This document has been superseded by a later version. For the latest version go to the web site: http://fire.nist.gov/fds

NISTIR 6784

Fire Dynamics Simulator (Version 2) – User's Guide

Kevin B. McGrattan
Glenn P. Forney
Jason E. Floyd
Simo Hostikka

National Institute of Standards and Technology • Technology Administration • U.S. Department of Commerce

NISTIR 6784

Fire Dynamics Simulator (Version 2) – User's Guide

Kevin B. McGrattan
Glenn P. Forney
Jason E. Floyd
Fire Research Division
Building and Fire Research Laboratory

Simo Hostikka
VTT Building and Transport
Finland

November 2001

U.S. Department of Commerce
Donald L. Evans, Secretary

National Institute of Standards and Technology
Arden L. Bement, Director

Disclaimer

The US Department of Commerce makes no warranty, expressed or implied, to users of the Fire Dynamics Simulator (FDS), and accepts no responsibility for its use. Users of FDS assume sole responsibility under Federal law for determining the appropriateness of its use in any particular application; for any conclusions drawn from the results of its use; and for any actions taken or not taken as a result of analyses performed using these tools.

Users are warned that FDS is intended for use only by those competent in the fields of fluid dynamics, thermodynamics, combustion, and heat transfer, and is intended only to supplement the informed judgment of the qualified user. The software package is a computer model that may or may not have predictive capability when applied to a specific set of factual circumstances. Lack of accurate predictions by the model could lead to erroneous conclusions with regard to fire safety. All results should be evaluated by an informed user.

Throughout this document, the mention of computer hardware or commercial software does not constitute endorsement by NIST, nor does it indicate that the products are necessarily those best suited for the intended purpose.

Contents

1 Introduction

Fire Dynamics Simulator (FDS) is a computational fluid dynamics (CFD) model of fire-driven fluid flow. The software described in this document solves numerically a form of the Navier-Stokes equations appropriate for low-speed, thermally-driven flow with an emphasis on smoke and heat transport from fires. The formulation of the equations and the numerical algorithm are contained in a companion document, called *Fire Dynamics Simulator (Version 2) – Technical Reference Guide* [1]. The intent of this User's Manual is to explain how calculations are performed and how to analyze the results.

1.1 What's New in FDS 2?

Version 1 of FDS was publicly released in February 2000. To date, about half of the applications of the model have been for design of smoke handling systems and sprinkler/detector activation studies. The other half consist of residential and industrial fire reconstructions. Throughout its development, FDS has been aimed primarily at the first set of applications, but it is now clear that some improvements to the fundamental algorithms are needed to address the second set of applications. The two most obvious needs are for better combustion and radiation models to handle large fires in relatively small spaces, like scenarios involving flashover.

Combustion Model Version 1 of FDS contains a relatively simple combustion model that utilizes "thermal elements," massless particles that are convected with the flow and release heat at a specified rate. While this model is easy to implement and relatively cheap computationally, it lacks the necessary physics to accommodate underventilated fires. A more comprehensive method that handles oxygen consumption more naturally solves an equation for a conserved scalar quantity, known as the mixture fraction, which is defined as the fraction of gas at a given point in the flow field that originated as fuel. The model assumes that combustion is mixing-controlled, and that the reaction of fuel and oxygen is infinitely fast. The mass fractions of all of the major reactants and products can be derived from the mixture fraction by means of "state relations," empirical expressions arrived at by a combination of simplified analysis and measurement.

Radiation Transport Version 1 of FDS has a simple radiation transport algorithm that uses randomly chosen rays between radiation sources and targets, a method commonly known as "ray tracing." This method has two major problems. The first is that only the fire itself radiates; there is no wall to wall or gas to gas radiative heat transfer. Second, the method becomes expensive when the fire begins to occupy a large fraction of the space. A better method for handling radiative heat transfer is to return to the fundamental radiation transport equation for a non-scattering gray gas. The equation is solved using a technique similar to finite volume methods for convective transport, thus the name given to it is the Finite Volume Method (FVM). Using approximately 100 discrete angles, the finite volume solver requires about 15 % of the total CPU time of a calculation, a modest cost given the complexity of radiation heat transfer.

Sprinklers Sprinklers and heat detectors are handled differently in FDS 2. The research to better characterize a sprinkler's spray distribution and droplet size has led to a need for a new file format to record everything about a particular sprinkler in one file. Much of this will be transparent to the end user who will simply specify the physical coordinates of the sprinkler in the input file. Heat detectors will be handled more simply in FDS 2. All one needs to do is specify the physical coordinates and an RTI within the input file. There still remains the option of opening VENTs or removing OBSTstructions based on the activation of a heat detector, but there is also now a much simpler way of triggering these events at a particular point in time without having to use a heat detector.

Boundary Conditions The requirement that all internal obstructions (like walls) be at least two cells thick is gone in FDS 2. This is due to a revised method of storing information about solid surfaces. It is transparent to the user, but does result in less "leakage" of heat and mass through one cell thick obstructions. Also, the presentation of surface information in Smokeview has been improved.

Opening and Closing Doors and Windows There is greater flexibility in creating and removing obstructions, and opening and closing vents in FDS 2. These actions can be directed to happen at a certain time, or they can happen in response to the activation of a thermal device.

Vents and Fans Tangential boundary conditions may be specified at mechanical vents to simulate the effect of louvers. Also, forced vents may be prescribed away from solid surfaces to simulate a jet-ventilator.

1.2 What's New in Smokeview?

The major new features in Smokeview version 2 are the two new ways of visualizing fire dynamics data: animated isosurfaces and animated flow vectors. These and other new features are detailed below.

Animated Isosurfaces Isosurface animations may be used to represent flame boundaries, layer interfaces or various other gas phase variables. Multiple isocontours may be stored in one file, allowing one to simultaneously visualize several isosurface levels for the same variable. To reduce storage requirements, isosurface data files may be compressed using the standard compression program **gzip** (supplied with Smokeview).

Animated Flow Vectors Flow vector animations, though similar to slice file animations (the vector colors are the same as the corresponding slice file colors), are better than slice file animations at highlighting flow changes.

Menus The Smokeview menus have been reorganized to accommodate new features. Commands affecting the appearance of a visualization are now grouped under `Options`.

Dialogue Boxes Several dialogue boxes have been added, allowing for easier Smokeview input. These are: an "open file" dialogue box for opening other Smokeview cases, a dialogue box for changing data bounds, a dialogue box for moving/rotating the scene and a dialogue box for creating and editing FDS blockages.

Color Bar Several options have been added for altering the color bar in order to allow one to more easily identify features and behaviors found in the simulation data. One may now flip or reverse the order of colors in the colorbar. One may also click in the colorbar and slide the mouse to highlight data values in the scene. These options may be found under `Options` > `Shades`.

Time Bar The user may now click in the time bar and slide the mouse to change the simulation time displayed. One use for the time bar and color bar selection modes might be to determine when smoke of a particular temperature enters a room. A large font is now available to display text for both the color and time bars.

Blockage and Vent Opening New Smokeview keywords have been added to the .smv file to allow Smokeview to visualize when a blockage should be hidden and when a vent should opened or closed. These keywords are added to the .smv file automatically by FDS version 2. They may be manually added to .smv files generated with FDS version 1.

Viewpoint The viewpoint and orientation coordinates may be saved in the Smokeview preference file (**casename.ini** or **smokeview.ini**) so that one may restore a scene to a desired location. This is especially useful when comparing two similar calculations.

Units Smokeview and FDS use SI units by default. Alternate units may be displayed for temperature (K and F instead of C) and for velocity (f/s and mph instead of m/s) by using the Options/Units menu item.

Scene rendering Smokeview scenes are now rendered using version 2 of the GD graphics library [2, Appendix 4]. This library supports "full color" conversion allowing for more realistic scene representations. The former version of the library (used in Smokeview version 1) allowed only 256 distinct colors to be saved into an image file. This resulted in banding or contouring in the saved image where none was present in the original display. Images in Smokeview 2 are saved in either JPEG or PNG formats. GIF support was dropped in GD 2 and is therefore not available in Smokeview 2.

Scene Clipping In complicated geometries it is often difficult to visualize slice or boundary data due to the number of obstructed surfaces. These obstructed surfaces may now be seen more easily by "clipping" part of the scene away.

Line Drawing Line drawing quality has improved by using anti-aliasing (by drawing lines more smoothly).

Frame Rate A "real time" option has been added to the max frame rate menu, so that animations can be displayed in real time or multiples of real time if desired.

Stereo 3-D Smokeview can now produce stereo 3-D images by drawing a "left" and a "right" version of each frame displayed. This feature has only been tested on Silicon Graphics workstations but should work on other computer systems that have video cards that support the use of stereo with OpenGL.

2 Getting Started

Fire Dynamics Simulator consists of two computer programs. The first, called simply **fds2**, is a Fortran 90 computer program that solves the equations described in Ref. [1]. The second, called **smokeview**, is an OpenGL graphics program that allows one to visualize the results. All of the input parameters required by **fds2** to describe the particular scenario of interest are conveyed via one or two text files created by the user. Version 2 of **smokeview** has a more expanded Graphical User Interface (GUI) than Version 1, with most parameters prescribed via the GUI and a few minor parameters prescribed via a text file.

2.1 How to get FDS and Smokeview

All of the files associated with FDS and Smokeview are linked to the URL

`http://fire.nist.gov`

Information about new versions, bug fixes, *etc.*, will be found at the web site. Since FDS Version 2 is not backward compatible, the new executable will be referred to as **fds2**. Users may want to retain copies of FDS Version 1 (**fds**) for the purpose of comparing new and old output. The graphics program Smokeview is backward compatible, and users are urged to replace the old executable file **smokeview.exe** with the new. The supporting libraries for Smokeview have not changed.

The FDS distribution consists of a self-extracting set-up program for Windows-based PCs[1]. UNIX users will be directed to a public ftp (file transfer protocol) site for source code, some compiled executables, Makefiles, *etc.* After downloading the set-up program to a PC, double-clicking on the icon will walk the user through a series of steps as the program pieces get installed. The most important part of the installation is the creation of a directory (usually called **c:\nist\fds**) in which are installed the fds2 and Smokeview executables, the GLUT graphics libraries, the Smokeview preference file **smokeview.ini** and a few directories containing sample cases, reference manuals, and supplemental data files. The set-up program also defines PATH variables and associates the **.smv** file extension to the Smokeview program so that one may either type Smokeview at any command line prompt or double click on any **.smv** file.

Users who have already downloaded version 1 of FDS will retain the same file structure as before, only now new files will be placed into various directories. To avoid naming conflicts, files associated with version 2 will typically have a 2 somehow worked into the name.

2.2 Computer Hardware Requirements

FDS requires a relatively fast CPU and a substantial amount of random-access memory (RAM). For a Windows-based PC, the processor should be at least as fast as a 450 Mhz Pentium II, with at least 256 megabytes of memory. Of course, more is better, but this configuration should allow users to do some useful calculations in a reasonable time period just to get started and evaluate the program. Serious users ought to consider purchasing a computer with at least 512 megabytes of memory, or better 1 gigabyte. Plus, a large hard drive is needed to store the output of calculations. It is not unusual for a single calculation to generate on the order of 1 gigabyte of output files. Most computers now come with hard drives of at least 20 gigabytes. For UNIX-based workstations, the processor and memory should be at least as fast and as large as the PC specifications.

