Mac OS X |
| **CPU 5** | Intel | Core i7 2620M | 2011 | 2,700 | N/A | 43 | 21.3 | 2 | Mac OS X |
| **CPU 6** | Intel | Core i7 3720 QM | 2012 | 2,600 | N/A | 83 | 25.6 | 4 | Mac OS X |
| **GPU 1** | Intel | HD Graphic 4000 | 2011 | 1,350 | 346* | N/A | 25.6 | 16 | Mac OS X |
| **GPU 2** | NVIDIA | GT 650M | 2012 | 850 | 653 | N/A | 28.8 | 384 | Mac OS X |
| **GPU 3** | AMD | FirePro V4900 | 2012 | 800 | 768 | N/A | 64 | 480 | Win 7 |
| **GPU 4** | AMD | FirePro W8100 | 2014 | 824 | 4,200 | 2,100 | 320 | 2,560 | Win 7 |

*Estimated due to lack of official information

### 5.3.2 Case Description

To evaluate the performance of the parallelized FFD in OpenCL, we selected four different cases which cover the basic indoor airflow types. The benchmark data is available for all four cases.

### 5.3.2.1 Flow in a Lid-Driven Cavity

The flow in a lid-driven cavity is shown in Figure 5-2. The dimension is *1 m* × *0.0233 m* × *1 m*. The top of the cavity is moving at a speed of *1 m/s*. The Reynolds number is set to be 400, based on the lid velocity, length of the cavity in the *X* direction, and kinematic viscosity. The benchmark data is available (Ghia et al. 1982). A non-uniform grid of *129* × *3* × *129* was used for the simulation.

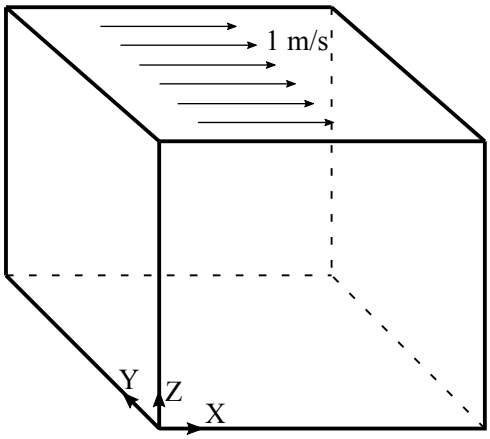



*Figure 5-2 Sketch of Lid-Driven Cavity case*

### 5.3.2.2 Forced Convection in an Empty Room

This case simulates an isothermal flow in an empty room (Wang and Chen 2009). The room size is *2.44 m* × *2.44 m* × *2.44 m* with other critical dimensions listed in Figure 5-3. The grid resolution is *40* × *40* × *40* and the inlet velocity is *1.36 m/s*. The experimental data (Wang and Chen 2009) is available at 10 different locations in Figure 5-4.

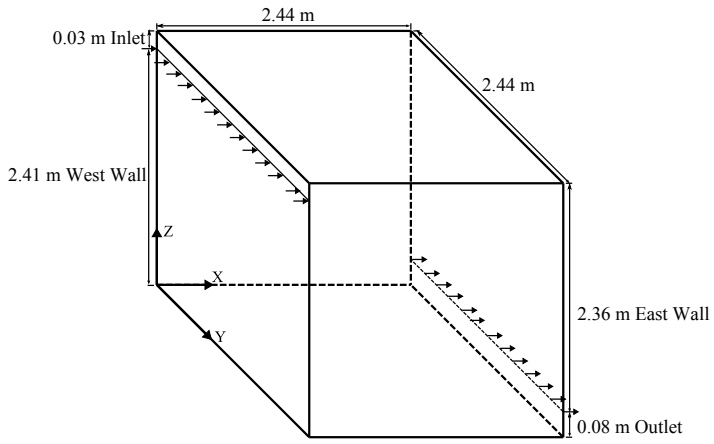



*Figure 5-3 Schematic of the forced convection in an empty room*

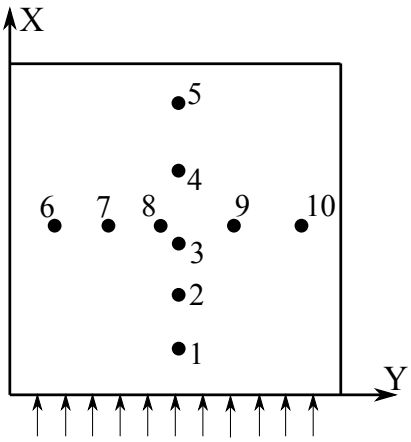



*Figure 5-4 the distribution of ten locations with experimental data available*

### 5.3.2.3 Forced Convection in a Room with a Box at Center

Based on the previous case, this case further increases the flow complexity by adding an obstacle (*1.22 m* × *1.22 m* × *1.22 m*) in the middle of the room (Figure 5-5). Again, detailed measurements at the locations described in Figure 5-4 are available (Wang and Chen 2009).

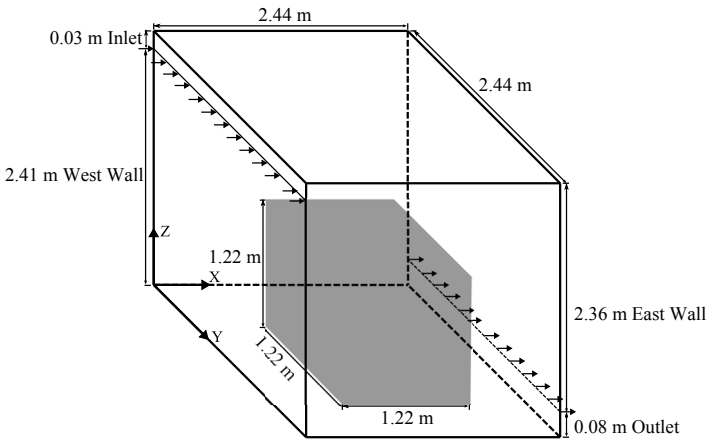



*Figure 5-5 Schematic of the forced convection in an empty room with a box.*

#### 5.3.2.4 Non-isothermal Flow with Stratified Distribution

Based on the case described in 3.2.3, this case analyses the non-isothermal flow with stratified distribution in a space, such as an aircraft cabin, by adding a heat source to the obstacle and controlling the temperature of the walls (Wang and Chen 2009). This case is identical to the one presented in chapter 3.4.1.

#### 5.3.2.5 CFD Simulation Setup

To validate the capability of FFD model, steady CFD simulations were performed for the above four cases using Fluent 16.1.0 (Fluent 2015) on the cloud. A laminar flow model was applied for the lid-driven cavity flow. A RNG $k$-$\varepsilon$ turbulence model (Yakhot and Orszag 1986) with the standard wall function was utilized for other cases as suggested by Chen (1995). The SIMPLE (Patankar and Spalding 1972) scheme was used to resolve pressure and velocity coupling. This study used standard scheme for pressure discretization and second-order upwind scheme for other equations discretization. The CFD simulation applied the same amount of grid as the FFD simulation although the grid distributions were different due to the wall function

applied in the CFD simulation. According to Wang and Chen (2009), the mesh grid used in the forced convection and non-isothermal flow with stratified distribution cases was fine enough to achieve grid independent results.

## 5.4 Analysis of Results

To provide clarity, we labeled the parallelized FFD in OpenCL as *OpenCL_FFD* and the sequential FFD in C as *C_FFD.* We employed two benchmarks including one from the experimental data and the other from the *C_FFD* on a single core of *CPU 2* (labeled as *C_Reference*).

### 5.4.1 Accuracy Evaluation

Figure 5-6-Figure 5-10 shows the comparison between *OpenCL_FFD* on the CPUs and GPUs with *C_Reference,* the CFD results, and the experimental data. Beginning with lid-driven cavity case which has a laminar flow, FFD (presented as *C_Reference*) is slightly worse than the CFD results which are perfectly aligned with the experimental data (Figure 5-6). By increasing the complexity of flows from laminar to turbulent, one can find in the forced convection the CFD still surpassed FFD in accuracy (Figure 5-7). FFD without the turbulence model was found to be deficient in capturing the flow features near the boundaries, especially the lower part of the profile. By further increasing the airflow features with an obstacle, one can see that even CFD with the turbulence model, cannot accurately predict the airflow near the boundaries (Figure 5-8). However, CFD still outperformed the FFD, which underpredicted the velocity magnitude, due to the omission of turbulent effect. Finally, in the non-isothermal flow with stratified distribution case (Figure 5-9 and

Figure 5-10) which was deemed the most complicated, interestingly the CFD predictions are closer with the experimental data than those in the forced convection (Figure 5-8). Again, FFD, due to lack of the turbulence models, predicted the trend with relatively poor accuracy.

To quantify the relative error between simulated and measured data, we employed a Euclidean norm estimator (Celebi et al. 2011):

$$\varepsilon = \frac{\sqrt{\sum_1^N \frac{(\hat{x}_i - x_i)^2}{x_i^2}}}{N}, \tag{25}$$

where $\hat{x}_i$ and $x_i$ are the simulated and measured value at *i* point, respectively; *N* is the total number of points selected for comparison. This estimator has been used to calculate the overall discrepancies at certain points between the results (Wang and Zhai 2012; Wang et al. 2010).

Table 5-2 summarizes the relative errors of FFD and CFD simulations, CFD simulations have relative errors within the 17% for all locations and cases. The averaged relative error for all locations and cases is 7.98%. As an intermediate method, FFD is less accurate than CFD in most locations. Surprisingly, the averaged relative error of FFD for the studied cases is 9.29%, which only slightly larger than that of CFD.

It is worth to mention that the CFD with the RNG *k-ε* turbulence model generated better results than that without the turbulence model, as was presented in

previous research (Jin et al. 2012b). This is consistent with the conclusion from Wang and Chen (2009) that RNG *k-ε* can generate overall good performance for cases 2-4.

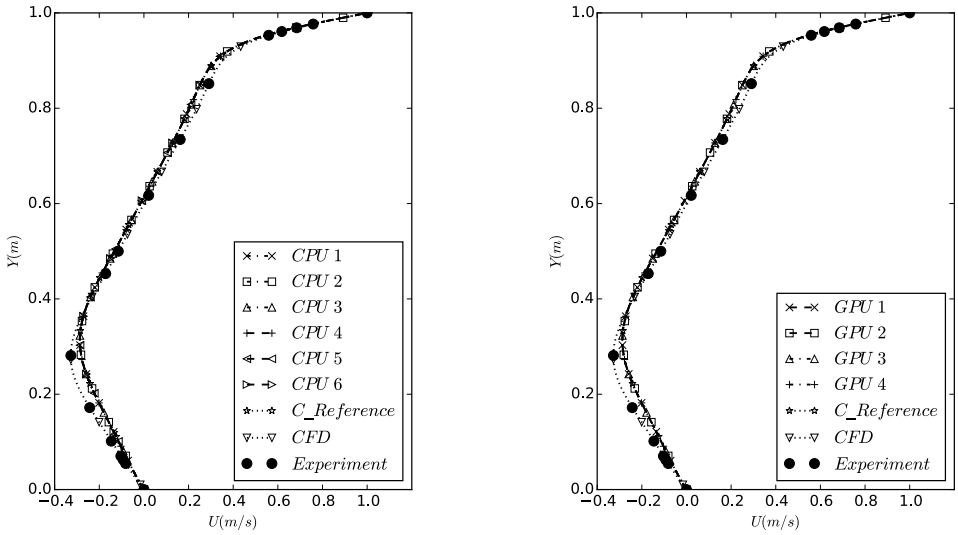
*Table 5-2 Relative Difference of Velocity Profiles Predicted by CFD and FFD*

| Case | Program | ε (%) | | |
|---|---|---|---|---|
| | | ***Location1**** | ***Location 3**** | ***Location 5*** |
| **Case 5.3.2.1: Flow in a Lid-Driven Cavity** | CFD | 0.18 | 2.88 | N/A |
| | FFD | 5.57 | 8.11 | N/A |
| **Case 5.3.2.2: Forced Convection in an Empty Room** | CFD | 5.53 | 16.70 | 7.67 |
| | FFD | 7.85 | 10.60 | 9.97 |
| **Case 5.3.2.3: Forced Convection in a Room with a Box at Center** | CFD | 14.86 | 5.53 | 16.37 |
| | FFD | 17.99 | 7.40 | 14.65 |
| **Case 5.3.2.4: Non-isothermal Flow with Stratified Distribution in a Room with a Box at Center** | CFD | 5.70 | 6.38 | 8.72 |
| | FFD | 10.40 | 3.67 | 5.98 |

*For case 5.3.2.1, Location 1 and Location 3 are the line at X=0.5m and the line at Z=0.5m, in the XZ plane which was sliced at Y=0.01165m, respectively.

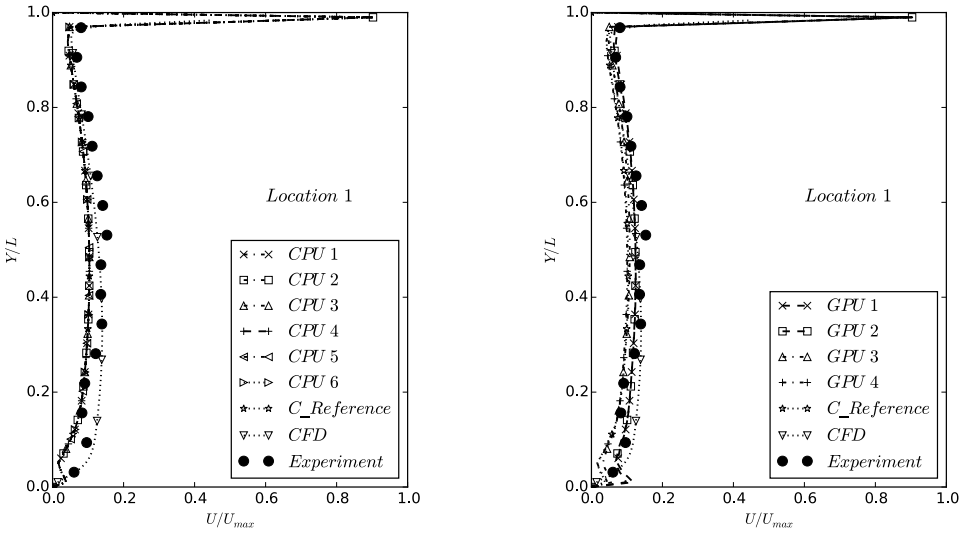
When running *OpenCL_FFD* on the CPUs, the results are the same as *C_Reference* for all the case studies (Figure 5-6a, Figure 5-7a, Figure 5-8a, Figure 5-9a, and Figure 5-10a). Surprisingly, results from *OpenCL_FFD* on the GPUs are not always consistent with *C_Reference.* When the flow is simple, e.g. lid-driven cavity case, all the GPUs generated identical results as *C_Reference* (Figure 5-6b). However, when the flow gets complex, the GPU results diverge slightly from

*C_Reference*, as well as from each other (Figure 5-7b, Figure 5-8b, Figure 5-9b and Figure 5-10b). This shows that the accuracy of the OpenCL_FFD depends on the executing GPU, which is contradictory to the hypothesis that all OpenCL devices should generate the same results.



