Incorporating Two Weeks Open-Source Software Lab Module in CFD and Fluids Courses

Sumit Verma, Zahra Mansouri, R. Panneer Selvam
Department of Civil Engineering, BELL 4190 University of Arkansas, Fayetteville, AR 72701, USA

Abstract

To train future engineers and to equip them with necessary tools and skills for real-world problem solving, it is important to provide exposure to real-world problem solving by incorporating a software lab module while teaching engineering courses such as Computational Fluid Dynamics (CFD) and/or related Fluids courses. High cost of commercial software packages and limited number of licenses available for course instruction creates several challenges in incorporating commercial software packages in the instructional workflow. To circumvent such limitations, open-source software packages could be a good alternative as open-source software packages can be downloaded and used free of cost and thus provides a wider accessibility to students and practitioners. With the same motivation, in this contribution, an outline for implementing a two-week course module by incorporating open-source software in the instructional workflow is proposed and demonstrated by considering an example of wind flow around a building. The course module outlined in this work can also be extended to formulate a full-fledged CFD course for instructional purposes. Besides the information provided in this paper, authors have also shared an extended report based on current work and the relevant case files via Github repository (https://github.com/rpsuark/ASEE21-OpenFOAM-Introduction) for a hands on learning experience. With the help of information contained in this paper along with the extended report and uploaded case files, readers can install the open-source software packages - 'OpenFOAM' and 'ParaView', make their own simple case files, run simulations, and visualize the simulated results.

Keywords

Computational Fluid Dynamics, Building Aerodynamics, OpenFOAM, ParaView.

1. Introduction

1.1 Need for Open Source CFD Program for Class Instruction

When teaching engineering courses such as Computational Fluid Dynamics (CFD), Wind Engineering and related Fluids courses, instructors are challenged to create a course content, which not only provides a good understanding about the fundamental topics of CFD and Wind Engineering but also equips students with tools for solving real-world problems of fluid flows. To develop a reliable CFD model from scratch, a sound understanding of several topics such as Fluid Dynamics, Numerical Methods, Programming, etc. is required. Developing proficiency in students on these topics within a semester or over a couple of semesters requires an overwhelming amount of work in short time. Some courses such as that from Cornell University uses commercial software package Ansys Fluent [1] for teaching Engineering Simulation. While commercial software package Ansys Fluent provides academic license to students free of cost but the functionalities available in the free version are restricted. The cost of license for different commercial software

packages can vary from one to another; however, in general, the license cost of commercial software for installation in a single PC roughly amounts to about $10,000 per year. Such high prices make the commercial software packages almost unaffordable to purchase an individual license.

The goal of engineering programs in the nation is to train students with necessary knowledge and skills enabling them to apply those skills beyond classrooms (such as in industry or research position); however, training students using commercial software packages is too expensive as stated above. In addition to high costs, there are several other challenges such as availability of a limited number of licenses which may not be sufficient for the entire class. For instance, if only 8 licenses of commercial software are available, but a class consists of 15 students, then, an alternative strategy of instruction such as dividing the class into groups and teaching the groups in different shifts must be adopted. As the same teaching material is delivered to different groups at different times, the effective class hours get compromised for learning new things. Besides, students may only be able to access the commercial software at some selected labs in the university. Due to limited available licenses, students may have to wait for their turn to run simulation jobs depending upon the load on machines. Similarly, if students must solve a fairly complex problem for their research, which requires a long simulation time (such as several days to weeks), then, the computers get locked up for a long time as the already-running simulation might make most of the processor cores in the machine busy. As a result, no further jobs can be submitted on the same machine for a long time restricting its accessibility/usage until the simulation is complete. In addition to these challenges, there are further bottlenecks if the course needs to be delivered remotely such as the instructor needs to ensure first that students can access and use commercial software installed within the university computer systems via remote access, which adds further challenges in course delivery. To tackle the challenges stated above, there is a need for an alternative to commercial software packages for wider accessibility of the class to a large group of students. In that regard, open-source software can be a good substitute in place of commercial software packages for teaching CFD, Wind Engineering and related Fluids courses. The following section provides an outline for implementing a course module by incorporating open-source software packages-OpenFOAM and ParaView, in the instructional workflow.

1.2 Suggested Course Module for Instruction

In the department of Civil Engineering at University of Arkansas, a 3-week module on ‘Introduction to Computational Fluid Dynamics’ was introduced during the course offering of CVEG-5383 (Finite Element Methods in Civil Engineering), in which students were introduced to real-world problem solving of fluid flows using modified research codes (programs/codes developed by course instructor while pursuing research work supported by various regional and national funding agencies) adapted for classroom teaching. Based on similar modality, the entire course on CVEG 563V (CFD for Wind Engineering), was taught as a combination of theoretical lectures and real-world problem solving using modified research codes. This modality was found to be useful in teaching fundamental concepts of the subject matter while also introducing the students to current research areas in the discipline using customized research codes adapted for teaching purposes.

With some changes applied to teaching modality described above, a new instruction modality for teaching CFD, Wind Engineering or related Fluids courses is proposed in this paper. The new

instruction modality consists of two weeks open-source software lab component using open-source software packages - OpenFOAM and ParaView with real-world problem-solving exercises. For demonstration purposes, an engineering problem of practical significance, i.e., wind flow around a building is considered for this work. The work described in this paper can be assimilated as a 2-week module software lab component in course offerings on CFD or related Fluids courses. In addition to the paper, additional materials such as the extended report based on this paper and the uploaded case files makes this contribution ready-to-use hands on material for instructional purposes. By expanding the scope of course outline described in this paper such as that by incorporating topics such as large eddy simulation, atmospheric boundary layer type flow profile at inlet, etc. from [2] and some additional relevant topics, a full fledge CFD course can be formulated. However, as this contribution is intended to serve as an introductory outline, so, the details on installation procedure of OpenFOAM and ParaView, setting up simple flow problems, obtaining and visualizing the simulated results are discussed here.

1.3 Additional Open-Source Material from Github for Hands on Experience

The CFD material illustrated in this paper can be adopted in a class or readers can also learn on their own with ease by referring to the extended report (*ASEE-Introduction.pdf*) and OpenFOAM case file (*buildingUniform*) available at [3] (**https://github.com/rpsuark/ASEE21-OpenFOAM-Introduction**). Due to space limitations, only key technical details and important illustrations are included in this paper. Readers can have hands on experience on OpenFOAM and ParaView with the help of information from this paper, extended report, and uploaded case files. All the hands-on experience listed below except for case file is available in *ASEE-Introduction.pdf* file. The hands-on experience from the Github page are as follows:

1. Installation of OpenFOAM and ParaView.

2. 'buildingUniform' case file that can be downloaded and used to run the job using OpenFOAM.

3. Detailed explanation on multi-block hexahedral mesh generation.

4. Procedure to run the case file and obtain ParaView visualization.

5. Step by step procedure to visualize the output file from OpenFOAM using ParaView. The explanation includes Grid visualization, contour, streamline plot and velocity vector in a 2D slice. Retrieving data from ParaView for further analysis.

1.4 Objectives of Current Work

1. To provide a detailed outline of the installation process of 'OpenFOAM' and 'ParaView' in Windows 10 operating system.

2. To demonstrate the workflow of setting up the case files and running simulations including explanation of case file structure in OpenFOAM.

3. To demonstrate the procedure of obtaining various data visualizations listed in section 1.3 above.

1.5 Proposed Outline of Two-Week Lab Module

To meet the objectives described in section 1.4 above, a tentative outline for the proposed two-week course module with a breakdown of topics, sequence of delivery of topics, time allocated for each topic and the learning deliverables from each topic is described with the help of a chart in Fig. 1 below.