[1] Some Windows PC users may want to just install Smokeview on their computers in order to view the output of FDS calculations performed by others. In this case, Smokeview may be obtained at the web site
`http://fire.nist.gov/smokeview`
This site contains a set-up program just for the Windows PC installation of Smokeview.

Most computers purchased today (November 2001) are perfectly adequate for running Smokeview with the caveat that additional memory should be obtained to bring the memory size up to at least 512 Mb in order to display results without "swapping" to disk and hence slowing down the visualizations. For Smokeview it is more important to obtain a fast graphics card than a fast CPU. If the computer is to run both FDS and Smokeview then it is important to obtain a fast CPU as well.

2.3 Computer Operating System and Software Requirements

The goal of making FDS and Smokeview publicly available has been to enable practicing fire protection engineers to perform fairly sophisticated fire simulations at a reasonable cost. Thus, FDS and Smokeview have been designed to run on both single processor UNIX/LINUX workstations and Windows-based PCs. Since most engineers use the latter, compiled versions of FDS and Smokeview are available for the PC. These programs will run under any flavor of Windows except early releases of Windows 95 that lacked the necessary OpenGL libraries needed by Smokeview[2].

UNIX and LINUX users can still run FDS and Smokeview by downloading the appropriate pre-compiled executables and installing them wherever they see fit. If the pre-compiled FDS executable does not work (usually because of library incompatibilities), the FDS source code can be downloaded and compiled using a Fortran 90 and C compiler (See Appendix A for details). If Smokeview does not work on the LINUX or UNIX workstation, one can either use a PC to view FDS output, or a note can be sent to glenn.forney@nist.gov via email requesting a custom compilation. Smokeview source code is not distributed, but a reasonable attempt will be made to compile Smokeview on whatever platform the user has available.

[2]Some users of Windows ME have noticed trouble manipulating the Smokeview window. If given a choice, one ought to run under Windows 2000 or Windows 2000 Professional.

3 Running FDS

Running **fds2** is relatively simple. All of the parameters that describe a given fire scenario are typed into a text file that will be referred to as the "data" or "input" file. In this document, the data file will be designated as **casename.data**. In practice, the user chooses the ID string "casename" so that all of the files associated with a given calculation all have a common prefix. A second text file, referred to as the "database" file, contains parameters describing specific materials and fuels referred to by name in the data file. Usually, the database file is kept in a separate directory, where it has been named **database2.data** to distinguish it from the database file released with FDS version 1. Individual sprinkler files are also stored along with the database file. The user is free to modify the database and sprinkler files or move them wherever appropriate.

In Section 6, several data files will be presented, along with entries pulled from the database file. It is suggested that a new user start with an existing data file, run it as is, and then make the appropriate changes to the input file for the desired scenario. By running a sample case, the user will become familiar with the procedure, learn how to use Smokeview, and ensure that his/her computer is up to the task before embarking on learning how to create new input files.

3.1 Creating the FDS Input Data File

The input data file provides the program with parameters to describe the scenario under consideration. The parameters are organized into groups of related variables. For example, the group `SURF` contains parameters to describe the properties of solid surfaces. Each line of the input file contains parameters belonging to the same group. These lines are written as Fortran namelist formatted records. Each record starts with the character `&` followed immediately by the name of the namelist group (`HEAD, GRID, VENT`, *etc.*), followed by a list of the input parameters corresponding to that group, and finally terminated with a slash. Details about the input parameters are to be found in in Section 5 along with several sample data files in Section 6.

3.2 Starting an FDS Calculation

Sample input files are provided with the program for new users who are encouraged to first run a sample calculation before attempting to write an input file. Assuming that an input file called **casename.data** exists in some directory, the user must run the program either in a DOS or UNIX command shell as follows:
Windows: Open up a DOS shell session, and change directories to where the data file for the case is, then run the code by typing

```
fds2 < casename.data
```

to begin a run. The character string `casename` is the name of the case designated in the user-generated input file called **casename.data**. The input parameters will be read in as standard input (indicated by the "<" sign), and the diagnostic output will be written out onto the screen. If one desires to save the diagnostic output, the job should be run

```
fds2 < casename.data > casename.out
```

The diagnostic output will then be saved to a file called **casename.out** (as indicated by the > sign). The output file can be checked periodically to monitor the progress of the calculation.

UNIX: Change directories to where the data file for the case is, then run the code by typing

```
fds2 < casename.data > casename.out &
```

to begin a run. The input parameters will be read in as standard input, and the diagnostic output will be written out as standard output to a file called **casename.out**. To watch the progress of a run on the screen, simply type

```
fds2 < casename.data
```

Note that in the latter case, the job will be run in the foreground and the diagnostic output will not be saved. It is preferable to run jobs in the background so that a record of the calculation diagnostics will be saved.

3.3 Monitoring Progress

Diagnostics for a given calculation are either written out onto the screen or into a file, usually called **casename.out** by the user. The CPU usage and simulation time will be written here, so a user can see how far along the program has progressed. At any time during a calculation, Smokeview can be run and the progress can be checked visually. To stop a calculation before its scheduled time, a user can either kill the process, or preferably create a file in the same directory as the output files called **casename.stop**. The existence of this file will stop the program gracefully, causing it to dump out the latest flow variables for viewing in Smokeview.

Since calculations can be hours or days long, there is a restart capability built into **fds2**. Details of how to use this feature are given in Section 7.1. Briefly, the user specifies at the beginning of calculation how often a "restart" file should be saved. Should something happen to disrupt the calculation, like a power outage, the calculation can be restarted from the time the last restart file was saved. Also, if the user stops the calculation by creating a file called **casename.stop**, a restart file will be created.

3.4 Reporting Bugs

Because FDS is a research code, there are inevitably going to be problems with various routines and features. The developers need to know if a certain feature is not working, and bug reporting is encouraged. However, the problem must be clearly identified. The best way to do this is to simplify the input file as much as possible so that the bug can be diagnosed. Also, limit the bug reports to those features that clearly do not work. Physical problems such as fires that do not ignite, flames that do not spread, *etc.*, may be related to the grid resolution or scenario formulation and need to be investigated first by the user before being reported.

4 Running Smokeview

Smokeview may be started almost immediately after an FDS calculation is started[3]. It is strongly recommended that Smokeview be used as soon as possible after starting an FDS calculation to ensure that the geometry has been input correctly. There is a very useful `Geometry Editor` under `Options` that allows a user to make adjustments to the geometry from within the Smokeview window. Also, as data begins to pour into the various output files, Smokeview can be used to ensure that the sequence of events set down in the FDS input file is being followed.

4.1 Starting and Stopping Smokeview

On the PC, Smokeview is started by double-clicking the file named **casename.smv** where casename is the name specified by the `CHID` keyword defined in the FDS input data file (See Section 5.1.1). The `Load/Unload` menu may be used to read in the data files to be visualized. `Show/Hide` may be used to change how the visualizations are presented. For the most part, the menu choices are self explanatory. Menu items exist for showing and hiding various simulation elements, creating screen dumps, obtaining help, *etc*.

To use Smokeview from a "command line", open a DOS (if running on a PC) or UNIX shell, change to the directory containing the FDS case to be viewed and type:

```
smokeview casename
```

where `casename` is the name specified by the `CHID` keyword defined in the FDS input data file. Data files may be loaded and options may be selected by clicking the right mouse button and picking the appropriate menu item.

Smokeview opens two windows, one displays the scene and the other displays status information. Closing either window will end the Smokeview session. Multiple copies of Smokeview may be run simultaneously if the computer has adequate resources.

4.2 Manipulating the Scene

Normally Smokeview is run during or after an FDS calculation. Smokeview may also be used as an aid in setting up FDS cases by visualizing geometric components such as blockages, vents, sensors, *etc*. One can then verify that these modeling elements have been defined and positioned as intended.

The scene may be rotated or translated either directly with the mouse or by using the scene movement dialogue box as illustrated in Figure 1. Clicking on the scene and dragging the mouse horizontally, vertically or a combination of both results in scene rotation or translation depending upon whether `Shift`, `Ctrl` or `Alt` is depressed or not during mouse movement. Axis labels are displayed, if desired, by selecting `Show/Hide`>`Labels`>`Axis labels`.

The modifier keys effect scene movement in the following ways:

No modifier key depressed Horizontal or vertical mouse movement results in scene rotation parallel to the XY or YZ plane, respectively. Note that rotation parallel to the YZ plane is disabled while *eye view* is in effect. The view is switched between *eye view* and *world view* by either depressing the `e` key or by selecting `Options`>`Rotation`. Equivalently, horizontal or vertical mouse movement results in scene rotation about the Z or X axis, respectively.

[3]Some computers store output data in buffers temporarily until enough data is available for writing out to a file(s). It may take a few minutes following the start of an FDS calculation for the necessary data to be written out, at which point Smokeview can be run.

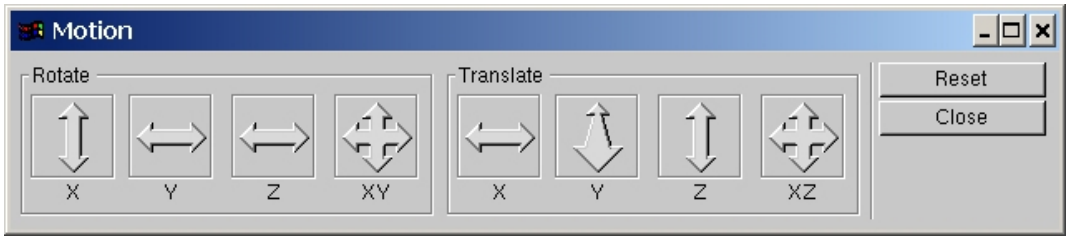


FIGURE 1: Motion Dialogue Box. Rotate or translate the scene by clicking an "arrow" and dragging the mouse. The Motion Dialogue Box is invoked by selecting `Options` > `Movement`

SHIFT key depressed Horizontal or vertical mouse movement results in scene rotation parallel to the XZ or YZ plane, respectively. Equivalently, horizontal or vertical mouse movement results in scene rotation about the Y or X axes, respectively.

CTRL key depressed Horizontal mouse movement results in scene translation from side to side along the X axis. Vertical mouse movement results in scene translation into and out of the computer screen along the Y axis.

ALT key depressed Vertical mouse movements results in scene translation along the Z axis. Horizontal mouse movement has no effect on the scene.

The scene motion dialogue box may be used to move the scene in a more controlled manner. For example, clicking on the "Rotate X" button in Figure 1 and dragging the mouse results in scene rotation about the X (and only the X) axis. Similarly, clicking the mouse in the "Translate Y" button results in scene translation along the Y axis. This dialogue box is opened from `Options` > `Movement`.

`Reset View` may be used to reset the scene back to either an external, internal (to the scene), or previously saved viewpoint.