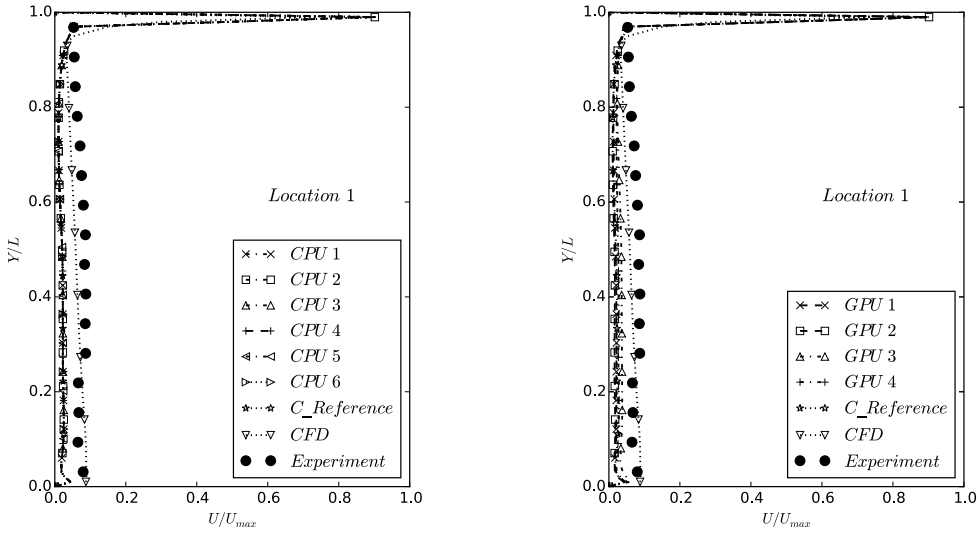

(a) (b)

*Figure 5-6 Horizontal velocity profiles in the vertical mid-section (X=0.5m) for the lid-driven cavity flow (case 5.3.2.1)*



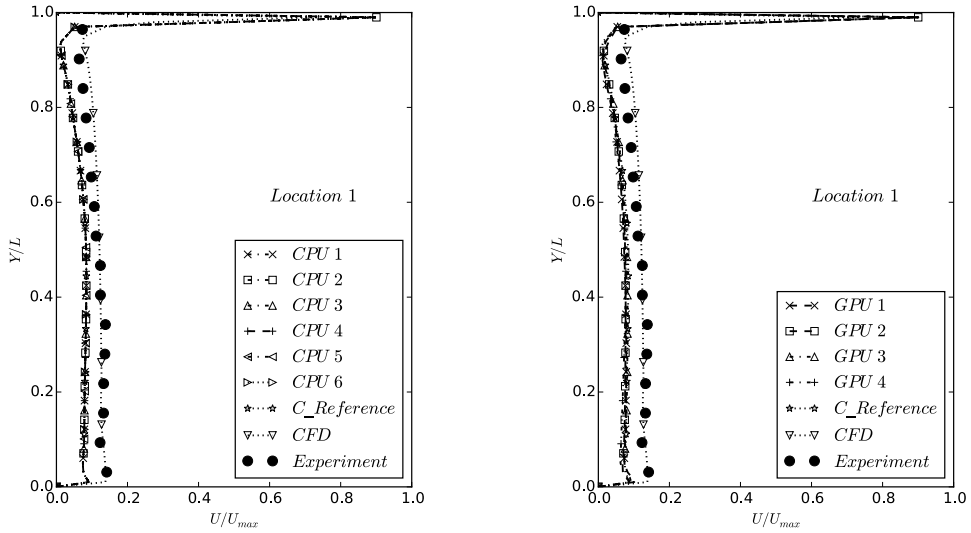

(a) (b)

*Figure 5-7 Comparison of velocity profiles for forced convection in an empty room (case 5.3.2.2)*



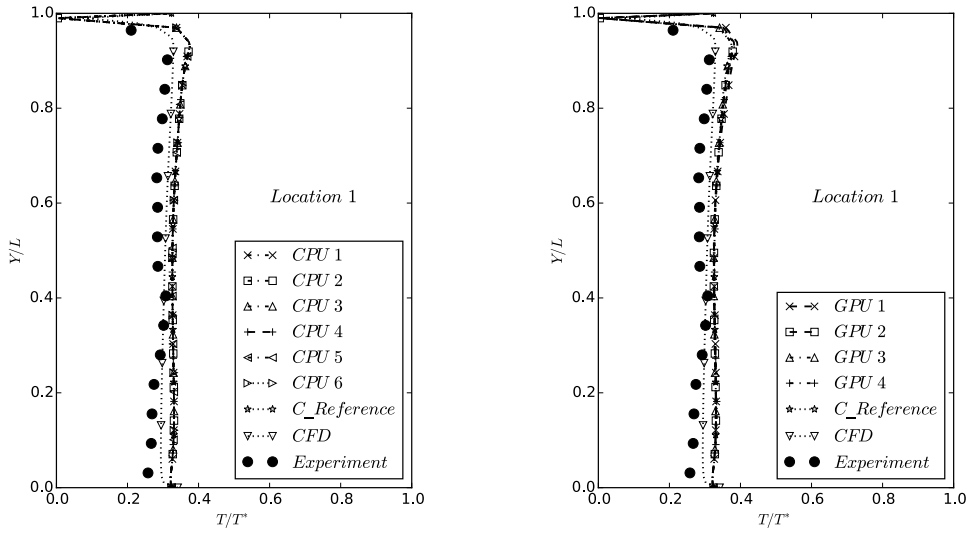

(a) (b)

*Figure 5-8 Comparison of velocity profiles for forced convection in a room with a box (case 5.3.2.3)*




(a) (b)

*Figure 5-9 Comparison of velocity for non-isothermal flow with stratified distribution in a room with a box (Case 5.3.2.4)*




(a) (b)

*Figure 5-10 Comparison of temperature for non-isothermal flow with stratified distribution in a room with a box (Case 5.3.2.4)*

We can use *Coefficient of Determination* $R^2$ to quantify the difference between *OpenCL_FFD* and *C_Reference.* The $R^2$ is defined as follows:

$$R^2 = 1 - \frac{\sum_1^n (x_i - \hat{x}_i)^2}{\sum_1^n (x_i - \bar{x})^2}, \tag{26}$$

where $x_i$ is the value at *i* point in the profile from *C_Reference*; $\bar{x}$ is the mean value of *n* points from *C_Reference*; $\hat{x}_i$ is the value at i point in the profile from *OpenCL_FFD*; *n* is the total number of points in the profile, which in this case is 100. Table 5-3 summarizes the $R^2$ for velocity and temperature profiles at different locations in Figure 5-7-Figure 5-9 for the three cases. For most cases, the value of $R^2$ is above 0.9 which indicates that the errors are not significant. Intriguingly, the value of $R^2$ seems independent to the GPU manufacturers, the complexity of the flow, or locations at which profiles were extracted.

*Table 5-3* $R^2$ *of the results from* OpenCL_FFD *on GPUs*

| | $R^2$ of velocity profile for the forced convection in an empty room | | | | $R^2$ of velocity profile for the forced convection in a room with a box | | | |
|---|---|---|---|---|---|---|---|---|
| | *GPU 1* | *GPU 2* | *GPU 3* | *GPU 4* | *GPU 1* | *GPU 2* | *GPU 3* | *GPU 4* |
| *Location 1* | 0.9106 | 0.9270 | 0.9946 | 0.9994 | 0.9994 | 0.9979 | 0.9825 | 0.9958 |
| *Location 3* | 0.9931 | 0.9902 | 0.9969 | 0.9993 | 0.9982 | 0.9969 | 0.9956 | 0.9964 |
| *Location 5* | 0.8907 | 0.8927 | 0.9852 | 0.9769 | 0.9678 | 0.9794 | 0.9794 | 0.9636 |
| | $R^2$ of velocity profile for the non-isothermal flow with stratified distribution in a room with a box | | | | $R^2$ of temperature profile for the non-isothermal flow with stratified distribution in a room with a box | | | |
| | *GPU 1* | *GPU 2* | *GPU 3* | *GPU 4* | *GPU 1* | *GPU 2* | *GPU 3* | *GPU 4* |
| *Location 1* | 0.9947 | 0.9969 | 0.9995 | 0.9931 | 0.9722 | 0.9939 | 0.9987 | 0.9941 |
| *Location 3* | 0.9846 | 0.9887 | 0.9966 | 0.9831 | 0.9990 | 0.9994 | 0.9999 | 0.9991 |
| *Location 5* | 0.9568 | 0.9544 | 0.9891 | 0.9700 | 0.9781 | 0.9706 | 0.9904 | 0.9786 |

To investigate why OpenCL on GPUs generated different results, we performed a numerical experiment to check the output at each step of *OpenCL_FFD* simulation. For example, Table 5-4 shows the comparison for case 3.2.2 at five control volumes. Two references were created by using *C_FFD* on the *CPU 2* and *CPU 4*.

The GPUs computed different values than the *C_Reference* at either the 1st or 100th time step. The difference is less than $2\times10^{-6}$ at the first step. This indicates that the inaccuracy may be a round-off error since SPFP is applied. However, the difference increased up to $5\times10^{-6}$ at the $100^{th}$ time step, which is likely due to the accumulation of the round-off errors.

A recent study (Gu et al. 2015) provided more insights on the inconsistency of the OpenCL-based calculations on the GPUs. It found that due to the lack of clarification in the current OpenCL specification, manufacturers could implement the Fused Multiply and Add (FMA) process in different ways, although they are all compatible with the *IEEE-754 2008 standard* (IEEE 2008). Moreover, current AMD-manufactured GPUs are not *IEEE-754* compatible since they implement a truncation instead of round-off operation during their FMA process. Therefore, the round-off errors observed in Table 5-4 are likely caused by the varying FMA implementations. As a result, the hypothesis that the OpenCL code will generate the same results on different results is not valid for the current OpenCL version 1.2 (Khronos 2012).

### *5.4.2 Computing Speed Evaluation*

#### 5.4.2.1 Case Study Speedup

We define the speedup *N* as

$$N = t_{bench}/t_{OpenCL}, \tag{27}$$

where $t_{bench}$ is the computing time used by the benchmark code (*C_Reference* on a *CPU)* and $t_{OpenCL}$ is the computing time used by *OpenCL_FFD* on different devices.

The speedup of *OpenCL_FFD* for the multi-core CPU was calculated based on the *C_Reference* using a single processor of the same CPU. The implementation of the *OpenCL_FFD* code flattened a two-dimensional array of variables into one-dimension, which reduced data access time when compared to *C_FFD*, which used a two-dimensional array for its variable storage. As a result, the optimization in implementation makes it possible that speedups of *OpenCL_FFD* can be higher than the number processors. For instance, the speedups of CPU 4 are larger than 2 which correlates with its number processors. It is also interesting to see that *OpenCL_FFD* on CPU 1 (which is on a Mac Computer running Windows using Boot Camp) is slower than the *C_Reference*. As a comparison, CPU 5, the same CPU on the Mac computer running on Mac OS X has a much higher performance. Since *OpenCL_FFD* achieved speedups on other CPUs while under a native Windows machine (CPU 2 and CPU 3), it is likely that the slowdown of CPU 1 is caused by the use of Boot Camp on the MacBook Pro laptop to run Windows.

*Table 5-4 Comparison of GPU results at 1st and 100th time steps in 5 control volumes for the non-isothermal flow with stratified distribution in a room with a box (Shading of cell indicates that the GPU result is different than the reference computed by CPU)*

| Device | OS | Value at selected points (i, j, k) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | U @ (2, 38, 34) | U @ (2, 38, 34) | U @ (2, 12, 38) | U @ (2, 12, 38) | T @ (13, 23, 40) | T @ (13, 23, 40) | V @ (1, 29, 35) | V @ (1, 29, 35) | U@ (14,23,38) | U @ (14,23,38) |
| | | 1st time step | 100th time step | 1st time step | 100th time step | 1st time step | 100th time step | 1st time step | 100th time step | 1st time step | 100th time step |
| **Reference 1** | Windows | 0.031247 | 0.033471 | 0.098339 | 1.359278 | 22.199999 | 22.200006 | 0.000635 | 0.000155 | 0.000000 | 1.351563 |
| **Reference 2** | Mac OS X | 0.031247 | 0.033471 | 0.098339 | 1.359278 | 22.199999 | 22.200006 | 0.000635 | 0.000155 | 0.000000 | 1.351562 |
| **GPU 1** | Mac OS X | 0.031247 | 0.033469 | 0.098339 | 1.359277 | 22.199999 | 22.200005 | 0.000634 | 0.000155 | 0.000000 | 1.351564 |
| **GPU 2** | Mac OS X | 0.031247 | 0.033469 | 0.098339 | 1.359277 | 22.199999 | 22.200005 | 0.000634 | 0.000155 | 0.000000 | 1.351564 |
| **GPU 3** | Windows | 0.031247 | 0.033469 | 0.098339 | 1.359278 | 22.200001 | 22.199999 | 0.000634 | 0.000155 | 0.000000 | 1.351564 |
| **GPU 4** | Windows | 0.031247 | 0.033471 | 0.098338 | 1.359278 | 22.200001 | 22.200001 | 0.000634 | 0.000155 | 0.000000 | 1.351563 |

*Table 5-5 Speedups of OpenCL_FFD on CPUs for all case study*

| Device | Number of Processors | Speedup | | | |
|---|---|---|---|---|---|
| | | Lid-driven cavity flow | Forced convection in an empty room | Forced convection in a room with a box | Non-isothermal flow with stratified distribution in a room with a box |
| **CPU 1** | 2 | 0.8 | 0.8 | 0.8 | 0.7 |
| **CPU 2** | 4 | 4.2 | 2.7 | 2.6 | 2.6 |
| **CPU 3** | 4 | 2.1 | 2.1 | 2.0 | 2.1 |
| **CPU 4** | 2 | 2.5 | 3.0 | 2.5 | 2.6 |
| **CPU 5** | 2 | 2.0 | 2.2 | 1.9 | 1.9 |
| **CPU 6** | 4 | 3.8 | 4.1 | 3.2 | 3.3 |

The speedup of *OpenCL_FFD* for the GPUs was calculated based on *C_Reference* on *CPU 2*. As shown in Table 5-6, a higher peak performance can lead to a larger speedup. For example, *GPU 4* has the highest peak performance which is about one order of magnitude larger than other studied GPUs. As a result, GPU 4 provided speedups which were one to two orders of magnitude higher than the others. However, the speedup is not perfectly proportional to the peak performance; other factors may also affect the speedup such as the global work size. The global work size is the number of all work items, which is equal to the total number of grids in our case. In next the section we discuss how the global work size affects the speedup.

*Table 5-6 Speedup of OpenCL_FFD on GPUs for all case study*

| Device | Peak Performance (FLOPS) | Speedup | | | |
|---|---|---|---|---|---|
| | | Lid-driven cavity flow | Forced convection in an empty room | Forced convection in a room with a box | non-isothermal flow with stratified distribution in a room with a box |
| **GPU 1** | 346* | 7.6 | 5.3 | 5.1 | 5.1 |
| **GPU 2** | 653 | 7.9 | 8.1 | 7.2 | 7.2 |
| **GPU 3** | 768 | 17.5 | 15.9 | 13.7 | 13.8 |
| **GPU 4** | 4,200 | 129.3 | 77.2 | 72.6 | 73.3 |

*Estimated due to lack of official information

### 5.4.2.2 Impact of Global Work Size

In order to analyze the impact of global work size on the performance of the devices, we measured the speedup of *OpenCL_FFD* using different ranges. The test was performed using a lid-driven cavity case described in section 3.2.1 but with a different dimension of *1m* × *1m* × *1m*. Three devices (*CPU 2*, *GPU 3*, and *GPU 4*) were examined in the test. Since the global work size is equal to the number of grids in our case, we can adjust the global work size by adjusting the number of grids.

As we can see from Figure 5-11, *OpenCL_FFD* on *CPU 2, GPU 3,* and *GPU 4* can eventually achieve a maximum speedup of 12, 140, and 1139, respectively. The speedup of OpenCL_FFD on *CPU 2* can be larger than its number of processors due to the optimization in the OpenCL implementation for more efficient data access, which is discussed in the section 4.2.1. Note that a threshold exists for each device which dictates

if the speedup increases or stalls with the increase of global work sizes. The threshold is about $1.25\times10^5$ for *CPU 2* and about $2.16\times10^5$ for *GPU 3* and *GPU 4*. When the global work size is below the threshold, the speedup increases with the work size because the computing capacity of the device is not fully utilized. After the work size exceeds the threshold, the speedup stops increasing because all the device's computing capacity is used up.

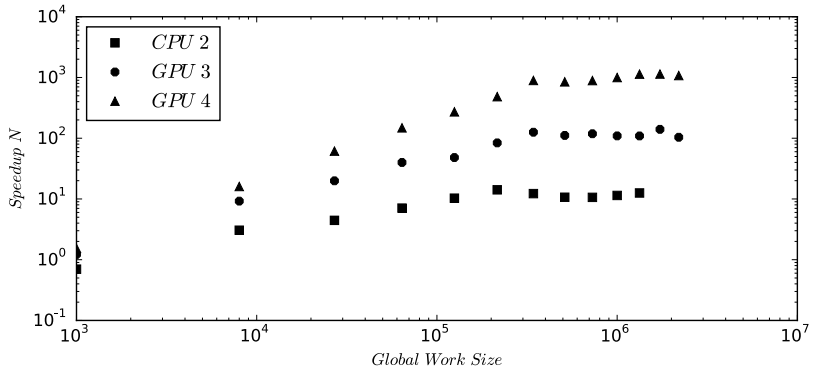



*Figure 5-11 Speedup of OpenCL with different global work size for the lid-driven cavity flow*

## 5.5 Conclusion

In this chapter, we evaluated two hypotheses for the cross-platform computing using OpenCL for indoor airflow simulation. The first hypothesis that the OpenCL code will generate the same results on different devices was not valid for the current OpenCL version 1.2. Although running the OpenCL code on the different CPUs produced the identical results, the results generated by GPUs differ with an $R^2$ larger than 0.9. The dissimilar results by GPUs are likely caused by the divergent FMA implementation from

different GPU manufacturers. Although the initial discrepancies are small at the level of $10^{-6}$, they can accumulate over time during simulations.

The second hypothesis that running in parallel on multiple processors of the same device will speed up the indoor airflow simulation was valid although the speedup is affected by the capacity of the device (e.g. peak performance) and the global work sizes. In addition, optimizing the data access can provide an additional speedup. A separate study on the relationship and number of grids showed that a speedup of 1139 times can be achieved using an AMD FirePro W8100 GPU.

In addition, the comparison of FFD and CFD with RNG $k$-$\varepsilon$ model showed that both CFD and FFD can predict the studied flows with averaged relative errors of 7.98% and 9.29%, respectively.