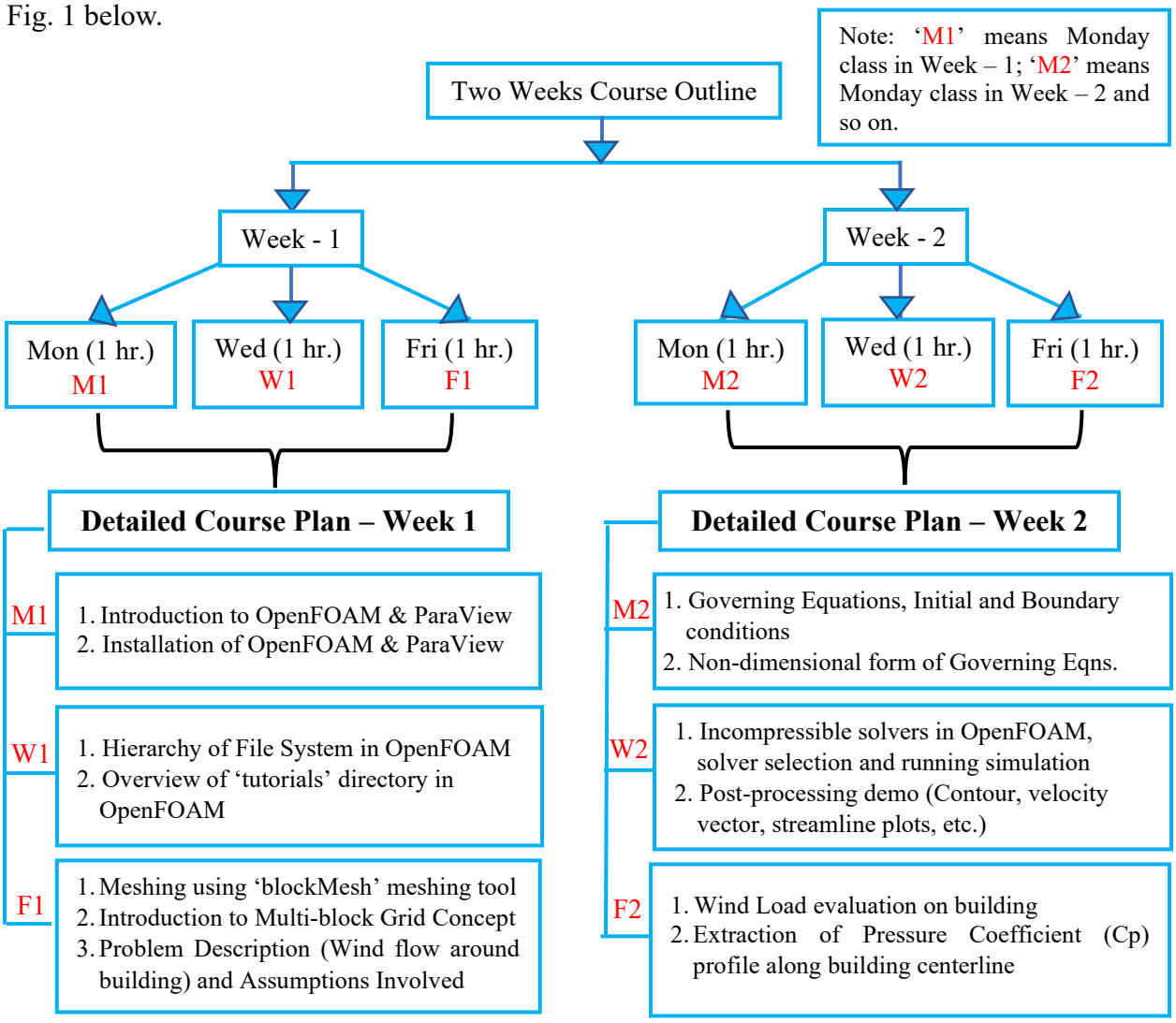


Fig. 1. Detailed course plan for proposed two-weeks (6 hrs. lectures) open-source lab module

Before delving into detailed description of problem and solution procedure, in the following section, a brief introduction to 'OpenFOAM' and 'ParaView' (open-source visualization software for solution obtained from OpenFOAM) and the key aspects of installation procedure are described briefly. For a more detailed procedure, readers are directed to an extended report (*ASEE-Introduction.pdf*) prepared by the authors which is available at [3].

2. A Brief Introduction to 'OpenFOAM' and 'ParaView'

OpenFOAM stands for 'Open-Source Field Operation and Manipulation'. It is a C++ toolbox for solving problems of continuum mechanics such as that applied to fluid flow and heat transfer problems in computational fluid dynamics (CFD). A brief introduction and overview of setting up some problems including meshing capabilities, solvers, and post-processing tools available in

OpenFOAM is described in OpenFOAM user guide [4]. OpenFOAM not only provides a platform for solving a variety of problems encountered in fluid dynamics, heat transfer, electromagnetics, and multiphase fluid flow problems, etc. but it also provides a platform to modify and develop custom solvers, applications, and libraries as per the need of users. Further details about 'OpenFOAM' programming can be obtained from OpenFOAM programmer's guide [5]. However, to develop customized applications and libraries, user should have some familiarity with C++ semantics such as object-oriented programming, classes and objects, operator overloading, inheritance, etc. Further details about the organization of namespace, classes, and file system available in OpenFOAM can be obtained from OpenFOAM source code guide [6]. However, for the scope of work described in this paper, the information in [4] is enough to set up the problem and obtain solutions. After obtaining solution from OpenFOAM, some visualization tool is required for visually inspecting the obtained solution and for further post-processing of results. For that purpose, another open-source software called 'ParaView' is considered for this work. Different techniques for visualizing and analyzing scientific data in 'ParaView' are covered in depth in the 'ParaView' documentation manual [7]. Besides, readers can also refer to section-B of an extended report (*ASEE-Introduction.pdf)* available at [3] on 'ParaView Visualization' for a detailed step by step procedure along with illustrations to obtain visualizations reported in this work.

After a brief introduction to essential software components used in current work, installation procedure of those two software components is briefly described in the following text. In section 2.1 below, the key aspects of OpenFOAM and ParaView installation in Windows 10 operating system (OS) is explained and references are cited to help the readers in installation process. For a detailed step by step procedure along with illustrations, readers can refer to section-A of an extended report (*ASEE-Introduction.pdf*) available at [3].

2.1 Setting up OpenFOAM and ParaView in Windows 10

While the OS on a machine may differ from one user (or reader) to another (i.e., Windows, Linux or macOS), in this work, Windows OS is considered as most of the students have a PC (Windows machine) of their own. As OpenFOAM runs under Linux environment, a Linux emulator (such as Ubuntu or openSUSE) must be installed in the PC. However, there is no such requirement for ParaView and it can be installed independently on a Windows machine. In the older versions of Windows OS, the feature 'Windows Subsystem for Linux' (WSL) was not available. However, in Windows 10, users can activate WSL by turning on the 'Developers Mode' feature and then installing 'Ubuntu' (a Linux distribution) in WSL platform. For a step-by-step process of activating 'Developers Mode' in Windows 10 including OpenFOAM installation, the Youtube video from [8] can be followed. Besides, readers can also refer to another Youtube video [9], which provides a good explanation for installing 'Ubuntu' in Windows 10. These resources are sufficient for the readers to download and install 'Ubuntu' in Windows 10 OS. Now, readers can proceed ahead with the installation process for OpenFOAM, which can be downloaded from OpenFOAM website cited in [10]. After downloading OpenFOAM in the PC, readers can follow along the Youtube video cited in [8] to proceed ahead with installation of OpenFOAM. Similarly, to install Paraview in PC, the download page on Paraview website can be referred [11]. These resources and the cited references should be enough to download and set up OpenFOAM as well as ParaView.