5 Setting up the Input File for FDS

The first step in performing a calculation is to generate a text input file that will provide the program with all of the necessary information to describe the scenario under consideration. The most important inputs determine the physical size of the overall rectangular domain, the grid dimensions, and the additional geometrical features. Next, the fire must be prescribed and other boundary conditions. Finally, there are a number of parameters that customize the output files to capture the most important flow quantities. Input data is prescribed by writing a small file that uses the namelist formatted records. Each line of the file begins with the character & followed immediately by the name of the namelist group (`HEAD, GRID, VENT`, *etc.*), followed by a space or comma delimited list of the input parameters corresponding to that group. Each list is terminated with a slash. Note that the parameters listed are only those that the user desires to be changed from the default. The structure of an input file is shown below.

```
&HEAD CHID='sample',TITLE='A Sample Input File' /
&GRID IBAR=24,JBAR=24,KBAR=48 /
&PDIM XBAR0=-.30,XBAR=0.30,YBAR0=-.30,YBAR=0.30,ZBAR=1.2 /
&TIME TWFIN=10. /
&MISC RADIATION=.FALSE. /
&SURF ID='burner',HRRPUA=1000. /
&OBST XB=-.20,0.20,-.20,0.20,0.00,0.05,SURF_IDS='burner','INERT','INERT' /
&VENT CB='XBAR' ,SURF_ID='OPEN' /
&VENT CB='ZBAR' ,SURF_ID='OPEN' /
&SLCF PBY=0.,QUANTITY='TEMPERATURE' /
&BNDF QUANTITY='HEAT_FLUX' /
```

The parameters in the input file can be integers (`IBAR=24`), reals (`XBAR=0.30`), groups of reals (`XB=-.20,0.20,...`), character strings (`CHID='sample'`), groups of character strings (`SURF_IDS = 'burner' 'INERT' 'INERT'`), or logicals (`RADIATION=.FALSE.`). A logical parameter is either `.TRUE.` or `.FALSE.` – the periods are a Fortran programming convention. Character strings that are listed in this User's Manual ought to be copied exactly as written – the code is case sensitive and underscores *do* matter. Parameters can be separated by either a comma, space, or line break. Comments and notes can be written into the file so long as nothing comes between the ampersand & and the slash / except appropriate parameters corresponding to that particular namelist group.

Rarely does anyone actually write an input file from scratch. Usually, users take one of the input files that are distributed with the FDS release and modify it appropriately. It is strongly encouraged that when looking at a new scenario, the user ought to first select a pre-written input file that resembles the case, make the necessary changes, then run the case at fairly low resolution to determine if things are set up correctly. It is best to start off with a relatively simple file that captures the main features of the problem without getting tied down with too much detail that might shroud a fundamental flaw in the calculation. Initial calculations ought to be gridded coarsely so that the run times will be less than an hour and corrections can easily be made without wasting too much time.

5.1 Preliminaries

The first few lines in the input file handle some custodial details like naming files and establishing some global parameters. The most important of these is the MISC (miscellaneous) namelist group which serves as a catch-all for a variety of inputs, some of which are listed here, and many others described further on when specific features are discussed.

5.1.1 Naming the Job: The HEAD Namelist Group

The first thing to do when setting up an input file is to give the job a name. The namelist group HEAD contains two parameters. CHID is character string of 30 characters or less used to tag output files with a given character string. If, for example, CHID='sample', it is convenient to name the input data file **sample.data** so that the input file can be associated with the output files. No periods are allowed in CHID because the output files are tagged with suffixes that are meaningful to certain computer operating systems. TITLE is a character string of 60 characters or less that describes the problem.

```
&HEAD CHID='sample',TITLE='A Sample Input File' /
```

5.1.2 Setting Time Limits: The TIME Namelist Group

TIME is the name of a group of parameters defining the time duration of the simulation and the initial time step used to advance the solution of the discretized equations. Usually, only the duration of the simulation is listed on this line, via the parameter TWFIN or Time When FINished. The default is 1 s. The initial time step size can also be prescribed with DT. This parameter is normally set automatically by dividing the size of a grid cell by the characteristic velocity of the flow. During the calculation, the time step will be adjusted so that the CFL condition is satisfied. The default value of DT is $5\,(\delta x\,\delta y\,\delta z)^{\frac{1}{3}}/\sqrt{gH}$ s, where δx, δy, and δz are the dimensions of the smallest grid cell, H is the height of the computational domain, and g is the acceleration of gravity.

5.1.3 Setting Global Parameters: The MISC Namelist Group

MISC is the namelist group of miscellaneous input parameters. Only one MISC line should be entered in the data file. The MISC parameters vary in scope and degree of importance. The most important parameter in this category is the one that determines whether a Large Eddy Simulation (LES) calculation is to be performed, or whether a Direct Numerical Simulation (DNS) is to be performed. By default, an LES calculation will be performed. If a DNS calculation is desired, enter DNS=.TRUE. on the MISC line. An example of a MISC line is

```
&MISC SURF_DEFAULT='PINE',REACTION='WOOD',
      DATABASE='c:\nist\fds\database2.data' /
```

This establishes that all bounding surfaces are to be made of PINE unless otherwise indicated, that the combustion stoichiometry will be similar to WOOD, and that the definition of PINE, WOOD, and other keywords throughout the input file will be found in the file defined by DATABASE. Other inputs found on the MISC line include:

DATABASE A character string indicating the name of a file that contains information about surface materials and reaction parameters for various fuels. This need only be specified if reaction parameters or surface materials other than those specified by the user are called for.

DATABASE_DIRECTORY A character string indicating the full path name of the directory where the database and sprinkler files are stored. If DATABASE_DIRECTORY is specified, there is no need to specify a DATABASE file, it will be assumed to be **database2.data**.

SURF_DEFAULT Character string indicating which of the listed SURF IDs is to be considered the default. The default is 'INERT'. SURF is a namelist group describing the properties of vents and surfaces, and will be discussed in Section 5.3.1.

REACTION Character string indicating which of the listed groups of reaction (REAC) parameters are to be used. The default is 'PROPANE', meaning that unless the user says otherwise, it will be assumed that the fuel is propane. See Section 5.3.2 for a description of reaction parameters.

TMPA Ambient temperature in degrees Celsius. (Default 20 °C)

TMPO Temperature outside the computational domain, in degrees Celsius. (Default 20 °C)

NFRAMES Default number of output dumps per calculation. Thermocouple data, slice data, particle data, and boundary data will be saved every TWFIN/NFRAMES unless otherwise specified with DTSAM on the THCP, SLCF, PART, and BNDF namelist lines. (Default 1000)

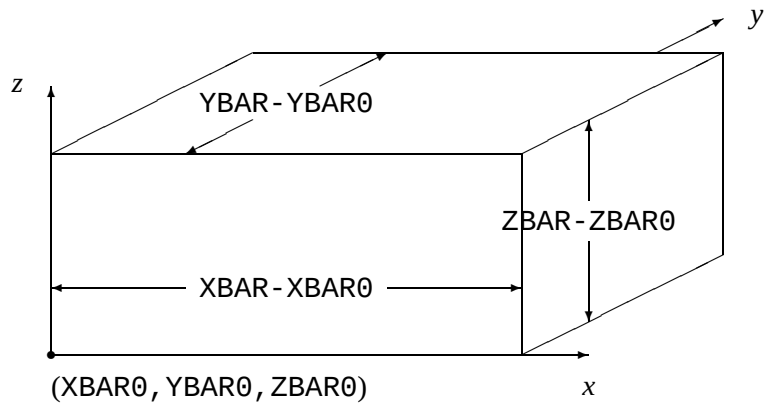


FIGURE 2: Domain geometry.

5.2 The Numerical Grid

All FDS calculations must be performed within a rectangular domain on a rectilinear numerical grid. All obstructions, vents, *etc.* will be forced to conform with the numerical grid established by the user. Creating a grid is relatively easy – first, the overall dimensions of the computational domain are specified via the PDIM namelist group, second, the number of grid cells spanning each coordinate direction are specified via GRID, and finally, the grid cells can potentially be stretched or shrunk in two of three coordinate directions via the TRNX, TRNY, and/or TRNZ groups (See Section 7.2).

5.2.1 Defining the Computational Domain: The PDIM Namelist Group

PDIM is the name of the group of parameters defining the size of the physical domain. The coordinate system spanned by these dimensions conforms to the right hand rule (See Fig. 2). The physical domain is a single right parallelepiped, *i.e.* a box. The origin of the domain is the point (XBAR0,YBAR0,ZBAR0), and the opposite corner of the domain is at the point (XBAR,YBAR,ZBAR). By default, XBAR0, YBAR0, ZBAR0 are zero, in which case the physical dimensions of the domain are given as XBAR, YBAR and ZBAR in units of meters. Unless otherwise directed, the domain will be subdivided uniformly to form a grid of IBAR by JBAR by KBAR cells specified by the GRID namelist group. If it is desired that the grid cells not be uniform in size, then the namelist groups TRNX, TRNY and/or TRNZ may be used to alter the uniform gridding (See Section 7.2).

5.2.2 Setting the Grid Size: The GRID Namelist Group

The namelist group GRID contains the dimensions of the computational grid. The grid consists of IBAR cells in the x direction, JBAR cells in the y direction, and KBAR cells in the z direction. Usually, the z direction is assumed to be the vertical direction. The longer horizontal dimension should be taken as the x-direction. Note that it is best if the grid cells are close to cubes, that is, the length, width and height of the cells ought to be roughly the same. Also, because an important part of the calculation uses a Poisson solver based on Fast Fourier Transforms (FFTs), the dimensions of the grid should each be of the form $2^l 3^m 5^n$, where l, m and n are integers. For example, $64 = 2^6$, $72 = 2^3 3^2$ and $108 = 2^2 3^3$ are good grid dimensions. However, 37, 99 and 109 are not.

```
&GRID IBAR=64,JBAR=32,KBAR=32 /
```

Following is a list of numbers between 1 and 1024 that can be factored down to 2's, 3's and 5's:

```
   2     3     4     5     6     8     9    10    12    15
  16    18    20    24    25    27    30    32    36    40
  45    48    50    54    60    64    72    75    80    81
  90    96   100   108   120   125   128   135   144   150
 160   162   180   192   200   216   225   240   243   250
 256   270   288   300   320   324   360   375   384   400
 405   432   450   480   486   500   512   540   576   600
 625   640   648   675   720   729   750   768   800   810
 864   900   960   972  1000  1024
```

5.3 Prescribing the Geometry and the Fire

Most of the work in setting up a calculation lies in specifying the geometry of the space to be modeled and applying boundary conditions to these objects. The geometry is described in terms of rectangular obstructions that can heat up, burn, conduct heat, *etc.*; and vents from which air or fuel can be either injected into the flow domain or drawn from it. A boundary condition needs to be assigned to each obstruction and vent describing its thermal properties. A fire is just one type of boundary condition. The following namelist groups describe how to prescribe the boundary conditions and the obstructions and vents to which the boundary conditions are assigned.

5.3.1 Prescribing Boundary Conditions: The SURF Namelist Group

SURF is the namelist group that defines boundary conditions for all solid surfaces or openings within or bounding the flow domain. The physical coordinates of obstructions or vents are listed in the OBST and VENT namelist groups below. Boundary conditions for the obstructions and vents are prescribed by referencing the appropriate SURF line(s) whose parameters will be described presently.

The default boundary condition for all solid surfaces is that of a cold, inert wall. If only this boundary condition is needed, there is no need to add any SURF lines to the input file. If additional boundary conditions are desired, they are to be listed one boundary condition at a time. Each SURF line consists of an identification string ID='...' to allow references to it by an obstruction or vent. Thus, on each OBST and VENT line, the character string SURF_ID='...' indicates the ID of the SURF line containing the desired boundary condition parameters. If a particular SURF line is to be applied as the default boundary condition, CONCRETE for example, set SURF_DEFAULT='CONCRETE' on the MISC line.