# Chapter 6 Fast and Self-Learning Indoor Airflow Simulation Using In Situ Adaptive Tabulation and Fast Fluid Dynamics

## *6.1 Background*

In order to reduce the computing time for airflow simulations to enable operation optimization using coupled simulation model in Chapter 3 and Chapter 4, it is feasible to use ROMs (Kolokotsa et al. 2009; Hazyuk et al. 2012; Desta et al. 2004; Hiyama et al. 2010; Kim et al. 2015; Ahuja et al. 2011). A common approach is to use a regression model with a limited number of inputs in order to construct the data-driven ROMs based on pre-calculated CFD results (Chen and Kooi 1988). However, they can rarely reflect the dynamics of a full order CFD model. On the other hand, ROMs can be built by using the Principal Orthogonal Decomposition method to extract important features (snapshots) of the flow and then project them to a Linear Time Invariant system (Li, Su, et al. 2013). Such ROMs can partially maintain the dynamics of the full order CFD model. Although it can be time-consuming to run various CFD simulations to generate training data, the trained ROMs can compute the solution almost instantaneously by either interpolating or extrapolating using an existing data set. However, conventional ROMs can only perform well when the inputs are within or near the training domain. Consequently, if the inputs are too far outside the training domain, the ROMs may resolve them without any guaranteed accuracy (Stockwell and Peterson 2002).

Obviously, it is too expensive to train a ROM for a domain which includes all the possible inputs of the application. Therefore, to overcome this drawback of conventional ROMs, we propose to a fast and self-learning indoor airflow simulation method. The idea is that we will train the ROM within a domain in which the system is most likely to operate. If the trained ROM cannot project the solutions accurately, a full-scale CFD simulation will be executed. The newly generated data from the CFD simulation will then be used to enlarge the training domain for the ROM.

To realize the proposed fast and self-learning airflow simulation method, we selected an ISAT algorithm. ISAT is a general function approximation method. ISAT was originally proposed to speed up combustion simulations (Pope 1997). It stores key simulation data in a data table and linearly interpolates the solutions from the table if the inputs are within the region where the interpolation accuracy is guaranteed. Otherwise, it executes a full-scale simulation to obtain the solution.

Given that FFD is a full-scale airflow simulation model of high speed, the ISAT, a fast and self-learning approach, can be evaluated on accelerating indoor airflow simulation using FFD. Although ISAT has accelerated multi-species combustion simulations (Singer and Pope 2004; Singer et al. 2006), it is not clear that ISAT will be suitable for indoor airflow simulations. This study implemented and then evaluated the performance of the proposed ISAT-FFD model. We first introduce mathematical

descriptions of FFD and ISAT. We then illustrate the implementation of the ISAT-FFD model on a hybrid computing platform consisting of a central processing unit (CPU) and a GPU. Afterward, using a stratified indoor airflow, we assess the performance of ISAT at the training stage as well as the evaluation stage. Finally, we present the conclusion and potential applications of this research.

## 6.2 In Situ Adaptive Tabulation

### 6.2.1 Mathematical Description of ISAT

For a nonlinear model such as CFD or FFD, we describe the outputs $\boldsymbol{y}$ as a function of the inputs $\boldsymbol{x}$:

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) \tag{28}$$

$$\boldsymbol{x} = \{x_1, x_2, \dots, x_m\} \tag{29}$$

$$\boldsymbol{y} = \{y_1, y_2, \dots, y_n\} \tag{30}$$

where $\boldsymbol{x}$ is a set of independent scalar variable $x_i$; and $\boldsymbol{y}$ is a set of dependent scalar variable $y_i$. As an example, if a *query point* $\boldsymbol{x}^{(q)}$ is close to a *tabulation point* $\boldsymbol{x}^{(0)}$, ISAT can estimate $\mathbf{y}(\boldsymbol{x}^{(q)})$ using a linear interpolation:

$$\mathbf{y}\left(\boldsymbol{x}^{(q)}\right) \approx \boldsymbol{y}^{\mathrm{l}}(\boldsymbol{x}^{(q)}) = \boldsymbol{y}\left(\boldsymbol{x}^{(0)}\right) + \boldsymbol{A}\left(\boldsymbol{x}^{(0)}\right)\delta\boldsymbol{x} \tag{31}$$

$$\boldsymbol{A_{ij}}\left(\boldsymbol{x}^{(0)}\right) = \frac{\partial \boldsymbol{f}_i\left(\boldsymbol{x}^{(0)}\right)}{\partial \boldsymbol{x}_j}, \delta \boldsymbol{x} = \left(\boldsymbol{x}^{(q)} - \boldsymbol{x}^{(0)}\right) \tag{32}$$

where $\boldsymbol{A}\left(\boldsymbol{x}^{(0)}\right)$ is called the mapping gradient matrix ($n \times m$) at $\boldsymbol{x}^{(0)}$ and it is stored together with $\boldsymbol{x}^{(0)}$ in the record (Pope 1997); here, $i$ and $j$ are index. Note that the superscript which is located inside the parenthesis, for example, in $\boldsymbol{x}^{(\boldsymbol{q})}$, $q$ denotes the $q_{th}$ query point. The parenthesis is used to avoid confusion of interpreting $\boldsymbol{x}^{(\boldsymbol{q})}$ as $\boldsymbol{x}$ to the power of $q$. To calculate $\boldsymbol{A}\left(\boldsymbol{x}^{(0)}\right)$, we can use user defined function or the ISAT algorithm using two neighbor points in the table. For instance, the ISAT algorithm finds neighbour record to $\boldsymbol{x}^{(\boldsymbol{0})}$, saying $\boldsymbol{x}^{(\boldsymbol{1})}$. Then the mapping gradient matrix can be calculated as below.

$$\boldsymbol{A_{ij}}\left(\boldsymbol{x}^{(0)}\right) = \frac{\boldsymbol{f}_i\left(\boldsymbol{x}^{(0)}\right) - \boldsymbol{f}_i\left(\boldsymbol{x}^{(1)}\right)}{\boldsymbol{x}^{(0)}{}_j - \boldsymbol{x}^{(1)}{}_j}. \tag{33}$$

ISAT can automatically detect if a linear interpolation can be performed for $\boldsymbol{x}^{(q)}$ based on its relationship to the region of accuracy of $\boldsymbol{x}^{(0)}$ within which the interpolation error is not larger than error tolerance. Finally, the interpolation error $\varepsilon$ is defined as:

$$\varepsilon = \left\| \boldsymbol{B}\left(\boldsymbol{y}\left(\boldsymbol{x}^{(q)}\right) - \boldsymbol{y}^{l}\left(\boldsymbol{x}^{(q)}\right)\right)\right\|_2 \leq \varepsilon_{total} \tag{34}$$

where $\varepsilon_{total}$ is the total error tolerance for all outputs, which is a scalar variable set by the user. $\boldsymbol{B}$ is an $n \times n$ scaling diagonal matrix which is predefined by the users before

executing ISAT (Pope 1997). Its primary function is to make the interpolation error of each output comparable given that number of outputs is larger than one. For example, suppose an output consisting of velocity magnitude and temperature, and velocity magnitude and temperature varying at 0.01 m/s and 0.1 $^oC$, it is important to multiply the interpolation error of velocity magnitude by a factor of 10 to ensure that the accuracy of velocity magnitude prediction is not neglected.

Now, by assuming a constant approximation of $\boldsymbol{y}(\boldsymbol{x}^{(q)}) = \boldsymbol{y}(\boldsymbol{x}^{(0)})$, and substituting Equation (31) in Equation (34), one obtains:

$$\varepsilon = \|\boldsymbol{B}\boldsymbol{A}\delta\boldsymbol{x}\|_2 \leq \varepsilon_{total} \tag{35}$$

$$\frac{\delta\boldsymbol{x}^T\boldsymbol{A}^T\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{A}\delta\boldsymbol{x}}{{\varepsilon_{total}}^2} \leq 1 \tag{36}$$

Equation (36) is the criterion used to determine if the linear interpolation Equation (31) is valid for point $x^{(q)}$. By subsequently applying Cholesky decomposition (Tuma 2002) to the semi-definite symmetrical matrix $\frac{\boldsymbol{A}^T\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{A}}{{\varepsilon_{total}}^2}$, one can obtain the equation below,

$$\delta\boldsymbol{x}^T\boldsymbol{Q}^T\boldsymbol{\Lambda}\boldsymbol{Q}\delta\boldsymbol{x} \leq 1 \tag{37}$$

where $\boldsymbol{Q}$ is an $m \times m$ unitary matrix and $\boldsymbol{\Lambda}$ is an $m \times m$ diagonal matrix. Equation (37) defines a hyper-ellipsoid area, which is called the *Ellipsoid of Accuracy* (EOA) in ISAT. The radius in the $i$ direction, $l_i$, is defined as:

$$l_i = \frac{\varepsilon_{total}}{\sigma_i} \tag{38}$$

The $\sigma_i$ is the $i_{th}$ diagonal entry of $n \times m$ diagonal matrix $\boldsymbol{\Sigma}$, which is derived from Singular Value Decomposition of $\boldsymbol{BA}$. The sketch of the EOA in different dimensions is shown in Figure 6-1. It is worthy to note that due to the incorporation of the constant assumptions and essence of linear approximation of nonlinear function, it is not guaranteed that the interpolation error $\varepsilon$ is less than the error tolerance $\varepsilon_{total}$ for all the points in the EOA (Pope 1997).

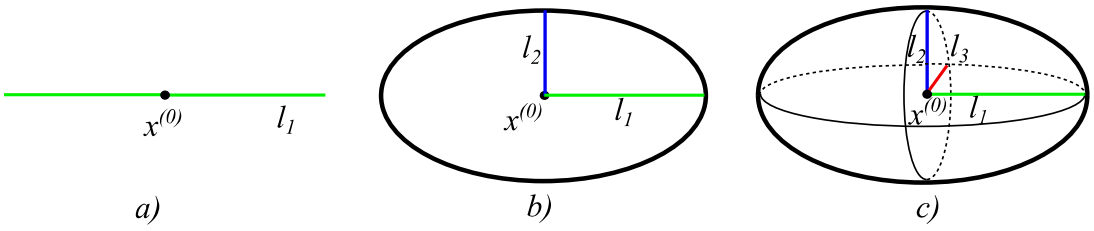



*Figure 6-1 Sketch of EOA in different dimensions (a: 1D; b: 2D; c: 3D)*

### 6.2.2 ISAT Workflow

Figure 6-2 shows the workflow of ISAT. Given a query point $\boldsymbol{x}^{(q)}$, ISAT will perform one of the following three actions: *retrieve*, *grow* or *add*. As the first step (step

1.1), ISAT will look up in the data table and find the nearest data point to $\boldsymbol{x}^{(q)}$, which is assumed hereby to be $\boldsymbol{x}^{(0)}$. Then step 1.2 is used to check if the query point $\boldsymbol{x}^{(q)}$ is within the EOA of $\boldsymbol{x}^{(0)}$. If so, the *retrieve* action will be executed using Equation (31) in step 2.1, and its linear interpolation $\boldsymbol{y}^{l}\left(\boldsymbol{x}^{(q)}\right)$ will be returned in step 2.2. If not, an evaluation of the nonlinear Equation (28) in step 3.1 will be performed. Furthermore, the inequality (34) will be used to determine if the difference between the solution of Equations (28) and (31) is less than the error tolerance $\varepsilon_{total}$ in step 3.2. If the inequality (34) is met, the *grow* action will be performed to enlarge the EOA with minimal volume increase to contain the query point $\boldsymbol{x}^{(q)}$ by updating the matrix $\boldsymbol{Q}$ and $\boldsymbol{\Lambda}$ in Equation (37) in step 4 using Householder matrix algorithm and rank-one modification algorithm (Pope 2008). The query point $\boldsymbol{x}^{(q)}$ is going to be abandoned. Otherwise, the *add* action will be performed to add the query point $\boldsymbol{x}^{(q)}$ as one additional record in the table of step 5.

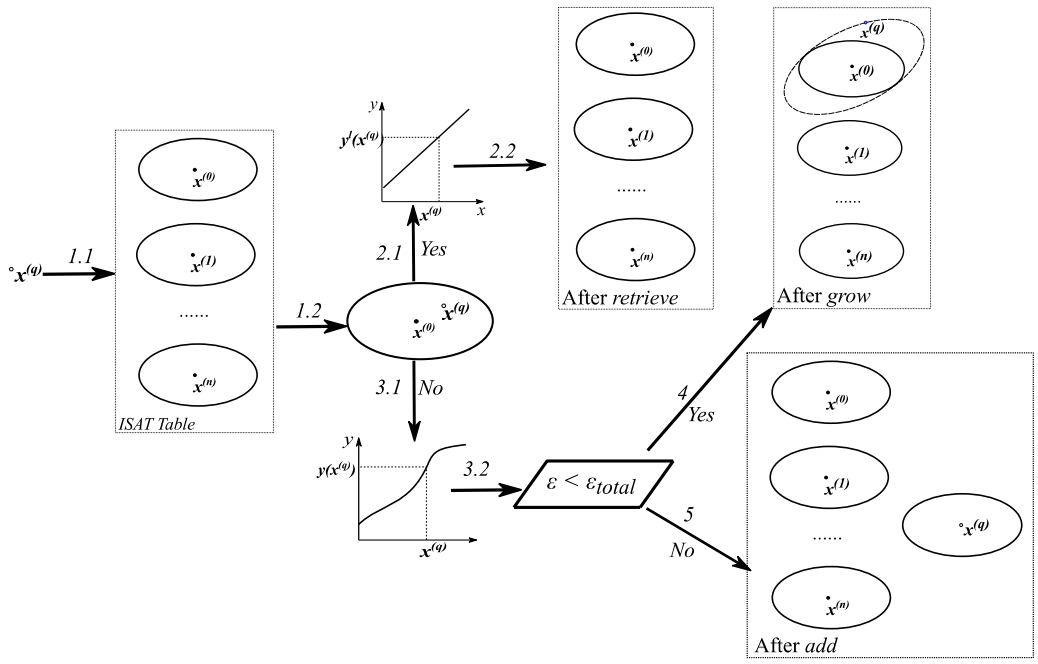



*Figure 6-2 Workflow of ISAT*

### 6.2.3 Training Method for ISAT

As a self-learning method, the ISAT's learning time depends on the training method. Besides a conventional *constant interval* method as sketched in Figure 6-3(a), we propose an *automatic interval refinement* method to generate the training data as shown in Figure 6-3(b). The *constant interval* method requires manually defining the interval in each dimension of $\boldsymbol{x}$. After all the query points are evaluated, the table is considered fully trained. The *automatic interval refinement* method dynamically refines the interval by halving the last one if the training is not complete. The completeness of training can be measured by:

$$N_{add+grow} \leq \eta N_{queries} \tag{39}$$

where $N_{add+grow}$ is the total number of *add* and *grow* under the current interval, $N_{queries}$ is the number of queries, and $\eta$ is a constant coefficient. By default, $\eta = 0$ and $N_{add+grow} = 0$ indicate that the training is complete if under the current interval there are no *add* and *grow* actions generated.

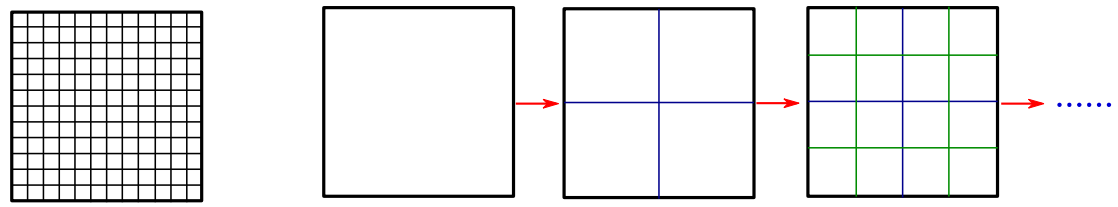



*Figure 6-3 Constant interval method and automatic interval refinement method to train the ISAT table.*

## 6.3 ISAT-FFD Integration

The ISAT algorithm and FFD models have been implemented in previous studies (Pope 1997; Zuo and Chen 2009c). We will now describe the implementation of the ISAT and FFD coupling, which enables setting up the ISAT algorithm, launching the FFD simulation, and extracting the FFD results for the ISAT algorithm. As shown in

Figure 6-4, the first stage in the ISAT-FFD scheme is initialization. This includes setting up ISAT parameters, e.g. error tolerance and dimensions of inputs and outputs,

and defining the training range for the studied problem. In the training stage, query points within the training domain are generated to populate the ISAT data table. If using the *constant interval* method, the ISAT table is completely trained once all the generated points are evaluated. If using *automatic interval refinement* method, the ISAT table training is completed once inequality (39) in section 3.3 is met. After the ISAT table is trained, the program moves to the evaluation stage, where query points within the evaluation domain are generated as inputs. During the evaluation stage, most queries will be handled by *retrieve* and the remaining few queries will be answered by calling FFD.

The ISAT-FFD framework was implemented using C code. To speed up the FFD simulation, a FFD program running in parallel on a graphic processing unit (GPU) was developed (Tian et al. 2017). The parallelization was realized using a hybrid code of C and OpenCL language (Khronos 2012). Cornell University provided the original ISAT source codes written in FORTRAN. The source codes are available at: *https://tcg.mae.cornell.edu/isat.html*. The simulation was performed using a DELL workstation with a Xeon E5-1603 CPU and AMD FirePro W8100 GPU.