After installing and setting up the environment to run simulation jobs, an example problem (simulating wind flow around a cubical building) is considered that would fit aptly for the two weeks open-source software lab module proposed in this work. In the following section, the detailed description of problem statement, procedure of modeling geometry for simulating wind flow around building as well as meshing is described.

3. Description of Problem (Wind Flow Around a Building)

In this work, wind flow around a cubical building model is computed using OpenFOAM. For that purpose, it is necessary to choose a suitable region in space around the building model (the chosen region in space around the building is also called computational domain) in which governing fluid flow equations are solved to analyze the wind flow pattern around a building.

For this work, a cube of dimension '1H' is considered for the building model, where 'H' is the height of building. Similarly, the computational domain is made up of a cuboid of dimension '12H' in X-direction, '7H' in Y-direction and '5H' in Z-direction respectively. As shown in Fig. 2, the direction of X-axis is the stream-wise direction and thus the flow enter computational domain through face "ACDB". The flow interacts with cubical building model located inside the computational domain and then exits the domain from face "EFHG". The interaction of fluid with the building model produces aerodynamic forces on the building, which will be computed and plotted in the later section. In Fig. 2, the face "ABFE" is the top face whereas the face "BDHF" is the front face of computational domain. The top view and the front view of computational domain are shown in Fig. 3 (a) and (b) respectively.

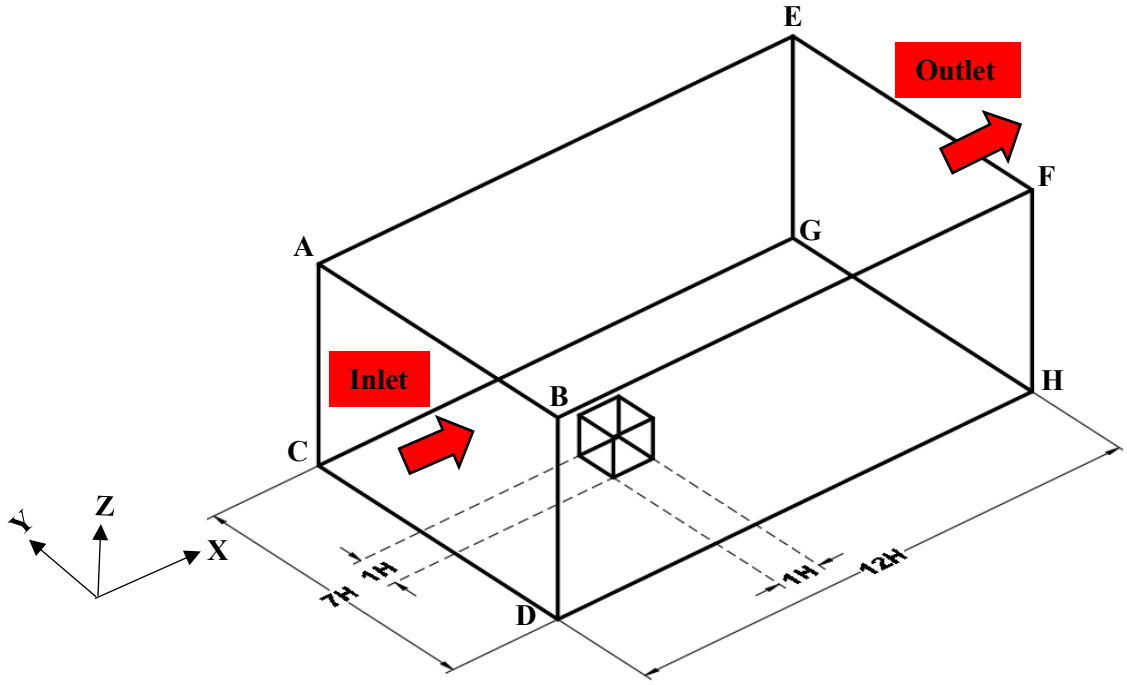
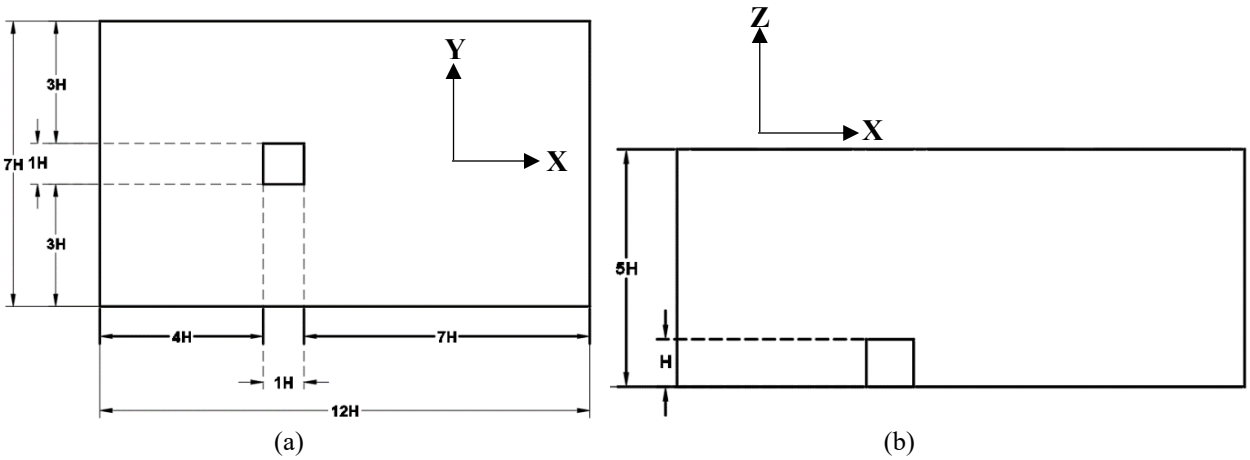


Fig. 2. Isometric view of computational domain with a cubical building model inside



(a) (b)

Fig. 3. Orthographic projection of computational domain with building (a) Top view (b) Front view

4. Hierarchical Structure (File System) of a Case Directory in OpenFOAM

Before describing the details about meshing and solvers used in current work, the hierarchy of file system in the case directory named 'buildingUniform', used for current work is described briefly. Relevant case files can be found under the same name 'buildingUniform' at [3].