Fire (Mixture Fraction Model) A fire is basically modeled as the ejection of pyrolyzed fuel from a solid surface or vent that burns when mixed with oxygen. This is the default mixture fraction model of combustion. The user specifies either a heat release rate per unit area **or** a heat of vaporization at the fuel surface. The stoichiometry of the reaction is set by the parameter REACTION on the MISC line. All of the species associated with the combustion process are accounted for by way of the mixture fraction variable and should not be explicitly prescribed by the user. The exception to this rule is where a non-reacting gas is introduced into the domain that merely serves as a diluent, like CO_2 from an extinguisher or H_2O from evaporated sprinkler droplets (see Section 7.4 for details). If a finite rate combustion model is desired instead of the default mixture fraction model, see Section 7.5.

Following is a list of parameters that are prescribed on a SURF line to designate a fire using the mixture fraction approach.

HRRPUA Heat Release Rate Per Unit Area (kW/m^2). This parameter is used to control the burning rate of the fuel, as in the case of a prescribed fire using a gas burner. If all one desires is a fire of a given size, then HRRPUA is the only thing that need be set, for example

```
&SURF ID='FIRE',HRRPUA=500. /
```

will apply 500 kW/m^2 to any surface with the attribute SURF_ID='FIRE'. See the discussion of **Time Dependent Boundary Conditions** below to learn how to ramp the heat release rate up and down.

HEAT_OF_VAPORIZATION (kJ/kg). This is an alternative to HRRPUA. This is the amount of energy required to vaporize a solid or liquid fuel once it has reached its ignition temperature TMPIGN. If it is desired that the burning rate of the fuel be dependent on heat feedback from the fire, use this parameter

rather than HRRPUA. Do *not* specify HRRPUA and HEAT_OF_VAPORIZATION on the same SURF line. They are mutually exclusive inputs. Also, if HEAT_OF_VAPORIZATION is specified for a given material, something else must serve as an ignition source to ignite the material.

DENSITY **or** SURFACE_DENSITY The density (kg/m^3) or surface density (kg/m^2) of the fuel. This parameter is only needed if it is desired that the fuel eventually burn away. Only one of these parameters ought to be prescribed. If neither is prescribed, the fuel will never be exhausted. Note that if SURFACE_DENSITY is prescribed, when the fuel is exhausted, the underlying solid will remain intact, whereas if DENSITY is prescribed, the underlying solid will be removed from the calculation.

Thermal Boundary Conditions: There are four types of thermal boundary conditions: fixed temperature solid surface, fixed heat flux solid surface, thermally-thick solid or thermally-thin sheet. For a given boundary condition (*i.e.* for the same SURF line), choose only one of these. For a solid surface of fixed temperature, set TMPWAL to be the surface temperature in units of °C. For a solid surface of fixed convective heat flux, set HEAT_FLUX to be the convective heat flux in units of kW/m^2. If HEAT_FLUX is positive, the wall will heat up the surrounding gases. If HEAT_FLUX is negative, the wall will cool the surrounding gases.

A solid surface that heats up due to radiative and convective heat transfer from the surrounding gas can be either thermally-thick or thermally-thin. For a thermally-thick solid, prescribe the thermal diffusivity ALPHA (m^2/s), the thermal conductivity KS (W/m·K), and the thickness DELTA (m) of the material. Setting these parameters will direct the code to perform a one-dimensional heat transfer calculation across the thickness of the material. The thickness is *not* the same as the thickness of the entire wall, but rather the lining material that forms the outermost layer of the wall. See discussion below for more details. For thermally-thin wall linings, prescribe C_DELTA_RHO, the product of the specific heat (kJ/kg-K), density (kg/m^3), and thickness (m) of the liner. A thermally-thin liner is assumed to be the same temperature throughout its width.

Fixed temperature or fixed heat flux boundary conditions are easy to apply, but only of limited usefulness in real fire scenarios. In most cases, walls, ceilings and floors are made up of several layers of lining materials, the most important of which is the outermost layer. FDS only considers the thermal properties for this outermost layer. It is assumed that either this layer backs up to an air gap at ambient temperature (like a sheet of gypsum board attached to wood studs), or it backs up to an insulated material in which case no heat is lost to the backing material. *In neither case does the heat get transferred through the entire wall into the next room.* By default, it is assumed that the wall liner backs up to an air gap. If the wall liner is assumed to back up against an insulating material, like a sheet of steel attached to a fiber insulating board, the expression BACKING='INSULATED' on the SURF line will prevent any heat loss from the back side of the material. An example of where this might be used is in home furnishing. Recent work by Fleischmann and Chen [3] on the ignition properties of upholstery suggests that treating a fabric covered slab of polyurethane foam as thermally-thin produces a slightly better correlation than thermally-thick. If their thermally-thin data is used, the attribute BACKING='INSULATED' should be invoked.

The emissivity of a solid surface may be set with EMISSIVITY, which is 1 by default. If the wall lining material is flammable, set its ignition temperature with TMPIGN, the temperature (C) at which the material begins burning. This is only set if the wall liner is thermally-thick or thermally-thin. Heat fluxes to solid surfaces are both convective and radiative. If TMPIGN is set, a heat release rate per unit area HRRPUA or HEAT_OF_VAPORIZATION should also be given. (Default: TMPIGN is 5000 °C, *i.e.* no burning will occur unless this parameter is explicitly prescribed.)

Following are a few examples of SURF lines. These and several others are found in the DATABASE file.

```
&SURF ID          = 'CONCRETE'
      FYI         = 'Thermally-thick material'
```

```
      ALPHA       = 5.7E-7
      KS          = 1.0
      DELTA       = 0.2 /
&SURF ID          = 'UPHOLSTERY'
      FYI         = 'Fleischmann and Chen, 100% acrylic'
      C_DELTA_RHO = 1.29
      BACKING     = 'INSULATED'
      TMPIGN      = 280.
      DENSITY     = 20.0
      HEAT_OF_VAPORIZATION=2500. /
&SURF ID          = 'SHEET METAL'
      FYI         = 'Thermally-thin material'
      C_DELTA_RHO = 4.7 /
```

Velocity Boundary Condition Velocity boundary conditions effect both the normal and tangential components of the velocity vector at boundaries. The normal component of velocity is controlled by the parameter VEL. If VEL is negative, the flow is entering the computational domain. If VEL is positive, the flow is exiting the domain. Sometimes it is desired that a given volume flux through a vent be prescribed rather than a velocity. If this is the case then VOLUME_FLUX can be prescribed instead of VEL. The units are m^3/s. If the flow is entering the computational domain, VOLUME_FLUX should be a negative number. Note that either VEL or VOLUME_FLUX should be prescribed, not both. The choice depends on whether an exact velocity is desired at a given vent, or whether the given volume flux is desired. Keep in mind that the dimensions of the vent that are prescribed will usually change because the prescribed vent dimensions are sometimes altered so that the vent edges line up with grid cells. Also note that a SURF group with a VOLUME_FLUX prescribed should only be called by a VENT, not an OBST. Finally, note that if HRRPUA or HEAT_OF_VAPORIZATION is prescribed, no velocity should be prescribed. The combustible gases will be ejected at a velocity computed by the code.

As an example, a simple blowing vent would be described by the line

```
&SURF ID='BLOWER',VEL=-1.2,TMPWAL=50. /
```

The vent with SURF_ID='BLOWER' would blow 50 °C air at 1.2 m/s into the flow domain. Making VEL positive would suck air out, making TMPWAL unnecessary.

The tangential velocity boundary condition controls how the gas "sticks" to a solid surface. In theory, the tangential component of velocity is zero at the surface, but increases rapidly through a narrow region called the boundary layer. For most practical problems, the grid is not fine enough to resolve the boundary layer, which is typically a few millimeters thick. For this reason, in an LES calculation, the velocity at the wall is set to be a fraction of its value in the grid cell adjacent to the wall. Only in a DNS calculation is the velocity at the wall set to zero. To alter these defaults the user can set a parameter called VBC. This parameter ranges from -1 to 1. If a no-slip wall is desired, VBC=-1. If a free-slip wall is desired, VBC=1. Numbers in between -1 and 1 can represent partial slip conditions, which may be appropriate for simulations involving large grid cells. (Default VBC is 0.5 for LES, -1.0 for DNS)

In the case of a blowing vent (or even a solid surface), it is possible to prescribe both the normal and tangential components of the flow (or just the tangential). The normal component is specified with VEL as described above. The tangential is prescribed via a pair of real numbers VEL_T representing the desired tangential velocity components. For example, the line

```
&SURF ID='LOUVER',VEL=-1.2,VEL_T=0.5,-0.3 /
```

is a boundary condition for a louver vent that pushes air into the space with a normal velocity of 1.2 m/s, and with a tangential velocity of 0.5 m/s in either the *x* or *y* direction and -0.3 m/s in either the *y* or *z* direction, depending on what the normal direction is.

Miscellaneous Boundary Conditions If PARTICLES=.TRUE. is included on the SURF line, colored particles will be ejected from the surface. Particles may be colored by specifying PARTICLE_COLOR='RED', 'BLUE', 'BLACK', 'YELLOW', 'GREEN', 'MAGENTA', 'WHITE' or 'CYAN'

Time Dependent Boundary Conditions At the start of any calculation, the temperature is ambient everywhere, the flow velocity is zero everywhere, nothing is burning, the mass fractions of all species are uniform. When the calculation starts temperatures, velocities, burning rates, *etc.*, are ramped-up from their starting values because nothing can happen instantaneously. By default, everything is ramped-up to their prescribed values in roughly 1 s. However, the user can control the rate at which things turn on, or turn off, by specifying time histories for the boundary conditions that are listed on a given SURF line. The above boundary conditions can be made time-dependent using either prescribed functions or user-defined functions. The parameters TAU_Q and TAU_V indicate that thermal or hydrodynamic quantities are to ramp up to their prescribed values in TAU seconds and remain there. TAU_Q is the characteristic ramp-up time of the heat release rate per unit area HRRPUA or wall temperature TMPWAL. TAU_V is the ramp-up time of the normal velocity at a surface VEL or the volume flux VOLUME_FLUX. If TAU_Q is positive, then the heat release rate ramps up like $\tanh(t/\tau)$. If negative, then the HRR ramps up like $(t/\tau)^2$. If the fire ramps-up like following a t^2 curve, it will remain constant after TAU_Q seconds. These rules apply to TAU_V as well. The default value for all TAUs is 1 s. If something other than a tanh or t^2 ramp up is desired, then a user-defined burning history must be input. To do this, set RAMP_Q or RAMP_V equal to a character string designating the ramp function to use for that particular surface type, then somewhere in the input file generate lines of the form:

```
 &RAMP ID='rampname1',T= 0.0,F=0.0 /
 &RAMP ID='rampname1',T= 5.0,F=0.5 /
 &RAMP ID='rampname1',T=10.0,F=0.7 /
                      .
                      .
                      .
 &RAMP ID='rampname2',T= 0.0,F=0.0 /
 &RAMP ID='rampname2',T=10.0,F=0.3 /
 &RAMP ID='rampname2',T=20.0,F=0.8 /
                      .
                      .
                      .
```

Here, T is the time, and F indicates the fraction of the heat release rate, wall temperature, velocity, mass fraction, *etc.*, to apply. Linear interpolation is used to fill in intermediate time points. Be sure that the prescribed function starts at T=0., the ignition time. Note that the TAUs and the RAMPs are mutually exclusive. For a given surface quantity, both cannot be prescribed.