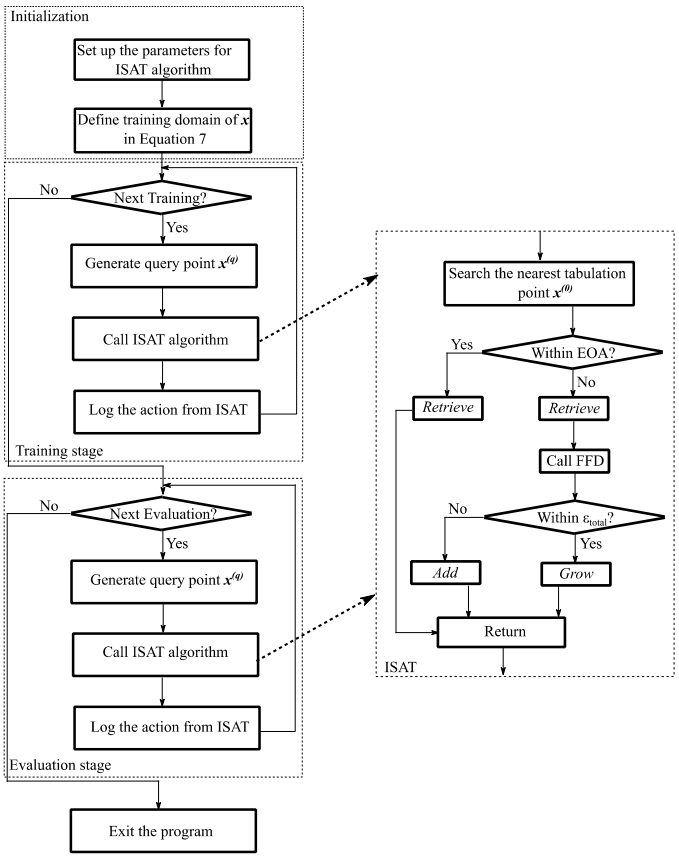



*Figure 6-4 Framework of ISAT-FFD approach*

## 6.4 Numerical Experiments

To evaluate the feasibility and performance of the proposed ISAT-FFD model for indoor airflow simulations, we evaluated it using a non-isothermal flow with stratified distribution in a room with a box at the center. Since the ISAT algorithm only allows one error tolerance for all outputs, we used the scaling matrix $\boldsymbol{B}$ defined in Equation (34) to

convert the errors from different outputs into a single aggregated error. However, users will have to construct the scaling matrix $\boldsymbol{B}$ on a case-by-case basis. The following two sections will discuss the case description and the construction of the scaling matrix $\boldsymbol{B}$ for this case study.

### 6.4.1 Case Description

This case, identical to the one presented in the section 3.4.1, involves a stratified airflow with strong buoyance in a closed space, by adding a heat source to the obstacle and controlling the temperature of the walls (Wang and Chen 2009). Figure 3-10 presents the velocity magnitude and temperature contours at the plane sliced at Y=1.22 m, as shown in the. Both contours show that the strong buoyance airflow has a stratified distribution in velocity and temperature. In terms of velocity, the high jet is formed near the ceiling and circulation was shown between the box and room. Regarding the temperature, the plume clearly occurred above the heated box.

### 6.4.2 Construction of the Scaling Matrix B

For the non-isothermal flow with stratified distribution defined above, we defined two sets of inputs: temperature (20 - 30 $^{o}C$) and inlet velocity magnitude (1 - 2 $m/s$) as shown in Table 6-1. The corresponding outputs can be normalized as:

$$V_{norm} = \frac{|V|}{|V|_{inlet}} \quad \textbf{(40)}$$

$$T_{norm} = \frac{T - T_{min}}{T_{max} - T_{min}} \quad (41)$$

where $|V|_{inlet}$ is the velocity magnitude at the inlet; $T_{min}$ and $T_{max}$ are the lowest temperature and highest temperature in the inputs, respectively. To get the highest ratio, we divided the highest $T_{norm}$ by the lowest $V_{norm}$. The results were summarized in Table 6-1. As the ratios are in order of 10 for all the tests, we set the diagonal entries of ***B*** corresponding to velocity magnitude outputs as 10.

*Table 6-1 Inputs and normalized outputs of the sensitivity study*

| **Scenario #** | **Input Variables** | | | | **Normalized Output Variables** | | | | ***Highest Ratio*** |
|---|---|---|---|---|---|---|---|---|---|
| | $T_{floor}$ (℃) | $T_{wall}$ (℃) | $T_{inlet}$ (℃) | $\|V\|_{inlet}$ (*m/s*) | $T_{occ}$ | $\|V\|_{occ}$ | $T_{sensor}$ | $\|V\|_{sensor}$ | |
| **1** | 25 | 25 | 20 | 1 | 0.984 | 0.1034 | 1.184 | 0.0799 | 14.8 |
| **2** | 30 | 25 | 20 | 1 | 1.136 | 0.1285 | 1.308 | 0.0634 | 20.6 |
| **3** | 25 | 30 | 20 | 1 | 0.756 | 0.1293 | 0.875 | 0.0645 | 13.6 |
| **4** | 25 | 25 | 20 | 2 | 0.946 | 0.1971 | 1.372 | 0.0174 | 78.9 |
| **5** | 30 | 30 | 25 | 2 | 0.796 | 0.1993 | 1.048 | 0.0159 | 65.9 |

## 6.5 Simulation Results

To gain a comprehensive understanding of ISAT-FFD for indoor airflow simulation, we studied its performance in both the training and evaluation stages. The results and findings are presented in the following two sections.

### 6.5.1 Performance of ISAT-FFD in Training Stage

This section accounts the performance of the ISAT-FFD model at the training stage. It focuses on the cost of training (measured by training time) associated with four key factors: training methods, error tolerances, numbers of inputs, and numbers of outputs.

#### 6.5.1.1 Training Methods

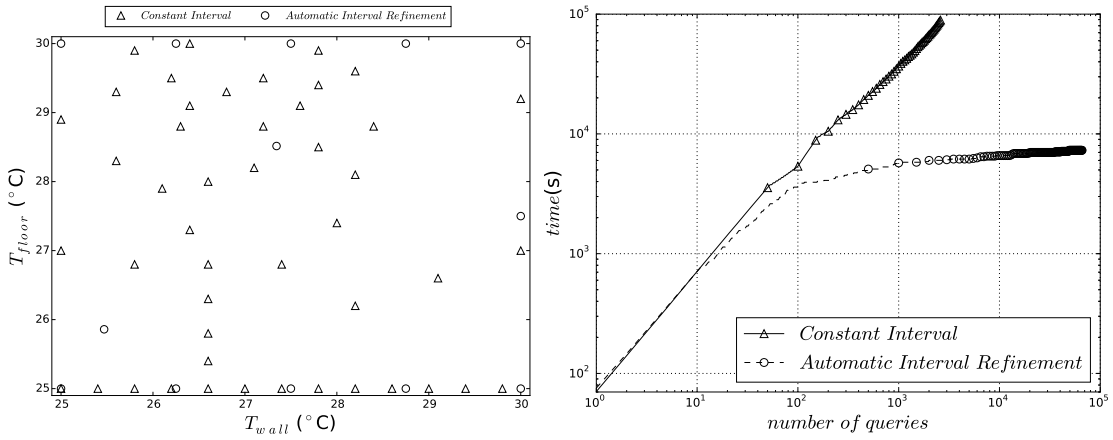
This test compares the training time by using the two training methods proposed in Section 3.3. We used two inputs consisting of $T_{wall}$ and $T_{floor}$ and one output of $T_{occ}$. In the training domain both temperatures ranged from 25 to 30 $^{o}C$. The error tolerance for the output was set to 0.4. When using the *constant interval* method, the increment in each dimension was defined to be 0.1 $K$.

By using the *constant interval* method, the ISAT-FFD evaluated 2,601 queries, which led to 1,424 *retrieve* actions, 1,130 *grow* actions, and 47 *add* actions, respectively. Even with a powerful GPU, the ISAT-FFD using the *constant interval* method took 24.8

hours to complete the training. On the contrary, the ISAT-FFD using the *automatic range refinement* method assessed 66,049 queries, which is approximately 25 times more than the ISAT-FFD with *constant interval* method did. However, about 99.9 % of the total queries were resolved by *retrieve* actions and only 0.01% of the queries resulted in *add* (13) and *grow* (59) actions that required calling the FFD simulation. It is worth mentioning that the ISAT will perform accuracy tests randomly based on the frequency of *grow* actions by calling the FFD. The accuracy tests may shrink the EOA to ensure the accuracy of the *retrieve* action. Simulations using the *constant interval* method triggered 103 accuracy tests and the number reduced to 31 when using the *automatic range refinement* method. As a result, the ISAT-FFD trained with the *automatic range refinement* method only took roughly 8% of the time when compared to the ISAT-FFD with the *constant interval* method.

To explain the training time difference between the two training methods, we plotted the *add* actions performed by both approaches (Figure 6-5a). As expected, the *automatic range refinement* method led to a more scattered distribution of *add* actions in the training domain than the *constant interval* method. Because each *add* action resulted in a new record in the ISAT table, fewer *add* actions mean that the ISAT needed fewer records to cover the training domain. Figure 6-5(b) compares the trajectories of training time used by the two methods. Since the time required by *retrieve* actions are negligible compared to that for *add* and *grow* actions, the time increase pattern actually reflects the

distribution of *add* and *grow* actions during the training. The increasing rate of the trajectory in the simulation using the *constant interval* method was approximately constant, which suggests the uniform distribution of *add* and *grow* actions over the whole evaluation. The training time by the *automatic range refinement* method increased fast for the first 100 queries, and then the rate of increase flattened afterwards. This indicates that the *add* and *grow* actions mainly happened at the beginning of the training, and the ISAT-FFD model then can answer the remaining queries mainly using the *retrieve* actions.




a) b)

*Figure 6-5 Comparison of the add actions (a) and training time (b) of the ISAT-FFD using different training methods*

### 6.5.1.2 Error Tolerances

Equation (38) explains the impact of the error tolerance on the shape of the EOA. Namely, the larger the error tolerance, the larger the size of the EOA and the less time the training process takes. To set up a numerical test, two inputs consisting of $T_{wall}$ and $T_{floor}$ and one output of $T_{occ}$ were used. In the training domain both temperatures ranged from 25 to 30 $^{o}C$. The *automatic range refinement* method was used and the error tolerance varied from 0.2 to 1.0 with increments of 0.2.

The simulation results (Figure 6-6) show that with the lowest error tolerance (0.2) the total number of queries was 263,169, out of which there were 262,813 (99.9%) *retrieve*, 304 *grow*, and 52 *add* actions. The total training time was approximately 8.2 hours. By comparison, using a larger error tolerance (1.0) the number of queries was only 4,225, of which 99.3% are *retrieve* actions. As a result, it only took 0.64 hours to complete the training. When the error tolerance increased from 0.2 to 0.4, the training time dropped significantly from 8.2 hours to 2.1 hours. Subsequently, the declining rate of training time flattened. The relationship between the error tolerance and training time can be represented by a regression curve which fits into the power function with an $R^2$ larger than 99%. It is worth to note that the coefficients of the power function will likely vary from case to case.

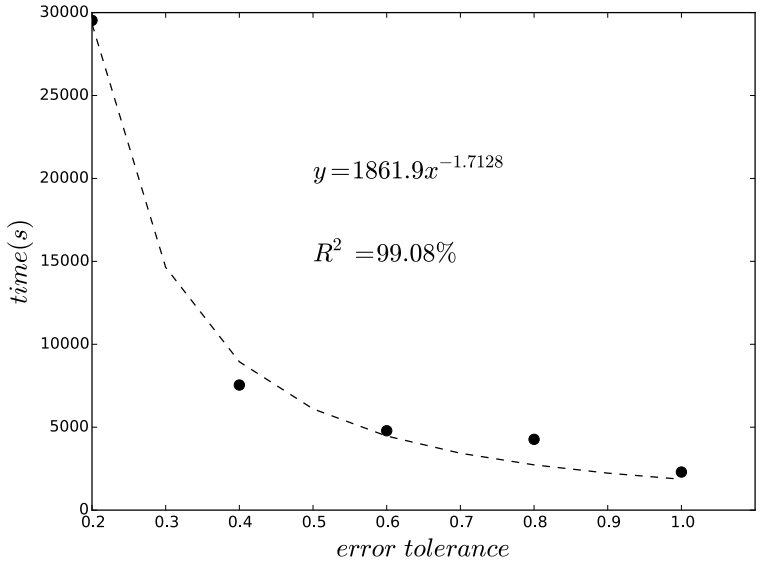



*Figure 6-6 Relationship between error tolerance and training time*

### 6.5.1.3 Number of Inputs

The number of inputs affects the training time as it determines the dimensions of the training domain. Also, the indoor airflow is usually sensitive to the boundary conditions, which in this study are the inputs of the ISAT-FFD. Changing the number of inputs will impact the mapping gradient matrix $\boldsymbol{A}$ in Equation (32) and hence change the shape of the EOA. To show the potential application of ISAT-FFD in real control purpose, we studied nine scenarios with the number of inputs varying from 1 to 9. The detailed information of inputs is shown in Table 6-2. We set $T_{occ}$ as the output and the error tolerance as 1.0, which allows a prediction error of 1.0 ℃. The *automatic range*

*refinement* method was also used in this scenario. To reduce the computing time for scenarios with large number of inputs, we set the $\eta$ in Equation (39) as 0.005.

*Table 6-2 Inputs and corresponding training domain in different scenarios*

| Scenario # | Input $\boldsymbol{x}$ | Training Domain |
|---|---|---|
| 1 | $[T_{wall}]$ | $T_{wall} \in [25.0,30.0]$ |
| 2 | $[T_{wall}, T_{floor}]$ | $T_{floor} \in [25.0, 30.0]$ and the rest is the same as Scenario #1 |
| 3 | $[T_{wall}, T_{floor}, T_{inlet}]$ | $T_{inlet} \in [20.0,25.0]$ and the rest is the same as Scenario #2 |
| 4 | $[T_{wall}, T_{floor}, T_{inlet}, \lvert V\rvert_{inlet}]$ | $\lvert V\rvert_{inlet} \in [1.0,2.0]$ and the rest is the same as Scenario #3 |
| 5 | $\begin{bmatrix} T_{wall}, T_{floor}, T_{Top}, \\ T_{inlet}, \lvert V\rvert_{inlet} \end{bmatrix}$ | $T_{top} \in [25.0, 30.0]$ and the rest is the same as Scenario #4 |
| 6 | $\begin{bmatrix} T_{wall}, T_{floor}, T_{top}, T_{east}, \\ T_{inlet}, \lvert V\rvert_{inlet} \end{bmatrix}$ | $T_{east} \in [25.0, 30.0]$ and the rest is the same as Scenario #5 |
| 7 | $\begin{bmatrix} T_{wall}, T_{floor}, T_{top}, T_{east}, \\ T_{west}, T_{inlet}, \lvert V\rvert_{inlet} \end{bmatrix}$ | $T_{west} \in [25.0, 30.0]$ and the rest is the same as Scenario #6 |
| 8 | $\begin{bmatrix} T_{north}, T_{south}, T_{floor}, T_{top}, \\ T_{east}, T_{west}, T_{inlet}, \lvert V\rvert_{inlet} \end{bmatrix}$ | $T_{north} \in [25.0, 30.0]$ and the rest is the same as Scenario #7 |
| 9 | $\begin{bmatrix} T_{block}, T_{north}, T_{south}, T_{floor}, \\ T_{top}, T_{east}, T_{west}, \quad T_{inlet}, \lvert V\rvert_{inlet} \end{bmatrix}$ | $T_{block} \in [30.0, 35.0]$ and the rest is the same as Scenario #8 |

As Shown in Figure 6-7, when one input was used (Scenario #1), the ISAT-FFD needed 9 queries to complete the training in 0.1 hours. Among them, only 2 *grow* and 3 *add* actions were needed. If the number of inputs is 4 (Scenario #4), the ISAT-FFD needed 655 queries, including 9 *add* and 95 *grow* actions. It took 2.5 hours to complete the training. Further increasing the number of inputs to 9 (Scenario #9), we found that number of queries went up to around two million consisting of 493 *grow* and 42 *add* actions. The training process took 12 hours. Figure 6-7 shows that power function can depict the relationship between the number of inputs and training time. It is worth to mention that the coefficients of the power function will likely vary for different cases.

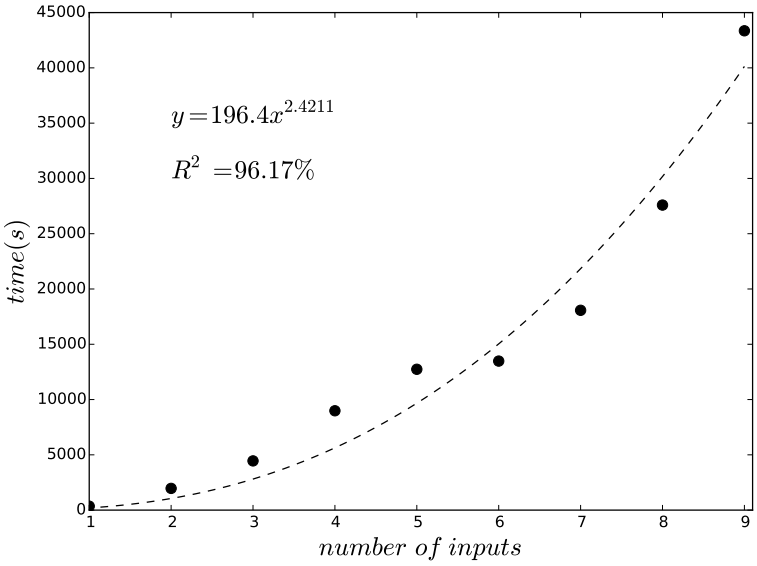



*Figure 6-7 the relationship between number of inputs and ISAT training time*

#### 6.5.1.4 Number of Outputs

The mapping gradient matrix **A** defined in Equation (32) is also affected by the number of outputs. Thus, the training time of the ISAT-FFD is furthermore a function of the number of outputs. Here we defined four scenarios with the number of outputs from one to four (Table 6-3). When the number of output is larger than two, we used the scaling matrix **B** to make the interpolation error of each output comparable. After applying the scaling matrix **B**, we intended to set the single error tolerance $\varepsilon_{\text{single}}$ for each output as 0.4. As introduced before, ISAT allows only one total error tolerance $\varepsilon_{\text{total}}$, which is calculated using the following definition:

$$\varepsilon_{total} = \sqrt{\sum_{1}^{n} \varepsilon_{single}^2} = \sqrt{n}\, \varepsilon_{single} \tag{42}$$

where $\boldsymbol{\varepsilon_{\text{single}}}$ is the error tolerance for the single output and **n** is the number of outputs.