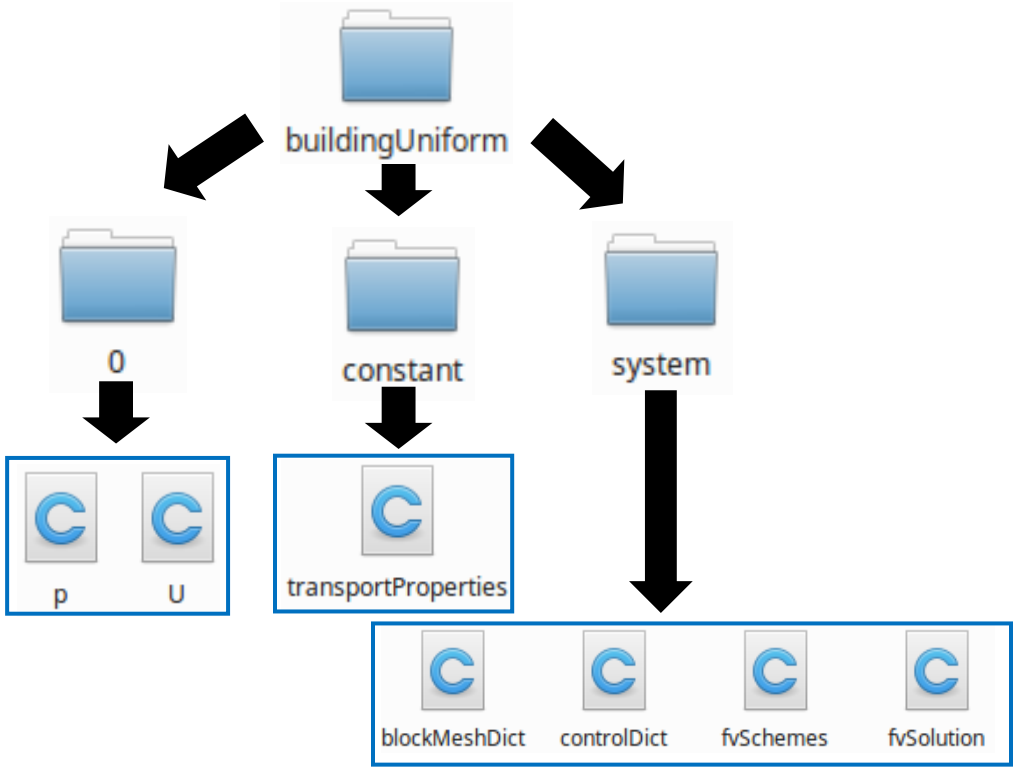


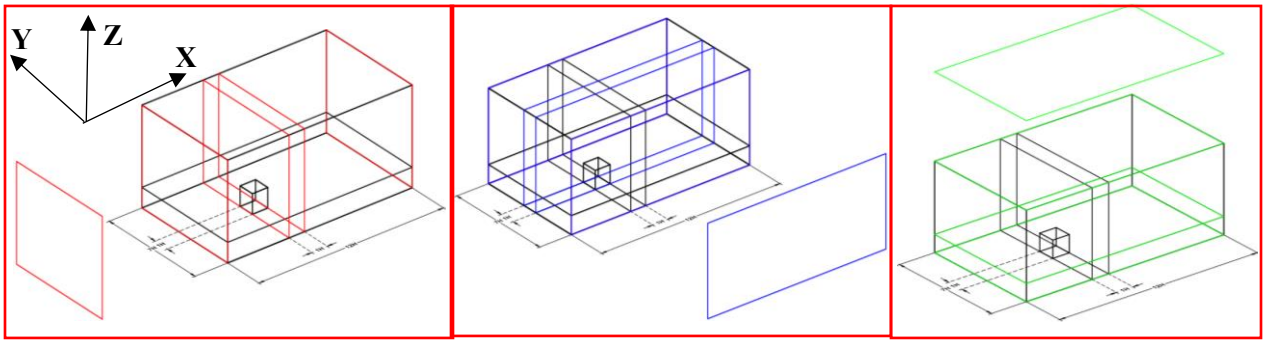
Fig. 4. Hierarchical structure of case directory considered for current problem in OpenFOAM

It should be noted that the hierarchical structure shown in Fig. 4 pertains to the problem described in current work. For some other work, the hierarchical structure may contain additional files or directories depending upon the problem; however, the basic hierarchical structure shown in Fig. 3 would stay the same even for other complicated problems. The details of the files contained inside

different directories (in hierarchical structure of Fig. 4) and the functions intended to be performed by each of those files during OpenFOAM simulation are described in extended report (*ASEE-Introduction.pdf*) available at [3]. Readers can also refer to OpenFOAM documentation guide [12] for further details.

5. Meshing of Computational Domain (Pre-Processing)

In CFD, the governing equations of fluid flow, which comprises of a set of partial differential equations (pdes) are solved by discretizing the pdes into a system of linear algebraic equations. For that purpose, the entire computational domain is discretized into several small regions, also called computational stencils or cells. The computational domain used in current work is formed from different blocks, which in turn are produced due to intersecting planes parallel to X, Y and Z-axes as shown in Fig. 5. For instance, in Fig. 5 (a), a plane normal to X-axis is placed at 4 different locations, i.e., x = 0, 4, 5, 12 (shown by red colored rectangles) whereas planes normal to Y-axis are placed at y = 0, 3, 4, 7 and planes normal to Z-axis are placed at z = 0, 1 and 5. When these planes intersect, several blocks are formed. The assembly of several blocks together makes up the computational domain for the current problem. For demonstration purposes, two such blocks (i.e., BLOCK-0 and BLOCK-2, which are color-coded red), formed due to intersecting planes at different locations as described earlier are shown in Fig. 6 below.



(a) (b) (c)

Fig. 5. Demonstration of computational domain formed by intersection of different planes normal to (a) X-axis (b) Y-axis and (c) Z-axis

5.1 Concept of Multi-Block Grid

As the geometry of computational domain considered for current work is relatively simple consisting of hexahedrons, so, 'blockMesh' utility available in OpenFOAM is considered for the current work. Further details about 'blockMesh' utility can be obtained from [13].

To clarify the concept of multi-block grid, isometric view of two block units (i.e., BLOCK-0 and BLOCK-2) are shown in Fig. 6 (a) (in red color) while other blocks are not shown to avoid congestion of labels in the figure. Also, the regions (or faces) formed due to intersection of different planes as explained earlier is shown in Figs. 6(b)-(d). In Fig. 6(b), the regions (or faces) formed due to intersection of planes at an elevation of z = 0 is shown. Similarly, in Fig. 6(c), the regions (or faces) formed due to intersection of planes at elevation of z = 1 is shown whereas that in Fig. 6(d) includes the region (or faces) at elevation of z = 5.