As an example, the simple blowing vent from above can be controlled via the lines

```
&SURF ID='BLOWER',VEL=-1.2,TMPWAL=50.,
      RAMP_V='BLOWER RAMP',
      RAMP_Q='HEATER RAMP' /
&RAMP ID='BLOWER RAMP',T= 0.0,F=0.0 /
```

```
&RAMP ID='BLOWER RAMP',T=10.0,F=1.0 /
&RAMP ID='BLOWER RAMP',T=80.0,F=1.0 /
&RAMP ID='BLOWER RAMP',T=90.0,F=0.0 /
&RAMP ID='HEATER RAMP',T= 0.0,F=0.0 /
&RAMP ID='HEATER RAMP',T=20.0,F=1.0 /
&RAMP ID='HEATER RAMP',T=30.0,F=1.0 /
&RAMP ID='HEATER RAMP',T=40.0,F=0.0 /
```

Now the temperature and velocity of the incoming air stream would follow the designated ramp functions. Note that the temperature and velocity are independently controlled.

5.3.2 Combustion Parameters: The REAC Namelist Group

There are two ways of designating a fire: the first is to prescribe a Heat Release Rate Per Unit Area HRRPUA on a SURF line. The other is to prescribe a HEAT_OF_VAPORIZATION, in which case the burning rate of the fuel will depend on the net heat feedback to the surface. In both cases, the mixture fraction combustion model will be used. The REAC line is used for various parameters associated with the gas phase reaction of fuel and oxygen. Note that if the user desires simply to specify a fire's heat release rate via HRRPUA, then these parameters usually do not require adjusting. However, if the user is defining the burning behavior of a fuel via a HEAT_OF_VAPORIZATION, care must be taken in selecting the appropriate reaction parameters. See Section 5.3.3 for additional details.

It is assumed that a single hydrocarbon fuel is being burned, with constant yields of CO and soot

$$\nu_f\ C_xH_yO_z + \nu_{O_2}\ O_2 \rightarrow \nu_{CO_2}\ CO_2 + \nu_{H_2O}\ H_2O + \nu_{CO}\ CO + \nu_{Soot}\ Soot \qquad (1)$$

The user specifies the *ideal* stoichiometric coefficients for the fuel, O_2, CO_2 and H_2O, and yields for CO and soot. The yield of CO, for example, is defined as the fraction of the fuel mass that is converted into CO, and is denoted y_{CO}. The user does not directly specify the stoichiometric coefficients for CO or soot because these numbers are usually reported in terms of mass yields. It is assumed that the CO and soot are created at the flame and transported with the combustion products. No growth, oxidation or after burning is assumed. Research continues to refine the handling of these products of incomplete combustion. For the moment, the ideal stoichiometric coefficients for O_2, CO_2 and H_2O are adjusted to account for the production of CO and soot.

The following parameters may be prescribed on the REAC line. Note that only one REAC line may be used. Most often, the user will select a reaction from the DATABASE via the REACTION parameter on the MISC line.

ID A character string naming the reaction. It is optional if the REAC line is included within the data input file, mandatory if the REAC line is included in the DATABASE file.

NU_O2, NU_H2O, NU_FUEL, NU_CO2 Ideal stoichiometric coefficients for the reaction of a hydrocarbon fuel. All numbers are positive. Default values are those of propane.

MW_FUEL Molecular weight of the fuel (g/mol). Default is for propane, 44 g/mol.

SOOT_YIELD The fraction of fuel mass converted into smoke particulate, y_s. Note that this parameter does not apply to the processes of soot growth and oxidation, but rather to the net production of the smoke particulate from the fire. (Default 0.01)

CO_YIELD The fraction of fuel mass converted into carbon monoxide, y_{CO}. Normally, this parameter need not be set, because by default y_{CO} is linked to the SOOT_YIELD, y_s, via the correlation developed by Köylü and Faeth

$$y_{CO} = \frac{12\,x}{M_f \nu_f}\, 0.0014 + 0.37\, y_s$$

where x is the number of carbon atoms in a fuel molecule, M_f is the molecular weight of the fuel, and ν_f is the stoichiometric coefficient of the fuel [4]. Note that this correlation refers to well-ventilated fires. Yields of CO and soot in underventilated fires is still a subject of active research.

EPUMO2 Energy Per Unit Mass Oxygen, ΔH_{O_2} (kJ/kg). The amount of energy released per unit mass of oxygen consumed. (Default 13,100 kJ/kg). Note that the heat of combustion

$$\Delta H \approx \frac{\nu_{O_2}\, M_{O_2}}{\nu_f\, M_f}\ \Delta H_{O_2}$$

but will be modified slightly due to the presence of soot and CO.

RADIATIVE_FRACTION The fraction of energy released from the flame as thermal radiation. This parameter requires some interpretation. See Section 5.3.3 for details. (Default 0.35)

A few sample REAC lines are given here. More can be found in the DATABASE file.

```
&REAC ID='METHANE'
      MW_FUEL=16
      NU_O2=2.
      NU_CO2=1.
      NU_H2O=2.
      RADIATIVE_FRACTION=0.15
      SOOT_YIELD=0.01 /

&REAC ID='WOOD'
      SOOT_YIELD = 0.01
      NU_O2      = 3.7
      NU_CO2     = 3.4
      NU_H2O     = 3.1
      MW_FUEL    = 87.
      EPUMO2     = 8850. /
```

5.3.3 Important Issues Related to Combustion

FDS version 2 contains more detailed information about the combustion process than version 1. For some calculations, these details will be irrelevant because the user will simply want a certain prescribed heat release rate. In other cases, the user may want to specify details about the combustion process, but should be aware of various approximations being made within FDS related to the simplified combustion model, less than desirable grid resolution, and parameters that are not well known because of the difficulty in obtaining them experimentally. This section will explain the various approximations being made. These approximations affect both the gas phase parameters (REAC line) and the solid (or liquid) phase parameters (SURF line).

Heat Release Rate: For coarsely gridded calculations, it is not possible to resolve the fire adequately, and as a result the heat release rate and flame height can be underestimated [5]. There are several ways to remedy

the problem, one of which is to choose a different value of the mixture fraction when defining the flame sheet. The program has built into it a routine that redefines the stoichiometric value of the mixture fraction

$$\frac{Z_{f,eff}}{Z_f} = \min\left(1\ ,\ C\,\frac{D^*}{\delta x}\right) \tag{2}$$

where Z_f is the ideal stoichiometric value of the mixture fraction, C is an empirical constant (not an input parameter), and D^* is the characteristic fire diameter

$$D^* = \left(\frac{\dot{Q}}{\rho_\infty c_p T_\infty \sqrt{g}}\right)^{\frac{2}{5}} \tag{3}$$

"Good" or "bad" resolution depends on both the size of the fire and the size of the grid cells. For relatively small values of D^* and/or large grid cells, the stoichiometric value of the mixture fraction will be intentionally reduced, resulting in a better estimate of flame height and structure. In cases where D^* is relatively large and the cells relatively small, no adjustment need be made to the stoichiometric value of the mixture fraction. As always, a sufficiently resolved grid eliminates the need for these types of corrections, but in many instances it is difficult to both finely grid the region near the fire while at the same time saving enough cells for the rest of the space.

By default, the re-adjustment of the stoichiometric value of the mixture fraction is applied automatically. To turn it off, specify `AUTOMATIC_Z=.FALSE.` on the `MISC` line, in which case the ideal stoichiometric value of the mixture fraction will be used to mark the flame zone. Also, minor adjustments can be made to the flame height by setting `Z_CONSTANT` to a value different than 1 on the `REAC` line. This parameter is essentially a normalized version of $1/C$ from Eq. (2). Reducing `Z_CONSTANT` will shorten the flame length; increasing it will lengthen the flame.

Radiative Fraction: The fraction of energy released from the fire as thermal radiation is a function of both the flame temperature and chemical composition, neither of which are well known in a large scale fire calculation. In calculations in which the grid cells are on the order of a centimeter and larger, the temperature near the flame surface cannot be relied upon when computing the source term in the radiation transport equation, especially because of the T^4 dependence. To compensate, if one prescribes a positive value to `RADIATIVE_FRACTION`, a grid cell cut by the flame will radiate that fraction of the chemical energy being released into it. Some of that energy may be reabsorbed elsewhere, yielding a χ_R that is less than `RADIATIVE_FRACTION`, depending mainly on the size of the fire. If the user desires simply to use the radiation transport equation as is, then `RADIATIVE_FRACTION` ought to be set to zero.

Heat of Combustion: The user does not usually prescribe directly the heat of combustion. Normally, the stoichiometric parameters listed on the `REAC` line are used to compute the heat of combustion. However, if a `HEAT_OF_VAPORIZATION` has been specified on a `SURF` line and the heat of combustion of the material differs from that specified by the governing `REACTION`, then the user ought to add a `HEAT_OF_COMBUSTION` (kJ/kg) to the `SURF` line. In a mixture fraction based combustion model, it is assumed that there is only one fuel. However, in a realistic fire scenario, there may be many fuels originating from the various burning objects in the building. In the model, the user specifies the stoichiometry of the combustion reaction via the `REAC` namelist group. Often these parameters are stored in the `DATABASE` file, in which case the user simply prescribes a `REACTION` on the `MISC` line. If the stoichiometry of the burning material differs from the global reaction, the `HEAT_OF_COMBUSTION` is used to ensure that an equivalent amount of fuel is injected into the flow domain from the burning object.

Predicting the Burning Rate: The user has a choice either to prescribe the burning rate of the fuel via the parameter `HRRPUA` on the `SURF` line, or the user can let the burning rate be predicted based on the

fuel's thermal properties and its HEAT_OF_VAPORIZATION. This latter action introduces a fair amount of uncertainty into the calculation because the burning rate is a function of the energy fed back from the fire via convection and radiation, and the energy conducted through the solid or liquid fuel. The various phenomena are still subjects of active research, thus the user ought to be aware of the potential errors introduced into the calculation. One of the greatest sources of error is the radiative heat flux from the fire to the fuel surface. The error is due to a combination of insufficient grid resolution in the boundary layer, and uncertainty in the absorption coefficient and flame temperature. As a result, the heat flux to the fuel surface is often over-predicted. Until improvement can be made in the radiation calculation, the user can prevent excess pyrolysis of fuel by prescribing a BURNING_RATE_MAX (kg/m^2/s) on the SURF line that will limit the burning rate of the fuel to its measured maximum.

5.3.4 Creating Obstructions: The OBST Namelist Group

OBST is the namelist group listing information about obstructions. Each OBST line contains the coordinates of a rectangular solid within the flow domain. This solid is defined by two points (x_1,y_1,z_1) and (x_2,y_2,z_2) that are entered on the OBST line in terms of the sextuplet XB=X1,X2,Y1,Y2,Z1,Z2. In addition to the coordinates, the boundary conditions for the obstruction can be specified with the parameter SURF_ID, which designates which SURF group to apply at the surface of the obstruction. If the obstruction has different boundary conditions for its top, sides and bottom, then instead of prescribing only one boundary condition with SURF_ID, use SURF_IDS, an array of three character strings specifying the boundary condition IDs for the top, sides and bottom of the obstruction, respectively. If the default boundary condition is desired, then SURF_ID(S) need not be set. However, if at least one of the surface conditions for an obstruction is the inert default, it can be referred to as 'INERT'.