*Table 6-3 Selection of outputs, scaling matrix and total error tolerance for different scenarios*

| Scenario # | Output $y$ | Scaling Matrix $B$ | $\varepsilon_{total}$ | Note |
|---|---|---|---|---|
| 1 | $[T_{occ}]$ | No | 0.4 | $T_{occ} = \frac{1}{N}\sum_{Z\leq 1.22} T_i$ |
| 2 | $[T_{occ}, \lvert V\rvert_{occ}]$ | $\begin{bmatrix}1 & 0\\ 0 & 10\end{bmatrix}$ | 0.5657 | $\lvert V\rvert_{occ} = \frac{1}{N}\sum_{Z\leq 1.22}\sqrt{{u_i}^2+{v_i}^2+{w_i}^2}$ |
| 3 | $[T_{occ}, \lvert V\rvert_{occ,}\, T_{sensor}]$ | $\begin{bmatrix}1 & 0 & 0\\ 0 & 10 & 0\\ 0 & 0 & 1\end{bmatrix}$ | 0.6928 | Sensor location: (1.22, 1.22, 1.24) m |
| 4 | $[T_{occ}, \lvert V\rvert_{occ}, T_{sensor}, \lvert V\rvert_{sensor}]$ | $\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 10 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 10\end{bmatrix}$ | 0.8 | Sensor location: (1.22, 1.22, 1.24) m |

In this case, we also used two inputs consisting of $T_{wall}$ and $T_{floor}$ and both temperatures ranged from 25 to 30 $^{o}C$. We also used *automatic range refinement* method during the training.

Simulation results showed that when the number of outputs was as small as one (Scenario #1), the training can be completed with 66,049 queries within about 2 hours. While increasing the number of outputs to four (Scenario #4), the required queries raised roughly 64 times to 4,198,401. The number of *grow* and *add* actions increased about twice the amount to 118 and 25, respectively. Correspondingly, the training time also increased about twice to 6.74 hours. By further conducting the nonlinear regression

between the training time and the number of outputs, as shown in Figure 6-8, it was found that the time growth along the number of outputs was fitted into a power function. Again, it is important to keep in mind that the coefficient in the regression model is not universal for all cases.

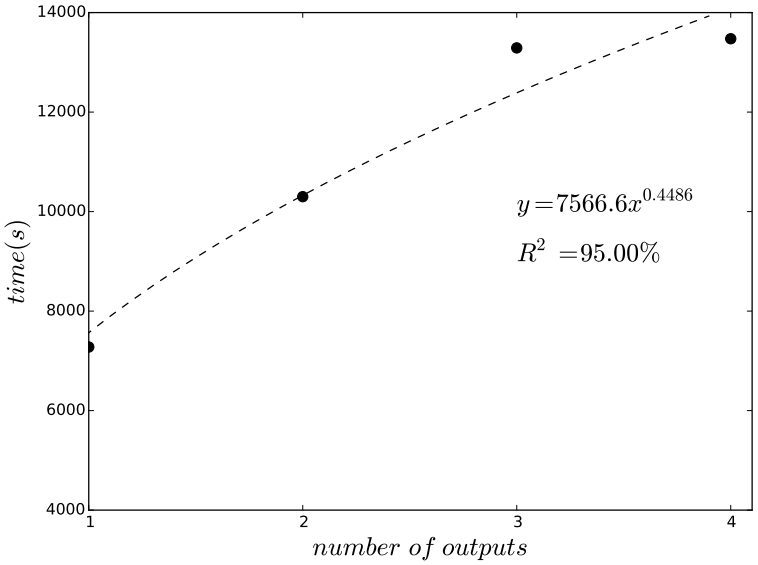



*Figure 6-8 Relationship between the number of outputs and ISAT-FFD training time*

### 6.5.2 Performance of ISAT-FFD in Evaluation Stage

This section evaluates the performance of a trained ISAT-FFD model by testing it with different new inputs. We first evaluated how it performed under different sizes of an evaluation domain. We then tested how the error tolerance in the training affected the aggregated local errors during the evaluation. The ISAT-FFD was first trained using four

inputs and four outputs. The inputs and training domain are defined by Scenario 4 in Table 6-2. The outputs and error tolerance are defined by Scenario 4 in Table 6-3. Similarly, the *automatic range refinement* method was used to train the ISAT-FFD. Due to the high sensitivity and turbulence of the flow, we found that it was not possible to complete the training for the entire training domain within an acceptable time. Thus, we set the $\eta$ in Equation (39) to 0.0005, which will deem the training completed if five or less *add* or *grow* actions are generated per 10,000 queries. Even with this setting, the training process took 23.2 hours to evaluate the total 1,185,921 queries with 122 *add* and 850 *grow* actions.

#### 6.5.2.1 Performance of ISAT under Different Evaluation Size

The variation of the evaluation domain is defined as:

$$T_{other} \in [25.0 - a, \quad 30.0 + a] \tag{43}$$

$$T_{floor} \in [25.0 - a, \quad 30.0 + a] \tag{44}$$

$$T_{inlet} \in [20.0 - a, \quad 25.0 + a] \tag{45}$$

$$|V|_{inlet} \in [1.00 - b, 2.00 + b] \tag{46}$$

where $a$ and $b$ are the constants defined in the Table 6-4. We used a normal distribution to pick the value of query points in each dimension. For instance, Figure 6-9 shows the

distribution of probability for one of the inputs, $T_{wall}$. This will cause the ratio of queries residing in the training domain be higher than the ratio of evaluation domains covered by the training domain. For instance, in Scenario 1 of Table 6-4, only 52% of evaluation domain is covered by the training domain, but 90.7% of the evaluating queries fall into the training domain. We believe that this is closer to a real situation than the data generated by a uniform distribution. A set of 108 queries were generated based on that methodology.

*Table 6-4 Generation of different evaluation domains*

| **Scenario #** | $a$ | $b$ | $S_{training}/S_{evaluation}$ | **Queries Within the Training Domain (Percentage)** |
|---|---|---|---|---|
| 1 | 0.5 | 0.05 | 52% | 98 (90.7%) |
| *2* | 1.0 | 0.10 | 30% | 87 (80.6%) |
| 3 | 1.5 | 0.15 | 19% | 70 (64.8%) |
| 4 | 2.0 | 0.20 | 12% | 55(50.9%) |
| 5 | 2.5 | 0.25 | 8% | 37(34.3%) |

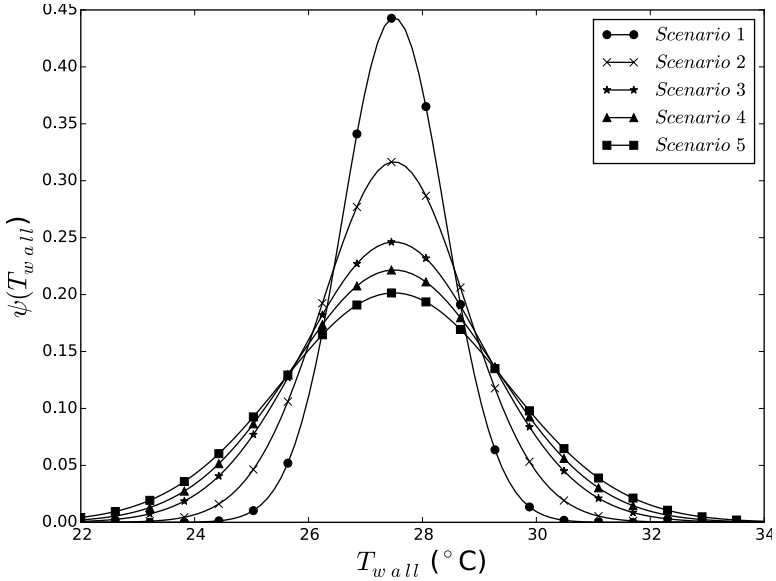



*Figure 6-9 Normal distribution used to randomly generate query points for temperatures of other walls.*

To better visualize a five-dimension data points on two-dimension plotting, we used the oval shape to represent four inputs and surface color of oval shape to show the *retrieve* error for the inputs. The coordinates of center point of ellipsoid represent the $T_{wall}$ and $T_{floor}$. The red dash line represents the training domain and the blue solid line represents the evaluation domain. The remaining two inputs, consisting of $T_{\boldsymbol{inlet}}$ and $V_{\boldsymbol{inlet}}$, are presented by half-length and half-height of the ellipsoid after normalization:

$$T^* = \frac{T_{inlet} - (20.0 - a)}{(25.0 + a) - (20.0 - a)} = \frac{T_{inlet} - 20.0 + a}{5 + 2a} \quad \textbf{(47)}$$

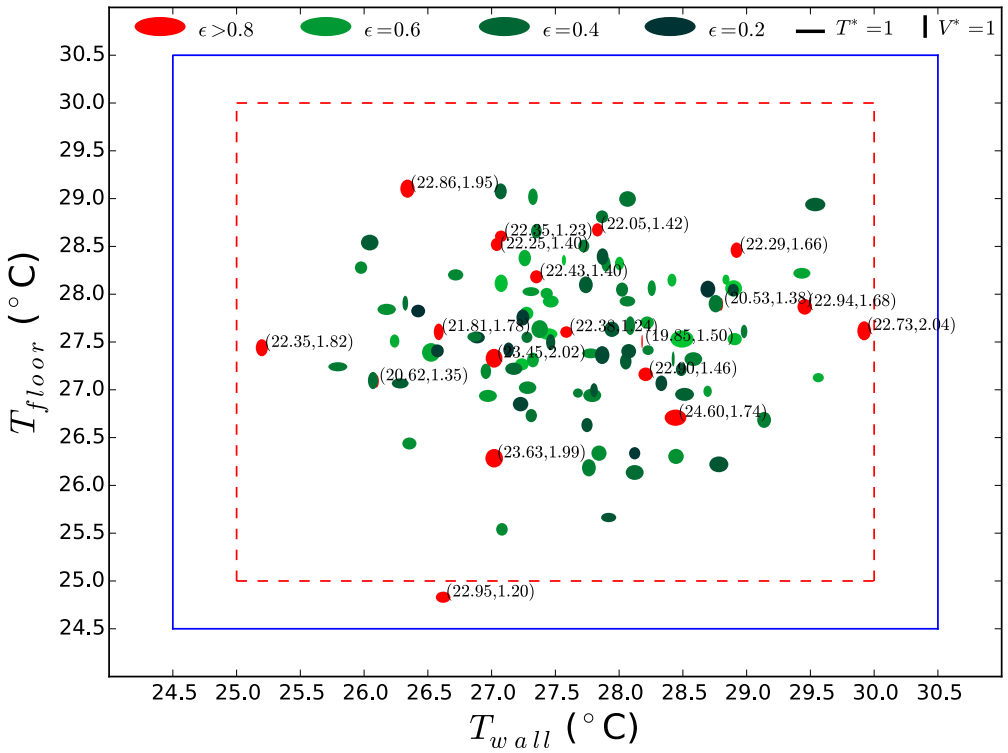
$$V^* = \frac{|V|_{inlet}}{2.00 + b} \quad \textbf{(48)}$$

where a and b are provided in Table 6-4 and varies scenario by scenario.

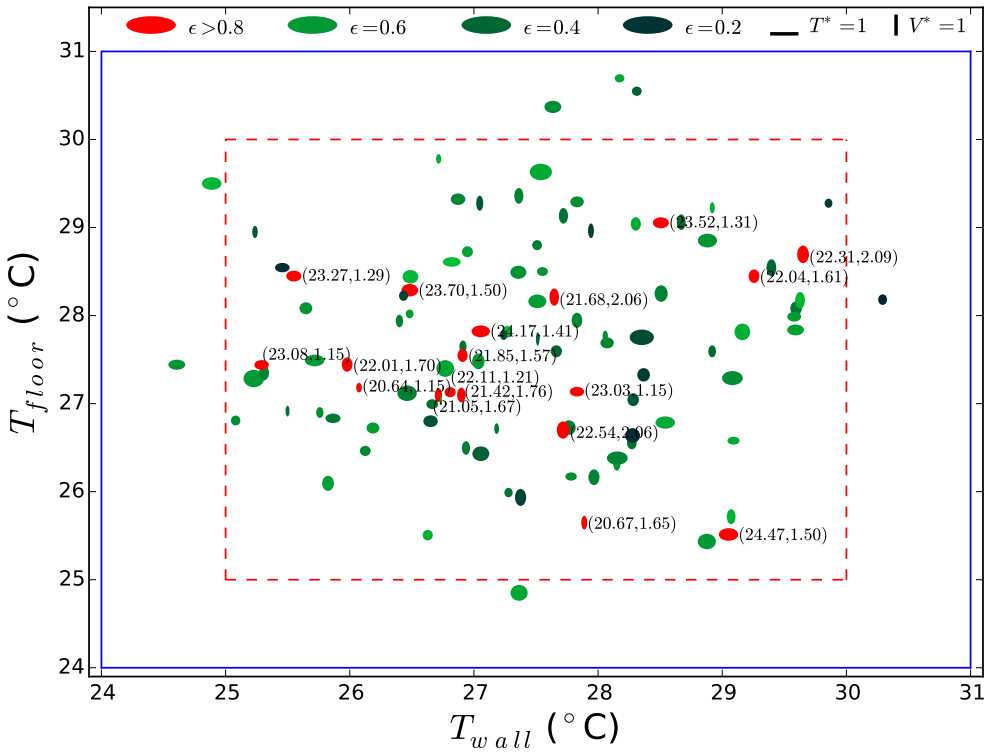
For scenario #1, because the generated query points were largely residing inside the training domain (90%), all the queries were resolved by *retrieve* actions (Figure 6-10a). Although there were 10 points located outside the training domain, ISAT still predicted the outputs using extrapolation. As a result, the evaluation took only 0.001s, which is negligible.

To evaluate the accuracy of the ISAT-FFD model, we compared the outputs of the ISAT-FFD with the standalone FFD simulation outputs. Here we consider the FFD simulation outputs as a reference. Since the *add* and *grow* actions actually returned the results of a FFD simulation, the only source of the error was from the *retrieve* action which approximated the FFD outputs using a linear interpolation. As coloured in red, 19 points were retrieved with actual error larger than the error tolerance of 0.8. The maximum actual error is 1.54, which is about 1.92 times larger than the error tolerance.

The results of Scenario #2 were similar to Scenario #1, as most of the queries (80%) were located within the training domain. There were 21 evaluation query points out of the domain. Again, ISAT can handle all the queries using *retrieve* action, as shown in Figure 6-10b.




a)




b)

*Figure 6-10 Distribution of evaluation points for retrieve actions (a) Scenario 1 (b) Scenario 2*

When the training domain coverage percentage dropped to 19% in Scenario #3 and 38% of the query points were outside the training domain, about 92% of query points can be handled by *retrieve* actions as shown in Figure 6-11a. Among them, 20 points were retrieved with actual error larger than the error tolerance of 0.8. The maximum

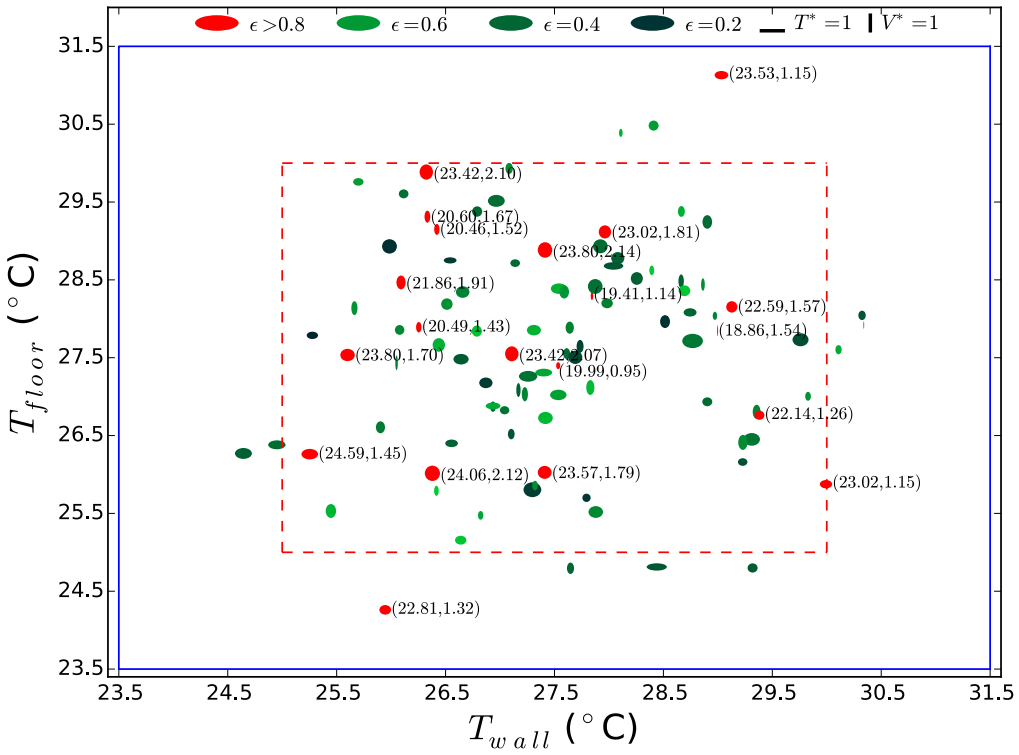
actual error is 1.62, which is about 2.02 times larger than the error tolerance. Moreover, there are 1 *grow* and 5 *add* actions, which were all located outside the training domain, as shown in Figure 6-11b.