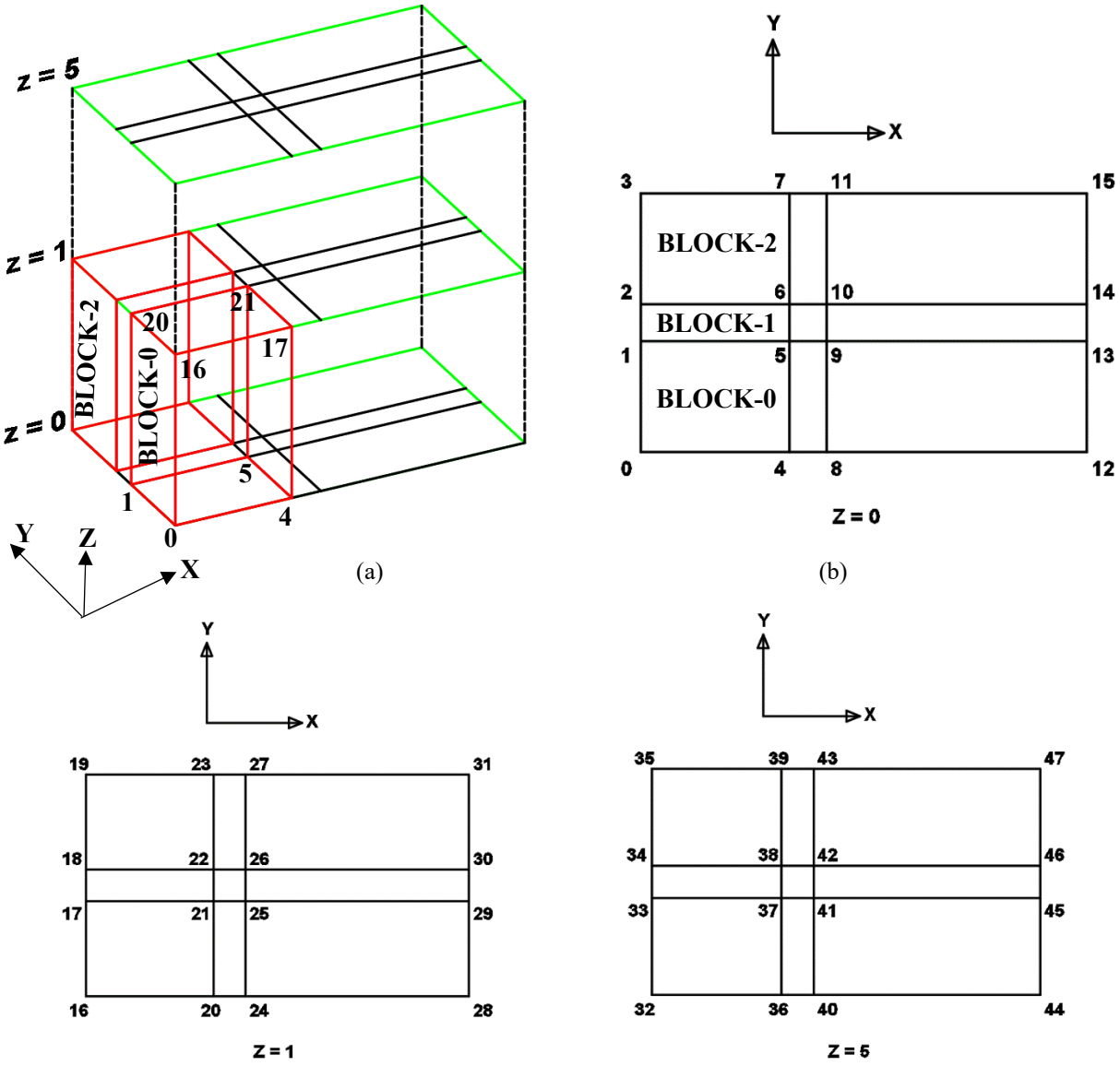


Fig. 6. (a) Isometric view of computational domain demonstrating some blocks used to make the computational domain and Numbering scheme for multi-block grid in (b) z = 0 plane (c) z = 1 plane (d) z = 5 plane (Sketches NTS)

To create a structured mesh consisting of hexahedral cells using 'blockMesh', certain set of points (or vertices) should be defined first. This is done in a file called 'blockMeshDict' inside the system directory of case file (Refer section 4). The numbering of vertices adopted for the current work is shown in Fig. 6 (b)-(d). For demonstration purposes, some vertices including their x, y and z-coordinates are shown in Fig. 7 (a). For instance, the coordinates of vertex '0' is x = 0, y = 0 and z = 0 and that of vertex '16' is x = 0, y = 0 and z = 1 and so on in Fig. 7 (a). Using the defined set of points (or vertices), several hexahedral blocks are created for the computational domain. Two blocks (i.e. BLOCK-0 & BLOCK-2) and the vertices that makes up those blocks are shown in Fig. 7 (b) for demonstration purposes, i.e. 'BLOCK-0' is made up of vertices 0,4,5,1,16,20,21,17

whereas 'BLOCK-2' is made up of vertices 2,6,7,3,18,22,23,19 and so on (Refer section 5.3 of OpenFOAM user guide [4]).

It should be noted that while specifying the vertices of block, the vertices should be listed in anticlockwise direction. For instance, the bottom plane of 'BLOCK-0' consists of vertices 0, 4, 5 and 1, which are listed in anticlockwise order (Refer Fig. 6 (b)) whereas the top plane of 'BLOCK-0' consists of vertices 16, 20, 21 and 17, which are again listed in anticlockwise order. The same idea is used while specifying the vertices of other blocks of the computational domain. In addition, the number of cells in X, Y and Z-direction are taken as 40, 30 and 10 respectively for 'BLOCK-0' (Refer. Fig. 7(b)). As the lengths in X, Y and Z-direction for 'BLOCK-0' are 4H, 3H and H respectively, 40 cells along X, 30 cells along Y and 10 cells along Z-direction implies that a grid resolution of 0.1H is achieved in each of X, Y and Z-directions. The same grid resolution is applied to the remaining blocks of computational domain in the current work.

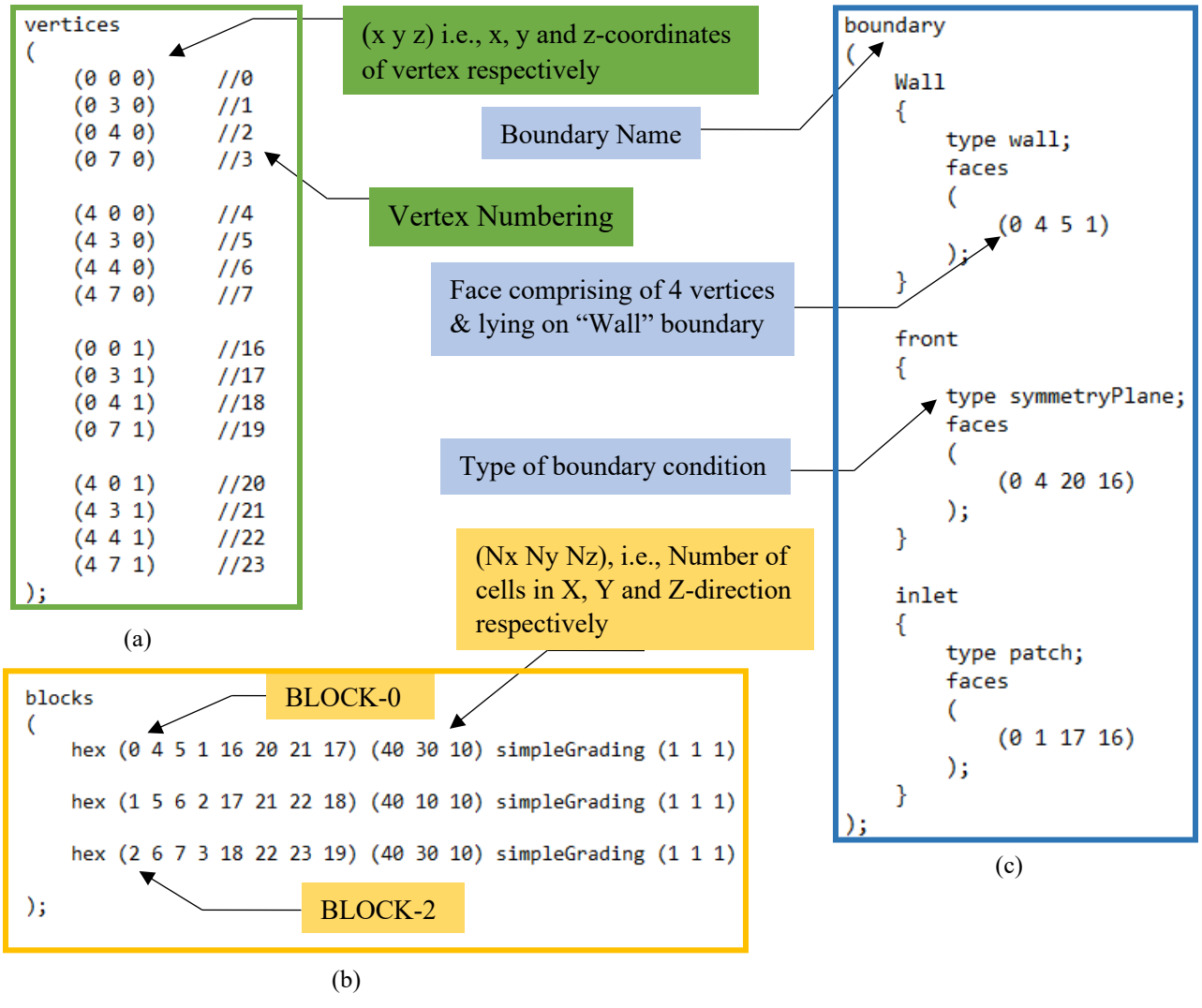
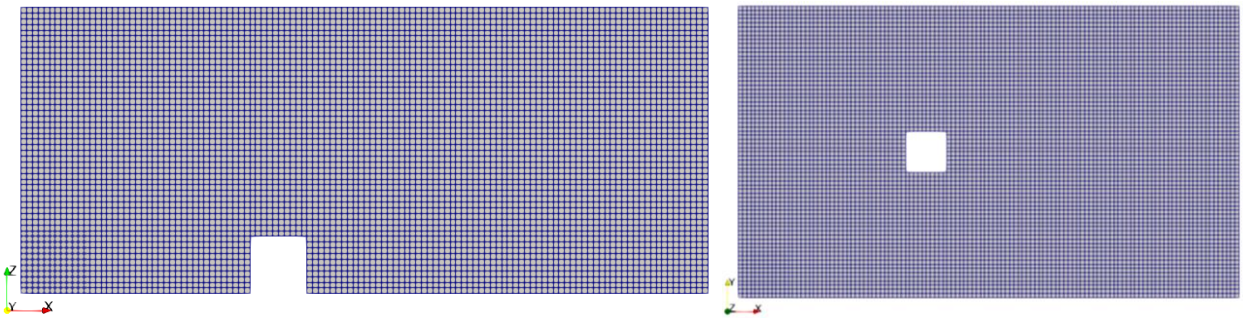