Obstructions may be created or removed during a simulation. See Section 7.3 for details. Also, obstructions may be colored by specifying BLOCK_COLOR='RED', 'BLUE', 'BLACK', 'YELLOW', 'GREEN', 'MAGENTA', 'WHITE' or 'CYAN'. To prevent an obstruction from being "painted" with boundary information, set BNDF_BLOCK=.FALSE. on the OBST line.

Example:

```
&OBST XB=2.3,4.5,1.3,4.8,0.0,9.2,SURF_IDS='FIRE','INERT','INERT' /
```

will put a fire on top of the obstruction. This is a simple way of prescribing a burner.

5.3.5 Designating Vents and Surfaces: The VENT Namelist Group

Whereas the OBST group is used to prescribe obstructions within the computational domain, the VENT group is used to prescribe planes adjacent to obstructions or external walls. The vents are chosen in a similar manner to the obstructions, with the sextuple XB denoting a plane abutting a solid surface. Two of the six coordinates must be the same, denoting a plane as opposed to a solid. An easy way to specify an entire external wall is to replace XB with CB, a character string whose value is one of the following: 'XBAR', 'XBAR0', 'YBAR', 'YBAR0', 'ZBAR' or 'ZBAR0' denoting the planes $x =$ XBAR, $x =$ XBAR0, $y =$ YBAR, $y =$ YBAR0, $z =$ ZBAR or $z =$ ZBAR0, respectively. Like an obstruction, the boundary condition index of a vent is specified with SURF_ID, indicating which of the listed SURF lines to apply. If the default boundary condition is desired, then SURF_ID need not be set.

The term "VENT" is somewhat misleading. Taken literally, a VENT can be used to model components of the ventilation system in a building, like a diffuser or a return. In these cases, the VENT coordinates form a plane on a solid surface forming the boundary of the duct. No holes need to be created through the solid; it is assumed that air is pushed out of or sucked into duct work within the wall. Less literally, a VENT is used simply as a means of applying a particular boundary condition to a rectangular patch on a solid surface.

A fire, for example, is usually created by first generating a solid obstruction via an OBST line, and then specifying a VENT somewhere on one of the faces of the solid with a SURF_ID pointing to a SURF line with the characteristics of the thermal and combustion properties of the fuel.

There are two reserved SURF_ID's that may be applied to a VENT. The first is SURF_ID='OPEN'. This is used only if the VENT is applied to the exterior boundary of the computational domain, where it denotes a passive opening to the outside. It is assumed here that the exterior boundary of the computational domain is a solid wall, and the OPEN vent is essentially an open door or window. The second reserved SURF_ID is for a symmetry plane, in which case the VENT has the attribute SURF_ID='MIRROR'. As with OPEN, a MIRROR can only be prescribed at an exterior boundary of the computational domain. Often, OPEN or MIRROR VENTs are prescribed along an entire side of the computational domain, in which case the "CB" notation is handy.

Vents to the outside of the computational domain (OPEN vents) may be opened or closed during a simulation. See Section 7.3 for details.

There is one exception to the rule that VENTs ought to be prescribed flush against a solid obstruction or external boundary. A VENT that is prescribed in the interior of the domain without any solid surface flush up against it can act as a fan. For example, the lines

```
&PDIM XBAR=1.0,YBAR=1.0,ZBAR=1.0 /
                .
                .
&SURF ID='BLOW',VEL=2.1,TAU_V=5.0 /
&VENT XB=0.50,0.50,0.25,0.75,0.25,0.75,SURF_ID='BLOW' /
```

create a plane within the unit cube domain that blows air at 2.1 m/s in the positive x direction, ramping up in about 5 s. In general, the value of VEL associated with a VENT that is not aligned with a solid surface will blow gases in the coordinate direction normal to the VENT – positive if VEL is positive, negative if VEL is negative. Note that only velocity boundary conditions are appropriate on a SURF line associated with a free-standing VENT because there is no solid obstruction on which to impose thermal or material boundary conditions.

5.4 Sprinklers and Detectors

Sprinklers and heat detectors are handled differently in FDS2. The research to better characterize a sprinkler's spray distribution and droplet size has led to a need for a new file format to record everything about a particular sprinkler in one file. Much of this will be transparent to the end user who will simply specify the physical coordinates of the sprinkler in the data file **casename.data**. The data for the particular sprinkler will be kept in a file called **sprinkler_name.spk**. Heat detectors will be handled more simply in FDS2. All one needs to do is specify the physical coordinates and an RTI within the data file **casename.data**. There still remains the option of opening VENTs or removing OBSTstructions based on the activation of a heat detector, but there is also now a much simpler way of triggering these events at a particular point in time without having to use a heat detector.

5.4.1 Specifying Sprinklers: The SPRK Namelist Group

Information about a given sprinkler is contained in a separate file, called, for example, **sprinkler_name.spk**[4]. A sample file is shown in Fig. 3. All the sprinkler characteristics will be stored in this file. The user will only have to provide lines of the form

```
&SPRK XYZ=3.0,5.6,2.3,MAKE='sprinkler_name' /
```

in the input data file **casename.data**. The physical coordinates of the sprinkler are given by a triplet of real numbers XYZ. The make of the sprinkler is given by a character string MAKE. The sprinkler will activate automatically based on its thermal properties listed in the sprinkler description file **sprinkler_name.spk**. The user has the option of manually activating the sprinkler by setting T_ACTIVATE in seconds on the SPRK line itself.

In addition to the sprinkler parameters listed on the SPRK line, the user may also want to adjust some parameters that control the number and frequency of droplet insertions. These parameters are typically entered under the namelist group PART. See Section 5.5.6 for details.

Sprinkler properties listed in the file **sprinkler_name.spk** are

RTI Response Time Index of the sprinkler in units of $\sqrt{\mathrm{m}\cdot\mathrm{s}}$. (Default 165.)

C-FACTOR C-Factor of sprinkler in units of $\sqrt{\mathrm{m/s}}$. (Default 0.)

K-FACTOR K-Factor of sprinkler in units of L/min/bar$^{\frac{1}{2}}$. (Default 166) The flow rate will be given by $\dot{m}_w = K\sqrt{p}$ where $\dot{m}_w$ is the flow rate in L/min, K the K-factor in L/min/(bar)$^{\frac{1}{2}}$ and p the gauge pressure in bar[5]

ACTIVATION_TEMPERATURE Link activation temperature (C). (Default 74 °C)

OPERATING_PRESSURE Sprinkler operating pressure in units of bar. This is the pressure at which the sprinkler was tested. If the user desires to flow the sprinkler at a different pressure than that at which it was tested, a single line

```
&PIPE PRESSURE=1.5 /
```

should be added to the input data file **casename.data**, and then all sprinklers will operate at this new pressure, 1.5 bar for example. The reason for the two pressures is that a slight adjustment is

[4] The MISC character string DATABASE_DIRECTORY can be used to point to a directory where sprinkler files are stored.

[5] 1 bar is equivalent to 14.5 psi, 1 gpm is equivalent to 3.785 L/min, 1 gpm/psi$^{\frac{1}{2}}$ is equivalent to 14.41 L/min/bar$^{\frac{1}{2}}$.

made to the droplet size distribution to account for the fact that the sprinkler will operate at a different pressure than that at which it was tested. In the absence of any user specified PRESSURE, the OPERATING_PRESSURE listed in the **.spk** file will be used.

OFFSET_DISTANCE Distance in meters from the sprinkler orifice where the water droplets are initialized. It is assumed that beyond the OFFSET_DISTANCE the droplets have completely broken up. (Default 0.10 m)

VELOCITY Description of the initial droplet velocity distribution. There are two options of inputs here, designated by either a 1 or a 2 on the line immediately following the keyword VELOCITY. For case 1, the parameters: Minimum Spray Angle, Maximum Spray Angle, and Speed should be given values. The angles outline a conical spray pattern relative to the bottom of the sphere (south pole). For example, a Minimum Spray Angle of 20° and a Maximum Spray Angle of 80 ° directs the water droplets to leave the sprinkler through a conical region 20° north of the south pole and 10° south of the equator. The droplets are uniformly distributed within this belt. If more detailed information about the sprinkler spray is known, then the keyword VELOCITY is followed by a 2, indicating that the normal component of velocity of the droplets will be given as a function of n_a azimuthal points starting at the bottom of the sphere, and n_l longitudinal points starting at one of the frame arms. The integers n_a and n_l should be listed on the line following the "2", as in the example (Fig. 3). The case 2 VELOCITY data is applicable on a sphere centered about the sprinkler whose radius is the OFFSET_DISTANCE. This is typically the distance from the sprinkler itself where all the spray characteristic measurements have been made.

FLUX Relative water mass flux (kg/m^2/s), entered in the same way as the case 2 for VELOCITY. If case 1 for VELOCITY is used, there is no need to even specify a FLUX.

SIZE_DISTRIBUTION Information about the droplet size distribution. Presently, the droplet size distribution data can only be prescribed as global values. Note in the sample file that the keyword SIZE_DISTRIBUTION is followed by the number 1 to indicate that the information given is a global average. The next line has the median volumetric diameter (800 μm), plus two parameters needed for the Rosin-Rammler distribution, γ (2.4) and σ (0.6), respectively. Normally, only the droplet median diameter is changed. See the Technical Reference Guide [1] for more details about the droplet size distribution.

5.4.2 Specifying Heat Detectors: The HEAT Namelist Group

HEAT is the namelist group used to specify heat detectors. A heat detector is designated in the data input file **casename.data** with a line(s) of the form

```
&HEAT XYZ=3.0,5.6,2.3,RTI=132.,ACTIVATION_TEMPERATURE=74. /
```

Like a sprinkler, the physical coordinates of the heat detector are given by a triplet of real numbers XYZ. RTI is the Response Time Index in units of $\sqrt{\text{m}\cdot\text{s}}$. ACTIVATION_TEMPERATURE is the link activation temperature in degrees C (Default 74 °C). Note that a heat detector can be used to trigger an event to happen, like the opening of a VENT to the outside or the removal of an OBSTruction. See Section 7.3 for details.

```
MANUFACTURER
Acme
MODEL
Splash2000
OPERATING_PRESSURE
0.50
K-FACTOR
80.
RTI
110.
C-FACTOR
0.
ACTIVATION_TEMPERATURE
74.
OFFSET_DISTANCE
0.10
SIZE_DISTRIBUTION
1
800 2.43 0.6
VELOCITY
2
61,36
6.3,6.4,6.5, ... (61 azimuthal points)
     .
     .
     .
FLUX
2
61,36
11.9,13.0,13.5, ...
     .
     .
     .
```

FIGURE 3: A sample sprinkler file.

5.5 Output Files

Before a calculation is started, the user ought to carefully consider what information should be saved. All output quantities must be specified at the start of the calculation. In most cases, there is no way to retrieve information after the calculation ends if it was not specified from the start. There are several different ways of visualizing the results of a calculation. Most familiar to experimentalists is to save a given quantity at a single point in space so that this quantity can be plotted as a function of time, like a thermocouple temperature measurement. The namelist group THCP is used to specify "thermocouple" or point measurements.