Similarly, further decreasing the coverage as low as 12% and 8% caused that 49.1% and 65.7% of evaluation points were outside the training domain, therefore fewer query points are evaluated by *retrieve* actions (Figure 6-11c and Figure 6-11e) and more needed either *grow* or *add* action (Figure 6-11d and Figure 6-11f).

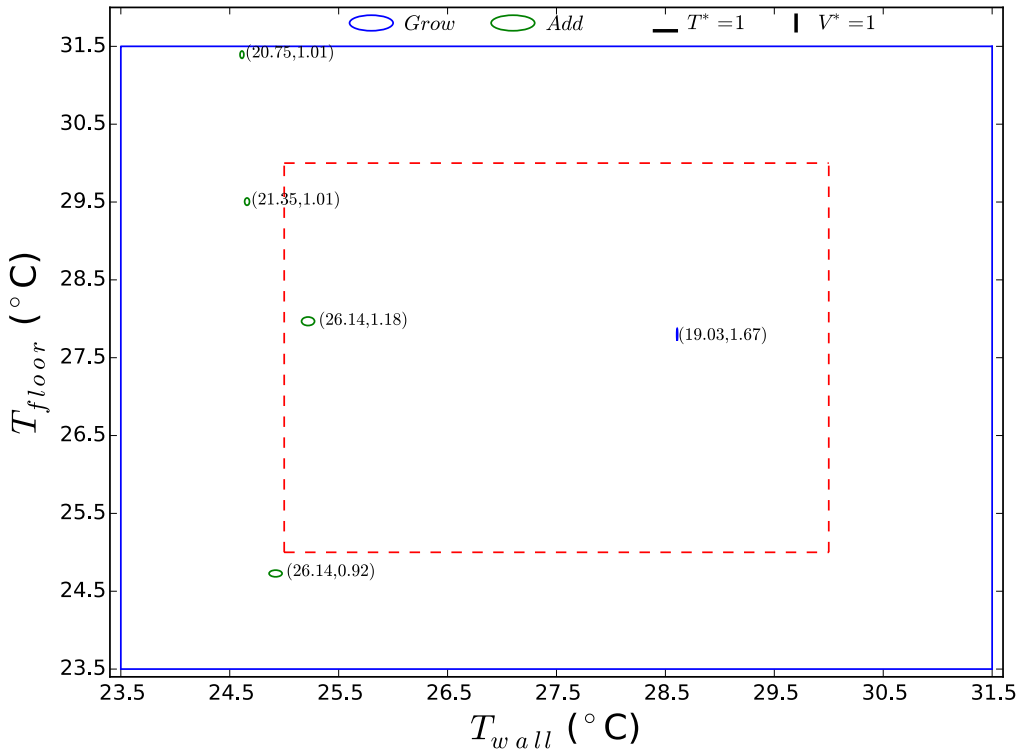
We further define speedups $n$ to quantify how fast the ISAT-FFD can be:

$$n = \frac{T_{DI}}{T_{ISAT}} \tag{49}$$

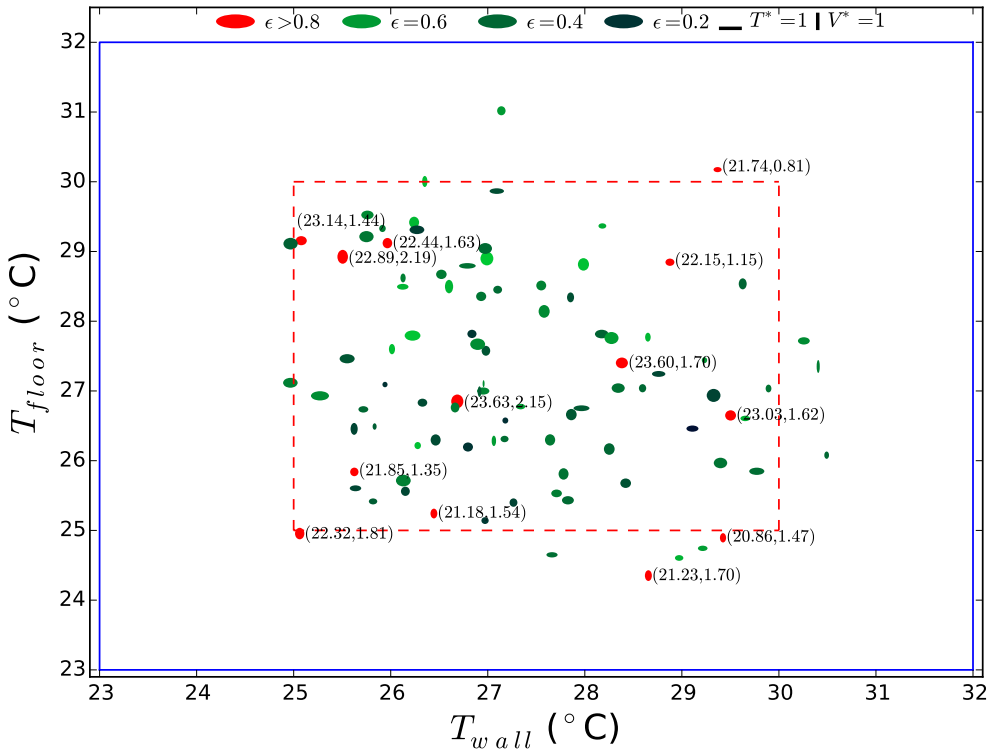
where $T_{DI}$ is the total estimated time for all queries done by directly launching a FFD simulation; $T_{ISAT}$ is the time cost for the all queries by the ISAT-FFD. We found that if the training domain covers over 30% of the evaluation domain and queries within training domain exceed 80.6%, the speedup can be as high as 1.5 million times. Even when the coverage percentage was as low as 8%, the speedup can be 5.9 times.



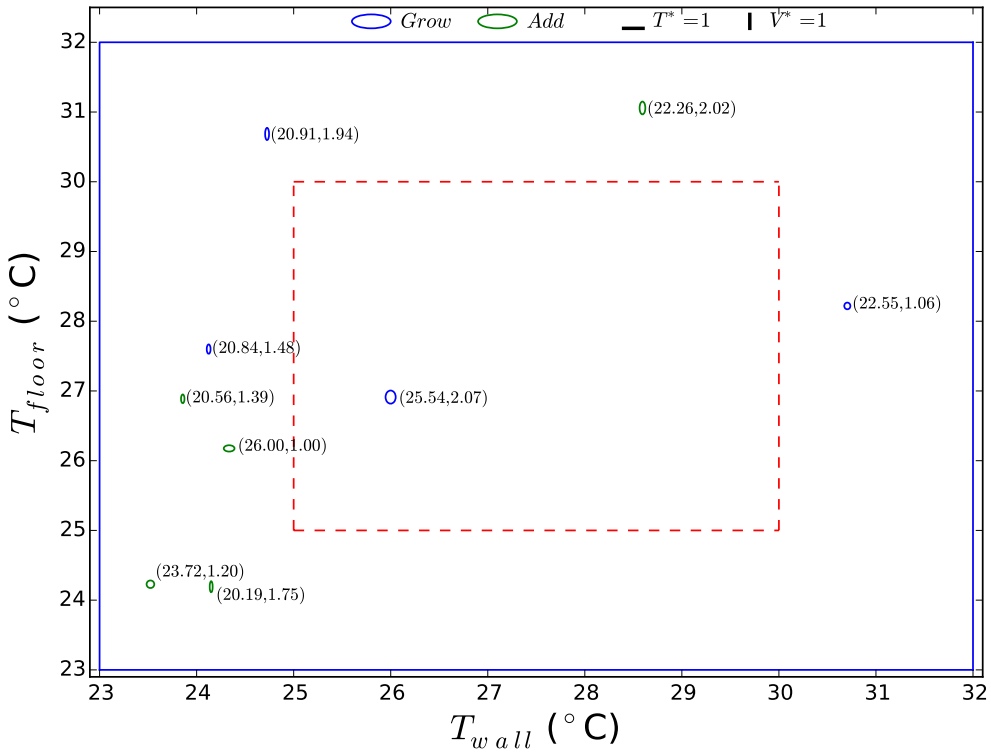

a)



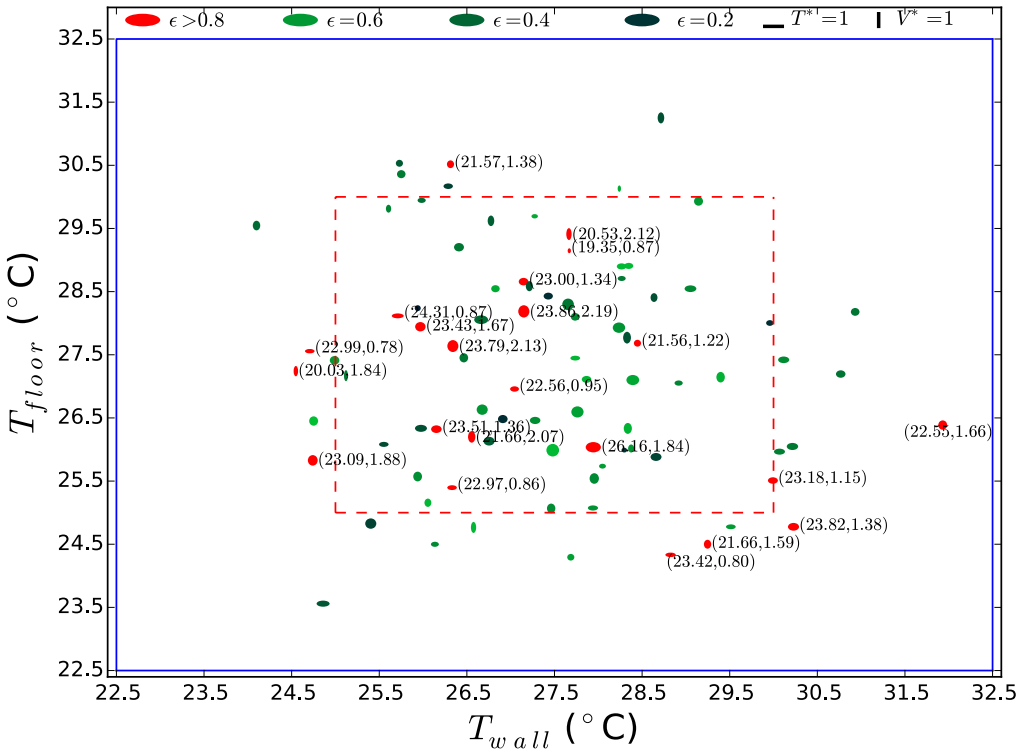

b)



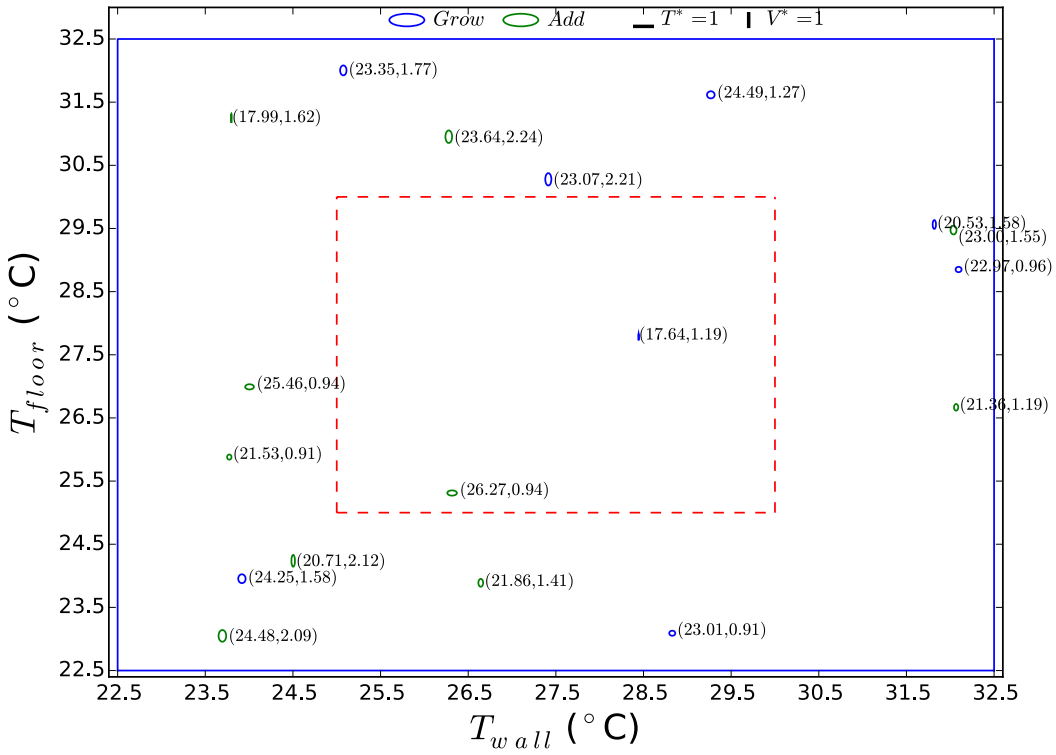

c)




d)




e)




f)

*Figure 6-11 Distribution of evaluation points for scenario 3-5. (a) Retrieve points in scenario 3; (b) Add and grow points in Scenario 3; (c) Retrieve points in Scenario 4; (d) Add and grow points in Scenario 4; (e) Retrieve points in Scenario 5; (f) Add and grow points in Scenario 5;*

### 6.5.2.2 Performance of ISAT-FFD under different error tolerance

The error tolerance $\varepsilon_{total}$ defined in Equation (42) was reported to have an impact on the aggregated errors during the evaluation (Pope 1997). To study their relationship in the context of airflow simulations, we performed the ISAT-FFD simulation with the error tolerance ranging from 0.8 to 2.4 with increments of 0.4. As a

result, the single error tolerance for each output increased from 0.4 (0.4 *K* for temperature and 0.04 *m/s* for velocity magnitude) to 1.2 (1.2 *K* for temperature and 0.12 *m/s* for velocity magnitude). The number of inputs and outputs and training method are identical to those in Section 6.2. The evaluation domain is defined as Scenario 1 in Table 6-4.

Using an error tolerance of 0.8, the aggregated errors during the evaluation were 63.735 for the 108 queries. The error for one *retrieve* during the evaluation on average was 0.59, which indicated that the overall performance of the ISAT-FFD was accurate. With the highest error tolerance being that of 2.8 in training, the total aggregated errors were 110.473 for the 108 queries. On average each *retrieve* action contributed 1.02 to the accumulated error, which is about 0.37 times of the error tolerance. Figure 6-12 shows the regression curve between the error tolerance and mean retrieve error in the evaluation. The solid line represents the error tolerance in the training. The regression curve shows that when the error tolerance increased, the mean retrieve error in evaluation also grew to some extent, yet it was always below the error tolerance of the training. It is worth to point out that the coefficient in the regression equation only pertains to the specific case study presented in this paper.

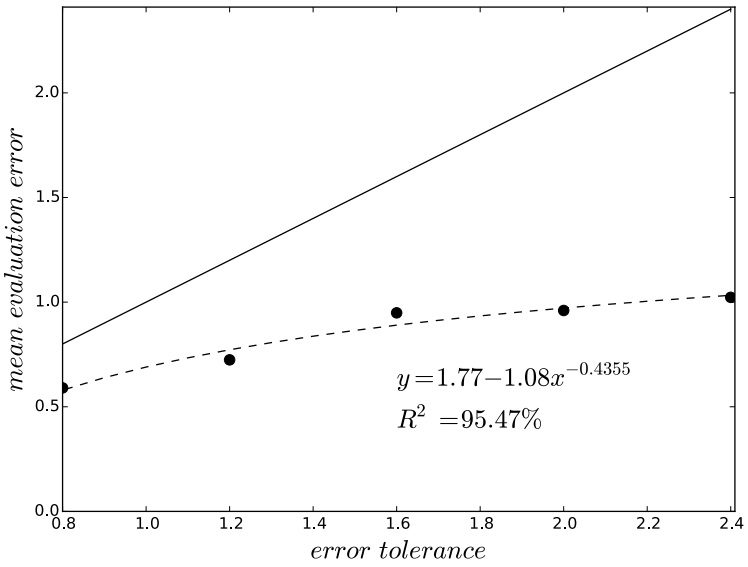



*Figure 6-12 the relationship between accumulated error and error tolerance settings*

## 6.6 Conclusion

In this chapter, we explored the feasibility of the ISAT in predicting key information for indoor airflows using FFD for airflow simulation as an example. First, the investigation showed that the *automatic interval refinement* method is an efficient approach for training ISAT. Second, the error tolerance, the number of inputs, as well as the number of outputs, can significantly impact the training time. Exploiting a trained ISAT, we found that it performed differently depending on various sizes of the training domain. Lastly, we identified that the error tolerance during training could affect the mean retrieve error during an evaluation.