Fig. 7. Demo of (a) Vertex numbering including x, y and z-coordinates (b) vertices making hexahedral blocks for meshing of computational domain (c) boundary faces formed by combining different vertices, boundary name & type

As shown in Figs.6-7, the computational domain used for current work is composed of multiple individual blocks sharing some common vertices and faces. Although the multi-block arrangement consists of individual block units, the units are glued together through common vertices and faces and thus act as a combined whole while solving the governing equations. Similarly, some faces lying on respective boundaries such as the face consisting of vertices 0, 4, 5, 1 and lying on 'Wall' boundary and the face consisting of vertices 0, 1, 17, 16 and lying on 'inlet' boundary is shown in Fig. 7(c). It is to be noted that the vertices, blocks, and boundary faces shown in Fig. 7 are for demonstration and explanation purpose only and do not represent the entire set of vertices, faces and blocks used in the current work. For complete details, readers are directed to uploaded case files under the name 'buildingUniform' available at [3].

Using the multi-block grid concept, altogether 17 hexahedral blocks were finally used to make the computational domain in current work. Using grid resolution of 0.1H in each of X, Y and Z-direction, the total number of cells in the mesh was obtained as 419,000 (Refer Fig. 8 for mesh used in current work).



(a) (b)

Fig. 8. Mesh of computational domain in (a) XZ-plane at y = 3.5 (b) XY-plane at z = 0.9

4. Governing Equations and Solver Details

Once the geometry of computational domain is created using multi-block grid and meshing is complete, the governing equations needs to be solved next. As the problem at hand deals with wind flow around a building and the Navier-Stokes (NS) equation when implemented with suitable boundary conditions can describe the wind flow behavior around a building, so, unsteady incompressible NS equation was used for the current work. However, only laminar flow conditions are assumed for current work neglecting any turbulence effects in the wind as the Reynolds number (Re) of flow is low, i.e. Re = 100. Nevertheless, turbulent fluctuations are inherent property of wind flow in real-life scenarios and thus, modeling of wind flow around buildings including turbulence effects depicts a real-life flow scenario better and is covered in depth in another follow-up paper [2].

4.1 Non-Dimensional Form of NS Equation

While seeking computational solution to problems, the governing equations are usually expressed in non-dimensional form. Large numerical values of variables expressed in physical units can lead to diverging solutions due to non-linearity of NS equation. So, for greater numerical stability, non-

dimensional form of NS equations are preferred in CFD. For the current work, the governing equations are non-dimensionalized using two values (a) reference length taken as the building height ($L_\infty = 1$) and (b) reference velocity at inlet ($U_\infty = 1$). Using L_∞ and U_∞, the Reynolds number is computed as $\mathrm{Re} = {U_\infty L_\infty}/{\nu}\ \left(= {1}/{\nu}\right)$, where ν is the kinematic viscosity of air for the current work. Thus, $\nu = 0.01$ is considered for this work to obtain a wind flow at Re = 100 and 'ν' is defined in 'transportProperties' file under the constant directory (Refer Fig. 4). The governing equations for unsteady incompressible laminar flow in non-dimensional form expressed in vector notation are as follows:

$$\textbf{Continuity equation: } \nabla . \mathbf{U} = 0 \tag{1}$$

$$\textbf{Momentum Equation: } \frac{\partial \mathbf{U}}{\partial t} + \mathbf{U} \nabla . (\mathbf{U}) = - \nabla \mathbf{p} + {1}/{\mathbf{Re}} \nabla . (\nabla \mathbf{U}) \tag{2}$$

Further details about the process of non-dimensionalization and how the non-dimensional variables are related to dimesional variables are described in the extended report (*ASEE-Introduction.pdf*) from [3]. Besides, additional details on NS equation can be obtained from [14] and [15] whereas about the non-dimensional form of NS equations can be obtained from [16].

4.1 Initial and Boundary Conditions

The governing equations described by Eqns. (1) and (2) above is not enough to obtain wind flow around a building in computational domain. Besides, the governing equations, initial and boundary conditions are also necessary to solve and obtain wind flow around the building. It is stressed out that proper initial and boundary conditions are critically important to obtain a physically realistic solution or even to obtain a solution. As the momentum equation in (2) has non-linearity in the convection term, so, improper initial and boundary conditions provided for the problem may cause the solution to diverge very quickly.

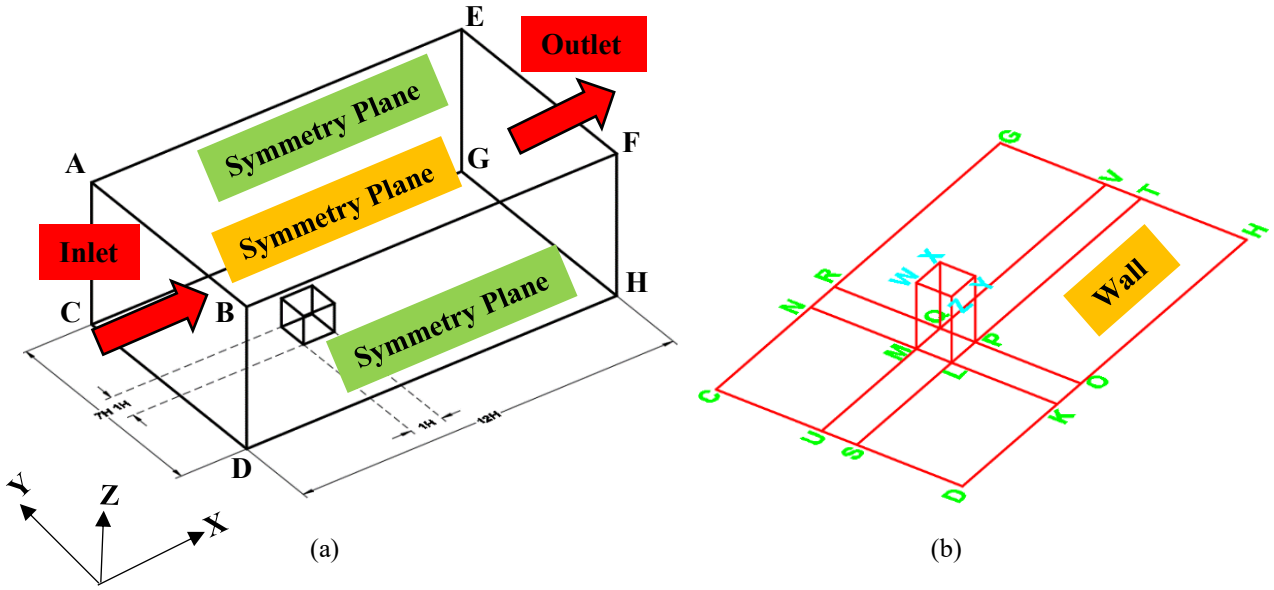


Fig. 9. (a) Isometric view of computational domain showing different boundary faces (b) Perspective view of computational domain showing the 'wall' boundary faces only

In Fig. 9(a), the isometric view of computational domain is shown including different boundary faces such as inlet, outlet, etc. Similarly, the boundary faces of computational domain where wall/no-slip boundary condition (BC) is applied is shown in Fig. 9(b). The boundary faces where no-slip condition is applied are collectively categorized under the name "FaceGroup1" for convenience of naming. As shown in Fig. 9(b), the faces on which wall BC is implemented are as follows: DKLS, SLMU, UMNC, KOPL, MQRN, OHTP, PTVQ, QVGR, MLZW, LPYZ, PQXY and QMWX.