A new feature of FDS version 2 is the animated isosurface (ISOF). An isosurface is a three dimensional contour of a scalar quantity. For example, a 300 °C temperature isosurface shows where the gas temperature is 300 °C. By default, the stoichiometric value of the mixture fraction is visualized via an isosurface. The user can choose several other quantities to visualize via an isosurface.

In order to visualize the flow patterns better, one can save planar slices of data, either in the gas or solid phases, by using the SLCF (SLiCe File) or BNDF (BouNDary File) namelist group. Both of these output formats permit one to animate these quantities in time.

For static pictures of the flow field, use the PL3D (PLot3D file) namelist group. Plot3D format is used by many CFD programs as a simple way to store specified quantities over the entire grid at one instant in time.

Finally, tracer particles can be injected into the flow field from vents or obstacles, and then viewed in Smokeview. Use the PART namelist group to control the injection rate, sampling rate and other parameters associated with particles. Note: unlike in FDS version 1, particles are no longer used to introduce heat into the flow, thus particles no longer are ejected automatically from burning surfaces.

The time increment between data dumps can be controlled by the user. For THCP, SLCF, ISOF and PART files, data will be written out to files every TWFIN/NFRAMES s. For BNDF files, data will be written out to files every TWFIN/NFRAMES/2 s. The parameter NFRAMES may be specified on the MISC line. For the PL3D format, data will be written out to a file every TWFIN/5 s. To change the sampling rate of any output file type, the parameter DTSAM should be set on any one line of that particular group. For example, if one desires that THCP data be written out every second, then the string DTSAM=1. should be written on one of the THCP lines. This one entry will dictate that all THCP data be written out every second. All of the other data files will be updated according to the default increment, or according to another prescription of DTSAM.

5.5.1 Point Measurements: The THCP Namelist Group

THCP lists thermocouple or other point measurements. Each line in this group consists of the coordinates of the point at which the measurement is to be recorded, XYZ, and a quantity to record, QUANTITY. These are either quantities associated with a given grid cell (see Table 1) or a surface cell (see Table 2). When prescribing a surface cell quantity, be sure to position the coordinates within a solid wall. It is not always obvious where the wall is since the grid does not always align with the input obstruction locations. The code will spit out an error message if a surface quantity is not embedded in a wall. If the orientation of the probe is not obvious, specify IOR appropriately. If the probe is facing in the positive x direction IOR=1, negative x direction IOR=-1, positive y IOR=2, negative IOR=-2, positive z IOR=3, and negative z IOR=-3. The thermocouples can be labeled with the character string LABEL for easier identification in the output file.

5.5.2 Animated Planar Slices: The SLCF Namelist Group

The SLCF ("slice file") namelist group allows the user to record various gas phase quantities (Table 1) at more than a single point. A "slice" refers to a subset of the whole domain. It can be a line, plane, or volume, depending on the values of XB. The sextuple XB indicates the boundaries of the region for which

values of QUANTITY are to be recorded every DTSAM s. XB is prescribed as in the OBST or VENT groups, with the possibility that 0, 2, or 4 out of the 6 values be the same to indicate a volume, plane or line, respectively. A handy trick is to specify, for example, PBY=5.3 instead of XB if it is desired that the entire plane $y = 5.3$ slicing through the domain be saved. PBX and PBZ control planes perpendicular to the x and z axes, respectively.

Slice file information is recorded in files labeled **casename_*n*.sf**, where n is the index of the slice file. A Fortran 90 routine **fds2ascii** will produce a test file from a line, plane or volume of data. See Section 5.5.8 for more details.

Figure 4 shows several snapshots of a vertical animated slice from Smokeview where the slice is colored according to gas temperature. Slice files are displayed by selecting the desired file from the Load/Unload menu in Smokeview.

Animated vectors are displayed using data contained in two or more slice files. The direction and length of the vectors are determined from the U, V and/or W velocity slice files. The vector colors are determined from the file (such as temperature) selected from the Load/Unload menu. Figure 5 shows a sequence of vector slices corresponding to the shaded temperature contours found in Figure 4.

5.5.3 Animated Boundary Quantities: The BNDF Namelist Group

The BNDF ("boundary file") namelist group allows the user to record surface quantities at all solid obstructions. The possible output quantities are listed in Table 2. As with the SLCF group, each quantity is prescribed with a separate BNDF line, and the output files are of the form **casename_*n*.bf**. No physical coordinates need be specified, however, just QUANTITY and DTSAM (if desired).

Note that BNDF files can become very large, so caution should be used in prescribing the time interval. One way to reduce the size of the output file is to "turn off" the drawing of boundary information on desired obstructions. On any given OBST line, if the string BNDF_BLOCK=.FALSE. is included, the obstruction will not be colored. To turn off all boundary drawing, set BNDF_DEFAULT=.FALSE. on the MISC line. Then individual obstructions can be turned back on with BNDF_BLOCK=.TRUE. on the appropriate OBST line.

Figure 6 shows several snapshots of a boundary file animation where the surfaces are colored according to their temperature. Boundary files are displayed by selecting the desired file from the Smokeview Load/Unload menu.

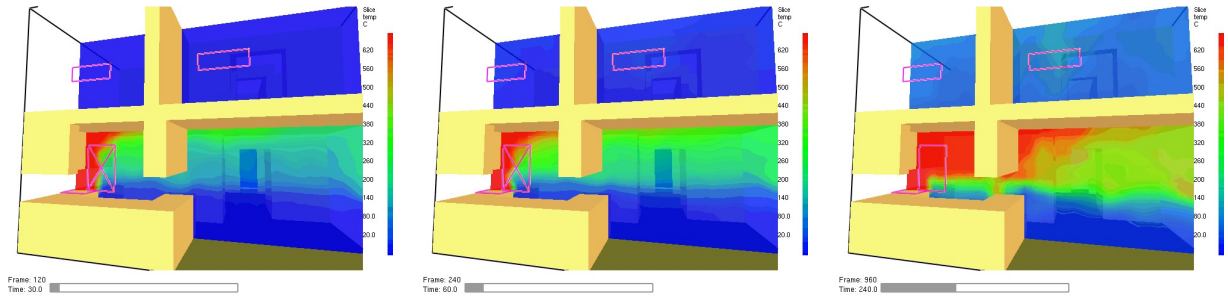
5.5.4 Animated Isosurfaces: The ISOF Namelist Group

The ISOF ("ISOsurface File") namelist group allows the user to save one or more values of a single gas phase quantity and render them as an animated sequence. The allowable quantities are DENSITY, TEMPERATURE, HRRPUV and MIXTURE_FRACTION. By default, the stoichiometric value of the mixture fraction is saved. For example, three different values of the temperature can be saved via the line

```
&ISOF QUANTITY='TEMPERATURE',VALUE(1)=50.,VALUE(2)=200.,VALUE(3)=500. /
```

where the values are in degrees C. Note that the isosurface output files **casename_*n*.iso** can become very large, so the user is urged to experiment with different sampling rates.

Figure 7 shows a snapshot of an isosurface file animation at several time steps for a fire (mixture fraction level=0.05) and a smoke layer interface (mixture fraction level = 0.001). Isosurface files are displayed by selecting the desired file from the Smokeview Load/Unload menu.

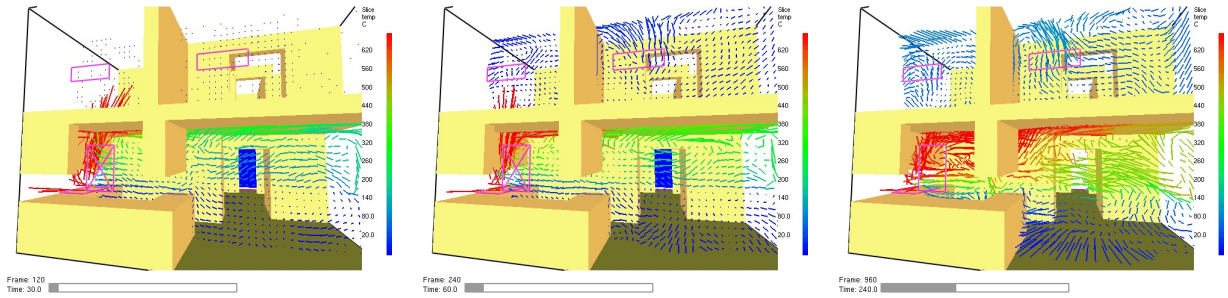


30 s 60 s 240 s

FIGURE 4: Slice file snapshots of shaded temperature contours in a vertical plane centered at the fire origin at t=(30, 60, 240) s after ignition. The data file for these contours was generated by adding the line

```
&SLCF PBY=1.5,QUANTITY='TEMPERATURE' /
```

to an FDS input file.



30 s 60 s 240 s

FIGURE 5: Vector slice file snapshots of shaded vector plots in a vertical plane centered at the fire origin at t=(30, 60, 240) s after ignition. The data files for these vector plots were generated by adding the line

```
&SLCF PBY=1.5,QUANTITY='TEMPERATURE',VECTOR=.TRUE. /
```

to an FDS input file.

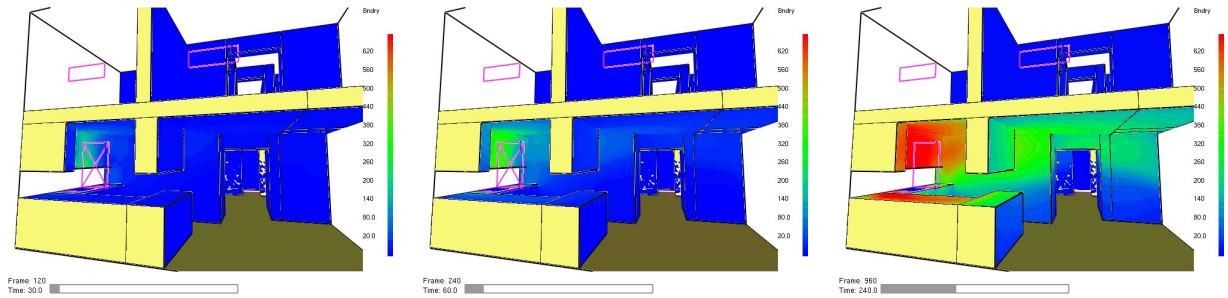


FIGURE 6: Boundary file snapshots of shaded wall temperatures at t=(30, 60, 240) s after ignition. The data file for these snapshots was generated by adding the line
`&BNDF QUANTITY='WALL_TEMPERATURE' /`
to an FDS input file.

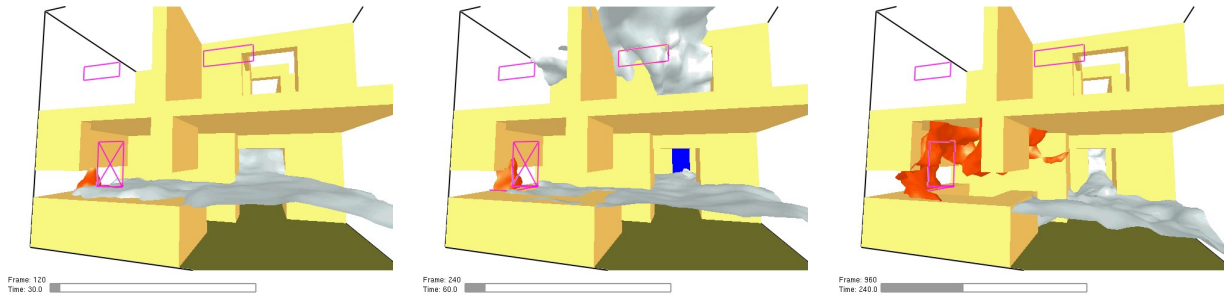


FIGURE 7: Isosurface file snapshots of mixture fraction levels at t=(30, 60, 240) s after ignition. The data file for these snapshots was generated by adding the line
`&ISOF QUANTITY='MIXTURE_FRACTION',VALUE(1)=0.05,VALUE(2)=0.001 /`
to an FDS input file.