To conclude, a well-trained ISAT table can provide timely and reasonably accurate predictions of indoor airflows simulations. Before training the ISAT, users need to validate the high order models (e.g. CFD or FFD) using experimental data to ensure the accuracy of flow field prediction, from which the training data is extracted. Afterward, the validated high order models can generate the reliable training data to train the ISAT for predicting the key information. In addition, before applying ISAT into a real project, a sensitivity study needs to be done to select as few inputs as possible to reduce training time. Also, it is critical to determine an appropriate error tolerance for the ISAT training since trade-offs exist between accuracy, performance, and training time. Finally, it is beneficial to make the trained domain cover the evaluation domain as much as possible to avoid the need of *grow* or *add* actions.

# Chapter 7 Conclusion and Future Work

## 7.1 Conclusion

In this dissertation, a coupled simulation model between Modelica and CFD is proposed, designed, and developed within the context of Modelica *Buildings* library, to carry out simulations of the interaction between HVAC systems and non-uniform airflows. As for a building-scale application, the amelioration of the coupled simulation model is proposed, by adding the multizone models. CFD is used for few rooms with non-uniform air distribution while the rest by the multizone model. Noting that the coupled simulation model is not restricted to one specific CFD model, the work reported in this dissertation uses FFD model, which can be significantly faster than CFD by solving the same set of governing equations as CFD does, as an example.

As the coupled simulation models are proposed for HVAC design and operation optimization that require hundreds of iterations of simulation, this research further explores multiple ways to accelerate non-uniform airflow simulations, which usually dominate in computation time cost over HVAC simulations. First, the parallelization of FFD simulation using OpenCL is developed and the performance of accuracy and speed is systematically evaluated. Second, a ROM called ISAT trained by the outputs from parallel FFD simulations is proposed and its performance of training and predicting is thoroughly evaluated.

We can draw the conclusion of the research in this dissertation as follows:

1. The coupled simulation model, publicly released in Modelica *Buildings* library, can realistically represent the dynamics of HVAC system, control, and building envelope heat transfer, with full-scale modeling of the non-uniform airflows in a single zone. The model is shown to be able to achieve a faster-than-real-time speed.

2. The ameliorated model that couples FFD with multizone models in the Modelica *Buildings* library can study the dynamic response of HVAC systems in a building consisting of multi zones with stratified airflow distributions. The model is validated by simulating a non-uniform momentum distribution and a non-uniform temperature distribution.

3. With the capability of dynamic simulation of stratified air distributions and HVAC systems, the coupled simulation models can provide a more realistic environment for control engineers and researchers to study the control of stratified ventilation systems.

4. The parallelization of FFD simulations in OpenCL can achieve up to 1139 times speedup by running on the latest GPU. Minor differences are found in the results from different GPUs mounted on different operating

systems, as different manufacturers have their own interpretations of the *IEEE-754 2008 standard*. Applying it to replace the sequential FFD model in the coupled simulation models, we believe that the design optimization of HVAC systems can be promisingly performed.

5. A well-trained ISAT can provide timely and reasonably accurate predictions of indoor airflows simulations. The *automatic interval refinement* method proposed in this dissertation is an efficient approach for training ISAT. It is found that the error tolerance, the number of inputs, as well as the number of outputs, can significantly impact the training time. As ISAT can answer query points both inside and close to training domain using *retrieve* actions within a time less than 0.001s for each query, we believe that by replacing the FFD with ISAT in the coupled simulation model the model-based optimization in operation phase can be realized.

## 7.2 Future Work

Future work based on this dissertation can include:

1. Applications involving the contaminant distributions can be carried out by using the coupled simulation models. The case studies performed in this dissertation only deal with the thermal environment and its control.

2. For the parallelization of FFD simulations using OpenCL, the optimized compiling setting can be studied to further increase the speedup times. The compiling of the OpenCL codes in this study, regardless of operating systems, is done using the default settings.

3. More efficient training methods can be proposed to further improve the training efficiency of the ISAT.

4. Identify the application cases to perform design optimizations using the coupled simulation models. The space in the case can be either single zone or multizone zones. A typical example can be determining the best locations for the thermostat in an office room to achieve an optimal solution for thermal comfort and energy efficiency.

5. Identify the application cases to perform model-based control in operation phase using the coupled simulation models. A typical example can be reducing the cooling energy in a data center by using model-based control. Another example can be finding the best strategy to remove the hazardous gas in a building after the accidental spill

# WORKS CITED


Administration, U. S. F. "Fire Statistics." https://www.usfa.fema.gov/data/statistics/.

Afram, A., and F. Janabi-Sharifi. 2014. Theory and Applications of HVAC Control Systems–a Review of Model Predictive Control (Mpc). *Building and Environment,* 72:343-55.

Ahuja, S., A. Surana, and E. Cliff. 2011. Reduced-Order Models for Control of Stratified Flows in Buildings. *Proceedings of the Proceedings of the 2011 American Control Conference*.

AMD. "Firepro V4900." https://www.amd.com/Documents/AMDFirePro_FamilyBrochure.pdf.

AMD. http://developer.amd.com/tools-and-sdks/opencl-zone/amd-accelerated-parallel-processing-app-sdk/.

AMD. "Firepro W8100." http://www.amd.com/en-us/products/graphics/workstation/firepro-3d/8100.

Ansuini, R., R. Larghetti, M. Vaccarini, A. Carbonari, A. Giretti, S. Ruffini, H. Guo, and S. L. Lau. 2012. Hybrid Modeling for Energy Saving in Subway Stations. *Proceedings of the the First Building Simulation and Optimization Conference*, Loughborough, UK.

Ascione, F., L. Bellia, and A. Capozzoli. 2012. A Coupled Numerical Approach on Museum Air Conditioning: Energy and Fluid-Dynamic Analysis. *Applied Energy*.

Axley, J. 2007. Multizone Airflow Modeling in Buildings: History and Theory. *HVAC&R Research,* 13 (6):907-28.

Beausoleil-Morrison, I. 2000. The Adaptive Coupling of Heat and Air Flow Modeling within Dynamic Whole-Building Simulation. *Proceedings of the 7th International IBPSA Conference*, Rio de Janeiro, Brazil, August 13-15, 2001.

Birdsall, B. E., W. F. Buhl, R. B. Curtis, A. E. Erdem, J. H. Eto, J. J. Hirsch, K. H. Olson, and F. C. Winkelmann. 1985. "The Doe-2 Computer Program for Thermal Simulation of Buildings." In *Energy Sources: Conservation and Renewables*, edited by David Hafmeister. American Institute of Physics.

Blochwitz, T., M. Otter, M. Arnold, C. Bausch, C. Clauß, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, and T. Neidhold. 2011. The Functional Mockup Interface for Tool Independent Exchange of Simulation Models. *Proceedings of the Modelica'2011 Conference, March*.

Celebi, M. E., F. Celiker, and H. A. Kingravi. 2011. On Euclidean Norm Approximations. *Pattern Recognition,* 44 (2):278-83.

Chen, Q. 1995. Comparison of Different K-E Models for Indoor Air-Flow Computations. *Numerical Heat Transfer Part B: Fundamentals,* 28 (3):353-69.

Chen, Q., and J. v. d. Kooi. 1988. Accuracy—a Program for Combined Problems of Energy Analysis, Indoor Airflow, and Air Quality. *ASHRAE Transactions,* 94:196-214.

Chen, Q., X. Peng, and A. Van Paassen. 1995. Prediction of Room Thermal Response by CFD Technique with Conjugate Heat Transfer and Radiation Models. *ASHRAE TRANS,* 101:50-60.

Chen, Q., and W. Xu. 1998. A Zero-Equation Turbulence Model for Indoor Airflow Simulation. *Energy and Buildings,* 28 (2):137-44.

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

Cormier, B. R., R. Qi, G. Yun, Y. Zhang, and M. S. Mannan. 2009. Application of Computational Fluid Dynamics for Lng Vapor Dispersion Modeling: A Study of Key Parameters. *Journal of Loss Prevention in the Process Industries,* 22 (3):332-52.

Corrigan, A., F. F. Camelli, R. Löhner, and J. Wallin. 2011. Running Unstructured Grid‐Based CFD Solvers on Modern Graphics Hardware. *International Journal for Numerical Methods in Fluids,* 66 (2):221-9.

Courant, R., E. Isaacson, and M. Rees. 1952. On the Solution of Nonlinear Hyperbolic Differential Equations by Finite Differences. *Communications on Pure and Applied Mathematics,* 5 (3):243-55.

Crawley, D. B., J. W. Hand, M. Kummert, and B. T. Griffith. 2008. Contrasting the Capabilities of Building Energy Performance Simulation Programs. *Building and Environment,* 43 (4):661-73.

Daoud, A. 2008. Analysis of Heat Transfer and Mass Transfer in a Transitional Arena Using the Zonal Method. *Dissertation Abstracts International,* 69 (04).

Department of Energy. "Building Energy Data Book." https://catalog.data.gov/dataset/buildings-energy-data-book.

Desta, T. Z., K. Janssens, A. Van Brecht, J. Meyers, M. Baelmans, and D. Berckmans. 2004. CFD for Model-Based Controller Development. *Building and Environment,* 39 (6):621-33.

Devulder, T., R. Wilson, and J. Chilton. 2007. The Thermal Behaviour of Buildings Incorporating Single Skin Tensile Membrane Structures. *International Journal of Low-Carbon Technologies,* 2 (2):195-213.

Djunaedy, E., J. Hensen, and M. Loomans. 2003. Toward External Coupling of Building Energy and Airflow Modeling Programs. *ASHRAE Transactions,* 109 (2):771-87.

Djunaedy, E., J. Hensen, and M. Loomans. 2004. Comparing Internal and External Run-Time Coupling of CFD and Building Energy Simulation Software. *Proceedings of the Proc. 9th Int. Conf. on Air Distribution in Rooms-ROOMVENT*.

Djunaedy, E., J. Hensen, and M. Loomans. 2005a. External Coupling between CFD and Energy Simulation: Implementation and Validation. *ASHRAE Transactions,* 111 (1):612-24.

Djunaedy, E., J. L. M. Hensen, and M. Loomans. 2005b. External Coupling between Cfd and Energy Simulation: External Coupling between Cfd and Energy Simulation. *Ashrae Transactions,* 111 (1).

Dols, W. S., and G. N. Walton. 2002. *Contamw 2.0 User Manual: Multizone Airflow and Contaminant Transport Analysis Software*: US Department of Commerce, Technology Administration, National Institute of Standards and Technology.

EnergyPlus. "Energyplus Energy Simulation Software." www.energyplus.gov.

Fan, Y. Q., and K. Ito. 2012. Energy Consumption Analysis Intended for Real Office Space with Energy Recovery Ventilator by Integrating Bes and Cfd Approaches. *Building and Environment,* 52:57-67.

Fisk, W. J. 2000. Health and Productivity Gains from Better Indoor Environments and Their Relationship with Building Energy Efficiency. *Annual Review of Energy and the Environment,* 25:537-66.

Fluent. "Fluent 16.1.0." http://www.ansys.com/Products/Fluids/ANSYS-Fluent.

Fritzson, P. 1998. Modelica - a Language for Equation-Based Physical Modeling and High Performance Simulation. *Applied Parallel Computing,* 1541:149-60.

Gao, Y. 2002. Coupling of Computational Fluid Dynamics and a Multizone Airflow Analysis Program for Indoor Environmental Design. *Cambridge, MA., Massachusetts Institute of Technology*.

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

Goldsworthy, M. 2012. Dynamic Coupling of the Transient System Simulation and Fire Dynamics Simulation Programs. *Journal of Building Performance Simulation,* 5 (2):105-14.

Gorobets, A., F. Trias, and A. Oliva. 2013a. A Parallel Mpi+ Openmp+ Opencl Algorithm for Hybrid Supercomputations of Incompressible Flows. *Computers & Fluids,* 88:764-72.

Gorobets, A., F. X. Trias, and A. Oliva. 2013b. An Opencl-Based Parallel CFD Code for Simulations on Hybrid Systems with Massively-Parallel Accelerators. *Procedia Engineering,* 61:81-6.

Griffith, B., and Q. Y. Chen. 2004. Framework for Coupling Room Air Models to Heat Balance Model Load and Energy Calculations (Rp-1222). *HVAC&R Research,* 10 (2):91-111.

Gropp, W. D., D. K. Kaushik, D. E. Keyes, and B. F. Smith. 2001. High-Performance Parallel Implicit CFD. *Parallel Computing,* 27 (4):337-62.

Gu, Y., T. Wahl, M. Bayati, and M. Leeser. 2015. Behavioral Non-Portability in Scientific Numeric Computing. In *Euro-Par 2015: Parallel Processing*, 558-69. Springer.

Havens, J., and T. Spicer. 2005. Lng Vapor Cloud Exclusion Zones for Spills into Impoundments. *Process Safety Progress,* 24 (3):181-6.

Hazyuk, I., C. Ghiaus, and D. Penhouet. 2012. Optimal Temperature Control of Intermittently Heated Buildings Using Model Predictive Control: Part I–Building Modeling. *Building and Environment,* 51:379-87.

Hepner, G. F., and M. V. Finco. 1995. Modeling Dense Gaseous Contaminant Pathways over Complex Terrain Using a Geographic Information System. *Journal of Hazardous Materials,* 42 (2):187-99.

Hiyama, K., and S. Kato. 2011. Integration of Three-Dimensional CFD Results into Energy Simulations Utilizing an Advection–Diffusion Response Factor. *Energy and Buildings,* 43 (10):2752-9.

Hiyama, K., S. Kato, and Y. Ishida. 2010. Thermal Simulation: Response Factor Analysis Using Three-Dimensional CFD in the Simulation of Air Conditioning Control. *Proceedings of the Building Simulation*.

Huang, S., and W. Zuo. 2014. Optimization of the Water-Cooled Chiller Plant System Operation. *Proceedings of the 2014 ASHRAE/IBPSA-USA Building Simulation Conference*, Atlanta, GA.

Huang, S., W. Zuo, and M. D. Sohn. 2016. Amelioration of the Cooling Load Based Chiller Sequencing Control. *Applied Energy,* 168:204-15.

Huang, S., W. Zuo, and M. D. Sohn. 2017. Improved Cooling Tower Control of Legacy Chiller Plants by Optimizing the Condenser Water Set Point. *Building and Environment,* 111:33-46.

IEEE, S. A. 2008. Standard for Floating-Point Arithmetic. *IEEE 754-2008*.

Intel. "E5-1603." http://download.intel.com/support/processors/xeon/sb/xeon_E5-1600.pdf.

Intel. "I5-3210m." http://download.intel.com/support/processors/corei5/sb/core_i5-3200_m.pdf.

Intel. "I7-2620m." http://www.intel.com/content/dam/support/us/en/documents/processors/corei7/sb/core_i7-2600_m.pdf.

Intel. "I7-3720qm." http://download.intel.com/support/processors/corei7/sb/core_i7-3700_m.pdf.

Jin, M., Q. Chen, and W. Zuo. 2013. Validation of a Fast Fluid Dynamics Program for Simulating Natural Ventilation in Buildings. *Proceedings of the CLIMA 2013, the 11th REHVA World Congress and the 8th Conference on Indoor Air Quality, Ventilation and Energy Conservation in Buildings*, Prague, Czech, June 16-19.

Jin, M., W. Zuo, and Q. Chen. 2012a. Improvement of Fast Fluid Dynamics for Simulating Airflow in Buildings. *Numerical Heat Transfer, Part B Fundamentals,* 62 (6):419-38.

Jin, M., W. Zuo, and Q. Chen. 2012b. Improvements of Fast Fluid Dynamics for Simulating Air Flow in Buildings. *Numerical Heat Transfer, Part B: Fundamentals,* 62 (6):419-38.

Jin, M., W. Zuo, and Q. Chen. 2012c. Validation of Three Dimensional Fast Fluid Dynamics for Indoor Airflow Simulations. *Proceedings of the the 2nd International Conference on Building Energy and Environment (COBEE2012)*, Boulder, CO.

Jin, M., W. Zuo, and Q. Chen. 2013. Simulating Natural Ventilation in and around Buildings by Fast Fluid Dynamics. *Numerical Heat Transfer, Part A: Applications,* 64 (4):273-89.

Kats, G. 2003. *Green Building Costs and Financial Benefits*: Massachusetts Technology Collaborative Boston, MA.

Khronos, G. "The Opencl Specification, Version 1.2." https://www.khronos.org/registry/cl/specs/opencl-1.2.pdf.

Kim, D., J. Braun, E. Cliff, and J. Borggaard. 2015. Development, Validation and Application of a Coupled Reduced-Order CFD Model for Building Control Applications. *Building and Environment,* 93:97-111.