For the initial condition, both velocity and pressure are considered '0' inside of the computational domain, i.e., **U = 0 (**or u = v = w = 0**)** and p = 0 and the applied boundary conditions on different faces of computational domain is shown in Fig. 9. Further details about the name of boundary faces, their types and their mathematical form are described in detail in the extended report (*ASEE-Introduction.pdf*) available at [3].

4.2 Solver selection for current problem in 'OpenFOAM'

Several solvers are available in OpenFOAM to solve incompressible fluid flow problems such as '**simpleFoam**', '**icoFoam**', '**pisoFoam**', etc. Further information about the different solvers available in OpenFOAM can be found at [17]. As the current work deals with unsteady incompressible fluid flow condition without consideration for turbulent fluctuations, the suitable solver for the current case is '**icoFoam**' and it is thus chosen for this work. After selecting the solver, it is necessary to specify the simulation time step for transient simulation and the total time for which the simulation needs to be carried out. In the following section, the keywords that control simulation time step, total simulation time, interval for writing data files for visualization, etc. are described.

Table 1. Important Keywords for controlling time-step and total simulation time for 'icoFoam' solver

Keyword	Values	Description
startTime (t_o)	0	Start time for simulation
endTime (t_e)	10	End time for simulation
deltaT ($\triangle t$)	5e-4	time step size for simulation
writeInterval (n)	500	write interval of data files for visualization

In Table 1, as the value of 'writeInterval' is set at n = 500, it implies that every 500 time-steps the data file will be stored for post-processing or visualization purpose. Also, as the time-step size is set at $\triangle t = 0.0005$, thus, the data files will be written after every (n x $\triangle t$ = 500 x 0.0005) 0.25 time units. Further details about the 'icoFoam" solver in OpenFOAM can be obtained from references [18] and [19].

4.3 Running Simulation in OpenFOAM

After completion of meshing and solver selection, the following commands should be executed sequentially from bash terminal to start the simulation, i.e. (1) **blockMesh** and (2) **icoFoam**. Further details about the function carried out by each of these commands are included in extended report (*ASEE-Introduction.pdf*) available at [3].

6. Results/Post-Processing

Visualization plots are convenient means to inspect the flow field and to derive meaningful flow information from the simulated results. In section 4.2, the way data files are stored by setting up the write interval inside "controlDict" file was explained. For current work, simulation was run for a total time of 10 units and the data files were written by solver every 0.25 time units. Using the written data files, the following visualization plots are made in ParaView. These visualizations are from the final time-step of simulation, i.e., corresponding to t = 10 time units. For demonstration purposes, some important visualizations are listed below. A detailed documentation of step-by-step procedure for obtaining the following plots is explained in an extended report (*ASEE-Introduction.pdf*) available at [3].

In Figs. 10(a)-(b), pressure contour and velocity contour plots for wind flow around a cubical building are shown respectively. The red colored patch right in front of building on the windward face in Fig. 10(a) represents the stagnation region formed when the fluid elements are brought to rest upon impacting the building. As the windward wall faces direct impact due to straight line winds, a positive pressure is encountered on the windward face of the building. Similarly, in Fig. 10 (b), the blue colored patch (close to 0 value on contour scale) nearby the building faces indicates that the fluid elements are brought to rest in the vicinity of building, indicating that no-slip condition is implemented properly on the faces of building.

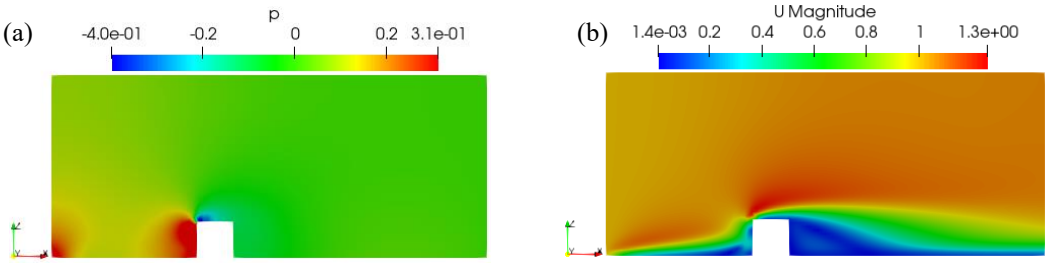


Fig. 10. Contour plots in computational domain through XZ-plane at y = 3.5 for (a) Pressure (b) Velocity

The streamline plots taken at roof height and half of roof height are shown in Fig. 11(a) and (b) respectively, in which recirculation bubble formed behind the building can be observed in Fig. 11(b). At the leeward face of building, not only a low velocity region is formed but interesting flow phenomena such as wake formation, formation of backflow region, etc., can be noticed.

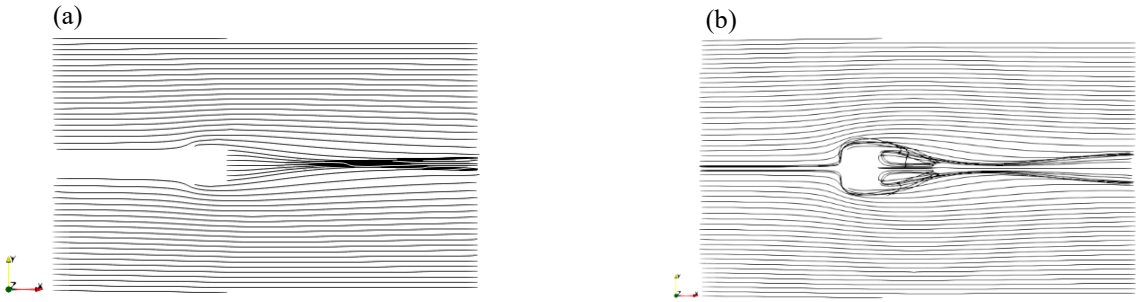


Fig. 11. Streamlines in computational domain through XY-plane at (a) z = 1.0 (b) z = 0.5

Finally, the profile of pressure coefficient along the centerline of building on the windward face, roof and leeward face of building is extracted and plotted in Fig. 12(b). To obtain pressure

coefficient profile along the centerline, the lines formed by joining vertices 0, 1, 2 and 3 (refer Fig. 12(a)) with a total length of 3 units (refer X-axis in Fig. 12(b)) was used and the cutting plane as shown in Fig. 12(a) was taken as XZ plane at y = 3.5. The pressure coefficient (Cp) was calculated from the solved pressure field using Eqn. (3).