Kim, D., W. Zuo, J. E. Braun, and M. Wetter. 2013a. Comparison of Building System Modeling Approaches for Control System Design. *Proceedings of the the 13th International Conference of the International Building Performance Simulation Association (Building Simulation 2013)*, Chambery, France.

Kim, D., W. Zuo, J. E. Braun, and M. Wetter. 2013b. Comparisons of Building System Modeling Approaches for Control System Design. *Proceedings of the Proceedings of the 13th International Conference of the International Building Performance Simulation Association (Building Simulation 2013)*.

Klein, S. A., J. A. Duffie, and W. A. Beckman. 1976. Trnsys-a Transient Simulation Program. *ASHRAE Transactions,* 82:623.

Kolokotsa, D., A. Pouliezos, G. Stavrakakis, and C. Lazos. 2009. Predictive Control Techniques for Energy and Indoor Environmental Quality Management in Buildings. *Building and Environment,* 44 (9):1850-63.

Li, D. H., L. Yang, and J. C. Lam. 2013. Zero Energy Buildings and Sustainable Development Implications–a Review. *Energy,* 54:1-10.

Li, K., H. Su, J. Chu, and C. Xu. 2013. A Fast-Pod Model for Simulation and Control of Indoor Thermal Environment of Buildings. *Building and Environment,* 60:150-7.

Liping, W., and W. N. Hien. 2007. The Impacts of Ventilation Strategies and Facade on Indoor Thermal Environment for Naturally Ventilated Residential Buildings in Singapore. *Building and Environment,* 42 (12):4006-15. doi: DOI 10.1016/j.buildenv.2006.06.027.

Liu, G., J. Zhang, and A. Dasu. 2012. Review of Literature on Terminal Box Control, Occupancy Sensing Technology and Multi-Zone Demand Control Ventilation (Dcv). *US Department of Energy, Tech. Rep*.

Lomas, K. J. 1996. The Uk Applicability Study: An Evaluation of Thermal Simulation Programs for Passive Solar House Design. *Building and Environment,* 31 (3):197-206.

Luketa-Hanlin, A., R. P. Koopman, and D. L. Ermak. 2007. On the Application of Computational Fluid Dynamics Codes for Liquefied Natural Gas Dispersion. *Journal of Hazardous Materials,* 140 (3):504-17.

Lydon, G., M. Keane, and D. Ingham. 2005. Implementation of an Interface between a CFD Energy Analysis and an Ifc Building Product Model. *Proceedings of the Information Visualisation, 2005. Proceedings. Ninth International Conference on*.

Microsoft. "Microsoft Visual Studio 2013 Professional." https://msdn.microsoft.com/en-us/library/dd831853(v=vs.120).aspx.

Nouidui, T., K. Phalak, W. Zuo, and M. Wetter. 2012. Validation and Application of the Room Model of the Modelica Buildings Library. *Proceedings of the the 9th International Modelica Conference*, Munich, Germany, September 3-5.

Nouidui, T., M. Wetter, and W. Zuo. 2012. Validation of the Window and the Room Models of the Modelica Buildings Library. *Proceedings of the the 5th National Conference of International Building Performance Simulation Association -USA (SimBuild2012)*, Madison, WI, August 1-3.

NVIDIA. 2007. *Nvidia Cuda Compute Unified Device Architecture-- Programming Guide (Version 1.1)*. Santa Clara, California: NVIDIA Corporation.

NVIDIA. "Gt 650m." https://www.techpowerup.com/gpudb/547/geforce-gt-650m.html.

Ohba, M., and I. Lun. 2010. Overview of Natural Cross-Ventilation Studies and the Latest Simulation Design Tools Used in Building Ventilation-Related Research. *Advances in Building Energy Research,* 4 (1):127-66.

Oldewurtel, F., A. Parisio, C. N. Jones, M. Morari, D. Gyalistras, M. Gwerder, V. Stauch, B. Lehmann, and K. Wirth. 2010. Energy Efficient Building Climate Control Using Stochastic Model Predictive Control and Weather Predictions. *Proceedings of the American Control Conference (ACC), 2010*.

Pan, Y., Y. Li, Z. Huang, and G. Wu. 2010. Study on Simulation Methods of Atrium Building Cooling Load in Hot and Humid Regions. *Energy and Buildings,* 42 (10):1654-60.

Patankar, S. V., and D. B. Spalding. 1972. A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-Dimensional Parabolic Flows. *International Journal of Heat and Mass Transfer,* 15 (10):1787-806.

Pope, S. B. 1997. Computationally Efficient Implementation of Combustion Chemistry Using in Situ Adaptive Tabulation. *Combust. Theory Modelling,* 1:41-63.

Pope, S. B. 2008. Algorithms for Ellipsoids. *Cornell University Report No. FDA*:08-1.

Riederer, P. 2005. Matlab/Simulink for Building and HVAC Simulation-State of the Art. *Proceedings of the Ninth International IBPSA Conference*.

Rong, Q., Y. Da, Z. Xin, and J. Yi. 2011. Research on a Dynamic Simulation Method of Atrium Thermal Environment Based on Neural Network. *Building and Environment*.

Scargiali, F., E. Di Rienzo, M. Ciofalo, F. Grisafi, and A. Brucato. 2005. Heavy Gas Dispersion Modelling over a Topographically Complex Mesoscale: A CFD Based Approach. *Process Safety and Environmental Protection,* 83 (3):242-56.

Singer, M., and S. Pope. 2004. Exploiting Isat to Solve the Reaction-Diffusion Equation. *Combustion Theory and Modelling,* 8 (2):361-83.

Singer, M., S. Pope, and H. Najm. 2006. Operator-Splitting with Isat to Model Reacting Flow with Detailed Chemistry. *Combustion Theory and Modelling,* 10 (2):199-217.

Srebric, J., J. Yuan, and A. Novoselac. 2008. On-Site Experimental Validation of a Coupled Multizone and CFD Model for Building Contaminant Transport Simulations. *ASHRAE Transactions,* 114 (1):273-81.

Steeman, H.-J. 2009. Modelling Local Hygrothermal Interaction between Airflow and Porous Materials for Building Applications.

Stockwell, D. R., and A. T. Peterson. 2002. Effects of Sample Size on Accuracy of Species Distribution Models. *Ecological modelling,* 148 (1):1-13.

Tan, G., and L. R. Glicksman. 2005. Application of Integrating Multi-Zone Model with CFD Simulation to Natural Ventilation Prediction. *Energy and Buildings,* 37 (10):1049-57.

Tian, W., T. A. Sevilla, and W. Zuo. 2017. A Systematic Evaluation of Accelerating Indoor Airflow Simulations Using Cross-Platform Parallel Computing. *Journal of Building Performance Simulation,* 10 (3):243-55. doi: 10.1080/19401493.2016.1212933.

Trcka, M., and J. Hensen. 2006. Model and Tool Requirements for Co-Simulation of Building Performance. *Proceedings of the Proceedings of the 15th IASTED Int. Conf. on Applied Simulation and Modelling*.

Trčka, M., J. L. Hensen, and M. Wetter. 2009. Co-Simulation of Innovative Integrated HVAC Systems in Buildings. *Journal of Building Performance Simulation,* 2 (3):209-30.

Trcka, M., M. Wetter, and J. Hensen. 2007. Comparison of Co-Simulation Approaches for Building and HVAC/R System Simulation. *Proceedings of the Building Simulation 2007*:1418-25.

Tuma, M. 2002. A Note on the LDLT Decomposition of Matrices from Saddle-Point Problems. *SIAM Journal on Matrix Analysis and Applications,* 23 (4):903-15.

Van Sickle, D., M. A. Wenck, A. Belflower, D. Drociuk, J. Ferdinands, F. Holguin, E. Svendsen, L. Bretous, S. Jankelevich, and J. J. Gibson. 2009. Acute Health Effects after Exposure to Chlorine Gas Released after a Train Derailment. *The American journal of emergency medicine,* 27 (1):1-7.

Villi, G., W. Pasut, and M. D. Carli. 2009. CFD Modelling and Thermal Performance Analysis of a Wooden Ventilated Roof Structure. *Proceedings of the Building Simulation*.

Walton, G. N. 1994. "Contam93 User Manual." In.: National Inst. of Standards and Technology (BFRL), Gaithersburg, MD (United States).

Wang, B., B. Zhao, and C. Chen. 2010. A Simplified Methodology for the Prediction of Mean Air Velocity and Particle Concentration in Isolation Rooms with Downward Ventilation Systems. *Building and Environment,* 45 (8):1847-53.

Wang, H., and Z. J. Zhai. 2012. Analyzing Grid Independency and Numerical Viscosity of Computational Fluid Dynamics for Indoor Environment Applications. *Building and Environment,* 52:107-18.

Wang, L., and Q. Chen. 2005. On Solution Characteristics of Coupling of Multizone and CFD Programs in Building Air Distribution Simulation. *Proceedings of the 9th International IBPSA Conference, Montréal*.

Wang, L., and Q. Chen. 2007a. Analysis on the Well-Mixing Assumptions Used in Multizone Airflow Network Models. *Proceedings of the the 10th International Building Performance Simulation Association Conference and Exhibition (Building Simulation 2007)*, Beijing, China.

Wang, L., and Q. Chen. 2007b. Validation of a Coupled Multizone-Cfd Program for Building Airflow and Contaminant Transport Simulations. *HVAC&R Research,* 13 (2):267-81.

Wang, L., and Q. Chen. 2008. Applications of a Coupled Multizone-CFD Model to Calculate Airflow and Contaminant Dispersion in Built Environments for Emergency Management. *HVAC&R Research,* 14 (6):925-39.

Wang, L., and N. H. Wong. 2008. Coupled Simulations for Naturally Ventilated Residential Buildings. *Automation in Construction,* 17 (4):386-98.

Wang, M., and Q. Chen. 2009. Assessment of Various Turbulence Models for Transitional Flows in an Enclosed Environment (Rp-1271). *HVAC&R Research,* 15 (6):1099-119.

Wang, Y., A. Malkawi, Y. Yi, and T. C. Center. 2011. Implementing CFD (Computational Fluid Dynamics) in Opencl for Building Simulation. *Proceedings of The 12th International Building Performance Simulation (Building Simulation 2011)*.

Wetter, M. 2005. Buildopt—a New Building Energy Simulation Program That Is Built on Smooth Models. *Building and Environment,* 40 (8):1085-92.

Wetter, M. 2009a. Modelica-Based Modeling and Simulation to Support Research and Development in Building Energy and Control Systems. *Journal of Building Performance Simulation,* 2 (2):143-61.

Wetter, M. 2009b. Modelica-Based Modelling and Simulation to Support Research and Development in Building Energy and Control Systems. *Journal of Building Performance Simulation,* 2 (2):143-61.

Wetter, M. 2010. Co-Simulation of Building Energy and Control Systems with the Building Controls Virtual Test Bed. *Journal of Building Performance Simulation,* 4 (3):185-203. doi: 10.1080/19401493.2010.518631.

Wetter, M. 2011. Co-Simulation of Building Energy and Control Systems with the Building Controls Virtual Test Bed. *Journal of Building Performance Simulation,* 4 (3):185-203.

Wetter, M., and C. Haugstetter. 2006. Modelica Versus Trnsys–a Comparison between an Equation-Based and a Procedural Modeling Language for Building Energy Simulation. *Proceedings of the Proc. of the SimBuild, 2nd National Conference of IBPSAUSA*.

Wetter, M., W. Zuo, and T. Nouidui. 2011. Modeling of Heat Transfer in Rooms in the Modelica "Buildings" Library. *Proceedings of the the 12th International Conference of the International Building Performance Simulation Association (Building Simulation 2011)*, Sydney, Australia, November 14-16.

Wetter, M., W. Zuo, T. S. Nouidui, and X. Pang. 2013. Modelica Buildings Library. *Journal of Building Performance Simulation*. doi: 10.1080/19401493.2013.765506.

Wetter, M., W. Zuo, T. S. Nouidui, and X. Pang. 2014. Modelica Buildings Library. *Journal of Building Performance Simulation,* 7 (4):253-70. doi: 10.1080/19401493.2013.765506.

Xcode. https://developer.apple.com/xcode/download/.

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

Yang, P. 2013. "Real-Time Building Airflow Simulation Aided by GPU and FFD." Concordia University.

Yuan, J. 2003. "Effective Prediction of Air Distribution and Contaminant Transport in Entire Buildings by Coupling Multi-Zone, CFD and Energy Models." Master's thesis, The Pennsylvania State University.

Yuan, X., Q. Chen, L. R. Glicksman, Y. Hu, and X. Yang. 1999. Measurements and Computations of Room Airflow with Displacement Ventilation. *ASHRAE Transactions,* 105:340.

Zeng, Z., X. Li, C. Li, and Y. Zhu. 2012. Modeling Ventilation in Naturally Ventilated Double-Skin Façade with a Venetian Blind. *Building and Environment*.

Zhai, Z. 2003. "Developing an Integrated Building Design Tool by Coupling Building Energy Simulation and Computational Fluid Dynamics Programs." Massachusetts Institute of Technology.

Zhai, Z., Q. Chen, P. Haves, and J. Klems. 2002a. On Approaches to Couple Energy Simulation and Computational Fluid Dynamics Programs. *Building and Environment,* 37 (8-9):857-64.

Zhai, Z., Q. Chen, P. Haves, and J. H. Klems. 2002b. On Approaches to Couple Energy Simulation and Computational Fluid Dynamics Programs. *Building and Environment,* 37 (8):857-64.

Zhai, Z., and Q. Y. Chen. 2003. Solution Characters of Iterative Coupling between Energy Simulation and CFD Programs. *Energy and Buildings,* 35 (5):493-505.

Zhai, Z. J., and Q. Y. Chen. 2005. Performance of Coupled Building Energy and CFD Simulations. *Energy and Buildings,* 37 (4):333-44.

Zhang, R., K. P. Lam, S.-C. Yao, and Y. Zhang. 2012a. "Annual Coupled Energyplus and Computational Fluid Dynamics Simulation of Natural Ventilation." In.: SIMBuild.

Zhang, R., K. P. Lam, S.-C. Yao, and Y. Zhang. 2012b. Coupled Energyplus and Computational Fluid Dynamics Natural Ventilation Simulation. *Proceedings of the Proceedings of 5th National SimBuild Conference*.

Zhang, R., K. P. Lam, S.-c. Yao, and Y. Zhang. 2013. Coupled Energyplus and Computational Fluid Dynamics Simulation for Natural Ventilation. *Building and Environment,* 68:100-13.

Zhang, W., K. Hiyama, S. Kato, and Y. Ishida. 2013. Building Energy Simulation Considering Spatial Temperature Distribution for Nonuniform Indoor Environment. *Building and environment*.

Zhao, K., X.-H. Liu, T. Zhang, and Y. Jiang. 2011. Performance of Temperature and Humidity Independent Control Air-Conditioning System in an Office Building. *Energy and Buildings,* 43 (8):1895-903.

Zuo, W., and Q. Chen. 2007. Validation of Fast Fluid Dynamics for Room Airflow. *Proceedings of the The 10th International IBPSA Conference (Building Simulation 2007)*, Beijing, China.

Zuo, W., and Q. Chen. 2009a. Fast Parallelized Flow Simulations on Graphic Processing Units. *Proceedings of the the 11th International Conference on Air Distribution in Rooms (RoomVent 2009)*, Busan, Korea.

Zuo, W., and Q. Chen. 2009b. High-Performance and Low-Cost Computing for Indoor Airflow. *Proceedings of the Proceedings of the 11th International IBPSA Conference (Building Simulation 2009)*, Glasgow, U.K.

Zuo, W., and Q. Chen. 2009c. Real-Time or Faster-Than-Real-Time Simulation of Airflow in Buildings. *Indoor Air,* 19 (1):33-44.

Zuo, W., and Q. Chen. 2010a. Fast and Informative Flow Simulations in a Building by Using Fast Fluid Dynamics Model on Graphics Processing Unit. *Building and Environment,* 45 (3):747-57.

Zuo, W., and Q. Chen. 2010b. Fast Simulation of Smoke Transport in Buildings. *Proceedings of the the 41st International HVAC&R congress*, Beograd, Serbian, December 1-3.

Zuo, W., and Q. Chen. 2010c. Simulations of Air Distribution in Buildings by FFD on GPU. *HVAC&R Research,* 16 (6):783-96.

Zuo, W., M. Jin, and Q. Chen. 2012. Reduction of Numerical Diffusion in FFD Model. *Engineering Applications of Computational Fluid Mechanics,* 6 (2):234-47.

Zuo, W., and M. Wetter. 2011. Advanced Simulations of Building Energy and Control Systems with an Example of Chilled Water Plant Modeling. *Proceedings of the The 8th International Forum and Workshop on Combined Heat, Air, Moisture and Pollutant Simulations (CHAMPS 2011)*, Nanjing, China.

Zuo, W., M. Wetter, D. Li, M. Jin, W. Tian, and Q. Chen. 2014. Coupled Simulation of Indoor Environment, HVAC and Control System by Using Fast Fluid Dynamics and Modelica. *Proceedings of the 2014 ASHRAE/IBPSA-USA Building Simulation Conference*, Atlanta, GA, Sep. 10-12.

Zuo, W., M. Wetter, W. Tian, D. Li, M. Jin, and Q. Chen. 2016. Coupling Indoor Airflow, HVAC, Control and Building Envelope Heat Transfer in the Modelica Buildings Library. *Journal of Building Performance Simulation,* 9 (4):366-81.