$$C_p = \frac{(p - p_\infty)}{\frac{1}{2}\rho V_\infty^2} \tag{3}$$

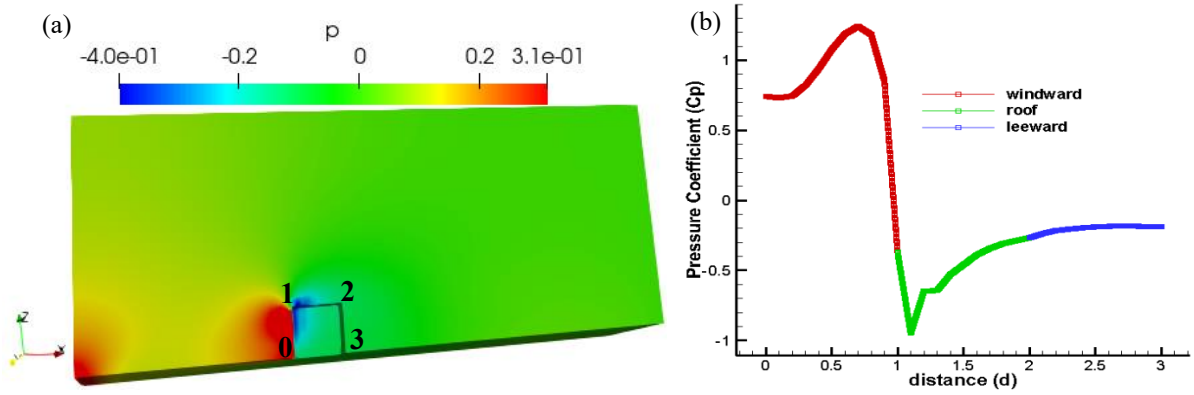


Fig. 12. (a) Clipped surface of the computational domain at XZ-plane (y = 3.5) for extraction of pressure profile along the centerline of building (b) Pressure coefficient profile along the centerline of building

In Eqn. (3), 'p' is obtained from the solved kinematic pressure field whereas 'p_∞' is the free-stream pressure value taken at the outlet for current problem. Similarly, 'ρ' is the density of fluid and 'V_∞' is the free-stream velocity of flow taken at inlet in the current case. The obtained pressure profile is important for estimation of wind loads due to straight line winds, which in turn is important for design of buildings against wind loads.

7. Conclusion

In this contribution, an outline for implementing a two-week course module by incorporating open-source software in the instructional workflow is proposed and demonstrated by considering an example problem of wind flow around a building. With the help of this brief but concise introduction along with the uploaded case files available at [3], students and practicing engineers can easily install OpenFOAM and ParaView, make their own simple case files, run simulations, and visualize results. Specifically, readers will be able to obtain several visualizations such as slicing of computational domain at different planes, obtaining contour plots (pressure and velocity contour), velocity vector plots and streamline plots for further analysis of the simulated results. Similarly, readers will also be able to obtain pressure profile along the centerline of building, which can be used further in design and analysis of building frames against wind loads.

Acknowledgements

The authors acknowledge the support received from National Science Foundation (NSF) under award number CMMI-1762999. Any opinions, findings, conclusions and/or recommendations from current work solely belong to the authors and do not necessarily reflect the views of NSF.

References

[1] FREE Cornell University Course Teaching Engineering Simulations, ansys.com https://www.ansys.com/blog/engineering-simulations-course

[2] Z. Mansouri, S. Verma and R.P. Selvam, "Teaching modeling turbulent flow around building using LES turbulence method and open-source software OpenFOAM" in Proc. 2021 ASEE Midwest Section Virtual Conference, Sep. 13-15, 2021.

[3] R. P. Selvam. ASEE21-OpenFOAM-Introduction, github.com https://github.com/rpsuark/ASEE21-OpenFOAM-Introduction

[4] C.J. Greenshields, "OpenFOAM User Guide version 8, The OpenFOAM Foundation". foam.sourceForge.net. http://foam.sourceforge.net/docs/Guides-a4/OpenFOAMUserGuide-A4.pdf

[5] C.J. Greenshields, "OpenFOAM Programmer's Guide, The Open Source CFD Toolbox". foam.sourceForge.net. http://foam.sourceforge.net/docs/Guides-a4/ProgrammersGuide.pdf

[6] OpenFOAM v8 The OpenFOAM Foundation C++ Source Code Guide, cpp.openfoam.org https://cpp.openfoam.org/v8/

[7] Welcome to ParaView Documentation!, paraview.org https://docs.paraview.org/en/latest/

[8] Jozsef Nagy. How to install OpenFOAM and run a simulation in Windows 10-tutorial. (Aug. 5, 2017). Accessed: Jun. 19, 2021. [Online Video]. Available: https://www.youtube.com/watch?v=xj0dB_PsElg

[9] ProgrammingKnowledge2. How to Install Ubuntu on Windows 10 (WSL). (Jun 8, 2020). Accessed: Jun. 19, 2021. [Online Video]. Available: https://www.youtube.com/watch?v=X-DHaQLrBi8

[10] OpenFOAM Installation on Windows 10. openfoam.com https://www.openfoam.com/download/openfoam-installation-on-windows-10

[11] Get the Software. paraview.org https://www.paraview.org/download/

[12] File structure of OpenFOAM cases. openfoam.com https://www.openfoam.com/documentation/user-guide/2-openfoam-cases/2.1-file-structure-of-openfoam-cases

[13] OpenFOAM v6 User Guide: 5.3 Mesh generation with blockMesh, cfd,direct https://cfd.direct/openfoam/user-guide/v6-blockmesh/

[14] H.K. Versteeg and W. Malalasekera, *The finite volume method for unsteady flows* in An Introduction to Computational Fluid Dynamics – the finite volume method, 2nd ed. Essex CM20 2JE, England: Pearson Education Limited, 2007.

[15] J.H. Fergizer and M. Peric, *Finite Volume Methods* in Computational Methods for Fluid Dynamics, 3rd ed. New York, USA: Springer-Verlag Berlin Heidelberg, 2002.

[16] Y.A. Cengel and J.M. Cimbala, *Approximate solutions of the Navier-Stokes equation* in Fluid Mechanics Fundamentals and Applications, 3rd ed. New York, NY 10020, USA: McGraw-Hill, 2014.

[17] Standard solvers. openfoam.com https://www.openfoam.com/documentation/user-guide/a-reference/a.1-standard-solvers

[18] Incompressible flow solver: IcoFoam, openfoamwiki.net https://openfoamwiki.net/index.php/IcoFoam

[19] OpenFOAM: User Guide v2012, openfoam.com

https://www.openfoam.com/documentation/guides/latest/doc/guide-applications-solvers-incompressible-icoFoam.html

Biographical information

Sumit Verma completed his bachelor's degree in Civil Engineering from the Institute of Engineering, Pulchowk Campus, Tribhuvan University, Nepal in 2015. Following his graduation, he worked as a Civil Engineer for an engineering consulting firm in Nepal before starting his graduate studies at the University of Arkansas from Fall of 2018. Currently, he is in the 3rd year of his Ph.D. program working as graduate research assistant in the Department of Civil Engineering at University of Arkansas.

Zahra Mansouri completed her bachelor's degree in Civil Engineering from University of Shahrekord and her master's degree in Hydraulic Engineering from K. N. Toosi University of Technology. She had 3 years of industrial experience in designing structures, including preparation of hydraulic calculations, structural design calculations, specifications, and sketches for hydraulic and residential structures. Currently, she is a Ph.D. student and graduate research assistant at University of Arkansas, Civil Engineering Department.

R. Panneer Selvam is currently the University Professor and Womble Professor of Computational Mechanics and Nanotechnology Modeling in the Department of Civil Engineering, University of Arkansas. He has been teaching finite element methods, computational fluid dynamics (CFD), CFD for wind engineering, structural dynamics, and structural loading. He has published a book on, 'Structural Dynamics and Loading' for undergraduate senior students. Currently he is writing a book on CFD for Wind Engineering to be published by Wiley in December 2021. He can be reached by email: rps@uark.edu .