5.5.5 Static Data Dumps: The PL3D Namelist Group

PL3D is the namelist group that defines how often and what quantities are to be output into files of Plot3D format. At most one PL3D line should be listed in the input file. Five quantities from Table 1 are written out to a file at one instant in time every DTSAM s. The default specification is

```
&PL3D DTSAM=TWFIN/5,QUANTITIES='TEMPERATURE',
      'U-VELOCITY','V-VELOCITY','W-VELOCITY','HRRPUV' /
```

It's best to leave the velocity components as is, because Smokeview uses them to draw velocity vectors. The first and fifth quantities can be changed with the parameter QUANTITIES(1) and QUANTITIES(5).

Data stored in Plot3D [6] files use a format developed by NASA and used by many CFD programs for representing simulation results. An FDS simulation will create a Plot3D file every DTSAM seconds. Plot3D data is visualized in three ways: as 2D contours, vector plots and isosurfaces. Figure 8a shows an example of a 2D Plot3D contour. Vector plots may be viewed if one or more of the *u*, *v* and *w* velocity components are stored in the Plot3D file. The vector length and direction show the direction and relative speed of the fluid flow. The vector colors show a scalar fluid quantity such as temperature. Figure 8b shows vectors in a doorway. Note the hot flow (red color) leaving the fire room in the upper part of the door and the cool flow (blue color) entering fire room in the lower part of the door. Figure 9 gives an example of isosurfaces for two different temperature levels. Plot3D data are stored in files with extension .q .

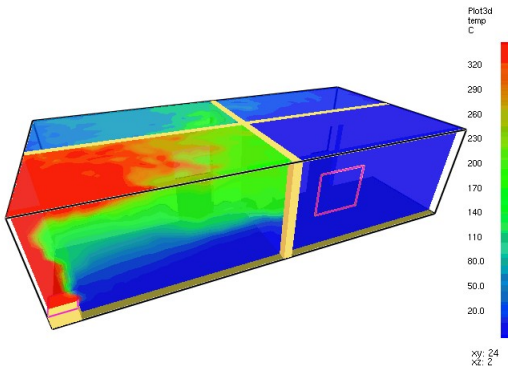
Plot3D files are more complicated to visualize than time dependent files such as particle, slice or boundary files. For example, only the transparency and color characteristics of a time file may be changed. With Plot3D files however, many attributes may be changed. One may view 2D contours along the X, Y and/or Z axis of up to six[6] different simulated quantities, view flow vectors and iso or 3D contours. Plot3D file visualization is initiated by selecting the desired file from the Load/Unload>Plot3D sub-menu and as with time files one may change color and transparency characteristics.

2D contours Smokeview displays a 2D contour slice midway along the Y axis by default when a Plot3D file is first loaded, To step the contour slice up by one grid cell along the Y axis, depress the space bar. Similarly to step the contour slice down by one grid cell along the Y axis, depress the - key. To view a contour along either the X or Z axis, depress the x or z keys respectively. Depressing the x, y or z keys while the contour is visible will cause it to be hidden. The Plot3D variable viewed may be changed by either depressing the p key or by selecting Show/Hide>Plot3D 3D Contours>Solution Variable.

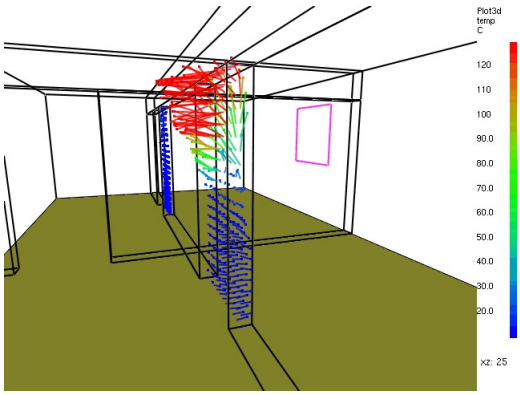
Iso-Contours Iso-contours also called 3D contours or level surfaces may be viewed by depressing the i key or by selecting Show/Hide>Plot3D 3D Contours.

Flow vectors If at least one velocity component is present in the Plot3D file then the v key may be depressed in order to view flow vectors. The length and direction of the vector indicates the flow direction and speed. The vector color indicates the value of the currently displayed quantity. A small dot is drawn at the end of the line to indicate flow direction. The vector lengths as drawn may be changed by depressing the a key. Vector plots may be very dense when the grid is finely meshed. The s key may be depressed in order to skip vectors. For example, all vectors are displayed by result. If the s is depressed then every other vector is skipped.

[6]The FDS software stores temperature, three components of velocity (denoted *u*, *v* and *w*) and heat release per unit volume. If at least one velocity component is stored in a Plot3D file then Smokeview adds speed to the Plot3D variable list.

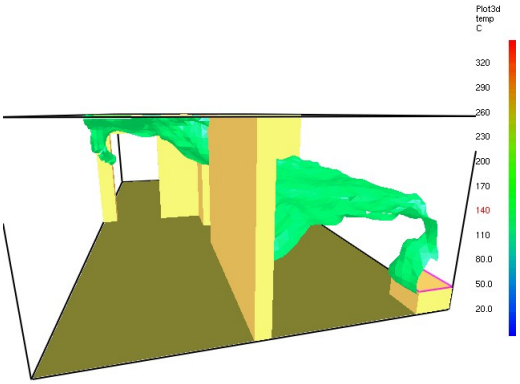


a) shaded 2D temperature contour plots in a vertical plane through the fire and a horizontal plane below the ceiling

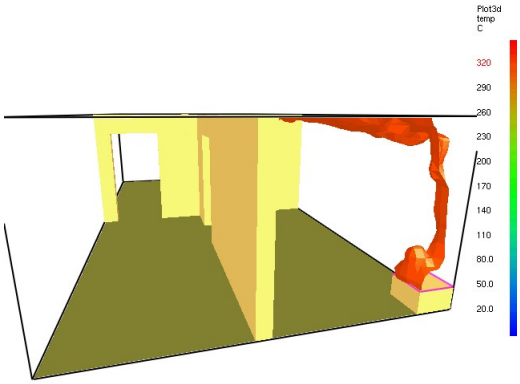


b) shaded temperature vector plot in a vertical plane through the doorway. The a key may be depressed to alter the vector sizes. The s key may be depressed to alter the number of vectors displayed.

FIGURE 8: Plot3D contour and vector plot examples.



a) isosurface at 140 °C



b) isosurface at 320 °C

FIGURE 9: Plot3D isocontour example.

5.5.6 Controlling Particles and Droplets: The PART Namelist Group

PART is the namelist group that is used to control the tracer particles and sprinkler droplets that are used in the calculation. Both particle and droplet data are dumped to the same file, called **casename.part**.

The following parameters allow the user to control the number of tracer particles and sprinkler droplets. Note that in version 2 of FDS, tracer particles play no role in the calculation. They are purely for visualization. It is important to monitor the number of particles used because the output file **casename.part** can become very large.

Traceer Particle Parameters

DTPAR Time increment between particle insertions in seconds. If more particles are desired, lower the input value of this parameter. (Default 0.05 s)

AGE Lifetime of a particle in seconds. If AGE is prescribed, a particle will be removed from the calculation AGE s after its insertion. This parameter can be used to reduce the number of particles that need to be tracked and save on CPU time. (Default infinity)

DTSAM Time increment between tracer particle (and sprinkler droplet) data dumps in seconds. These dumps add to a file called **casename.part** which can be used to produce an animation of the flow field. (Default TWFIN/NFRAMES)

NPSAM Sampling factor for the particle output file **casename.part**. This parameter can be used to reduce the size of the particle output file used to animate the simulation. (Default 1)

NPDIM Maximum number of particles tracked at any given time in the calculation. (Default 200000)

NPPS Number of particles per set. The maximum number of particles that can be output into the file **casename.part** every DTSAM seconds. (Default 100000)

QUANTITY A character string indicating which scalar quantity should be used to color the particles when viewed as an animation. Choices are 'TEMPERATURE' or 'HRRPUV'.

NIP Number of Initial Particles. This parameter allows the user to seed the domain with particles at the start of the simulation. (Default 0)

Sprinkler Droplet Parameters

DTSPAR Time increment between sprinkler droplet insertions (s). (Default 0.05 s)

NSPINS Number of sprinkler droplets inserted every DTSPAR seconds per active sprinkler. The number of water droplets introduced into the flow domain per sprinkler per second is thus NSPINS/DTSPAR. (Default NSPINS is 50)

DTSAM Incremental time between sprinkler droplet (and tracer particle) data dumps in seconds. (Default TWFIN/NFRAMES)

NSPSAM Sampling factor for the droplet output file **casename.part**. This parameter can be used to reduce the size of the output file **casename.part**. (Default 10)

NSPDIM Maximum number of droplets tracked at any given time in the calculation. (Default 200000)

`QUANTITY` A character string indicating which scalar quantity should be used to color the droplets when viewed as an animation. The only choice is `'DROPLET_DIAMETER'` in units of μm.

The particle file **casename.part** contains the locations of the tracer particles and sprinkler droplets used to visualize the flow field. Figure 10 shows several snapshots of a developing fire plume visualized using particles where the particles are colored according to the gas temperature. Sprinkler water droplets are (usually) colored blue.

5.5.7 Setting Data Bounds in Smokeview

Normally, Smokeview determines data bounds automatically when it loads data. Sometimes, however, it is desirable to override Smokeview's choice, for example so that several loaded files can have the same bounds. This allows for consistent shading when displaying several files simultaneously.

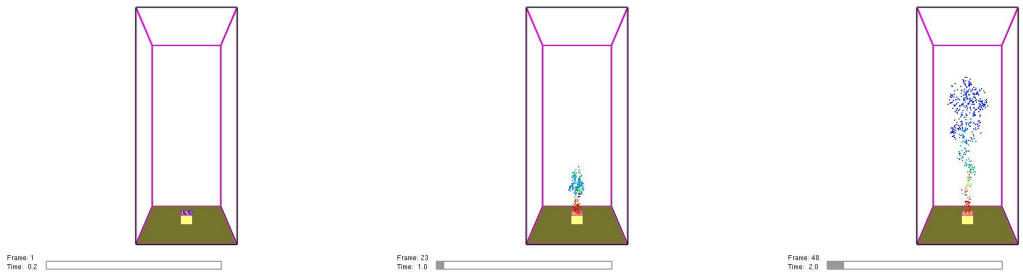
The `Set Bounds...` dialogue box is opened from the `Options` menu. Each file type in Figure 11 (slice, particle, Plot3D *etc*) has a set of "radio buttons" for selecting the variable type that data bound are to be applied too. These variable types are determined from the files generated by FDS and are automatically recorded in the `.smv` file. The data bounds are set in a pair of edit boxes. Check boxes adjacent to the edit boxes determine whether or not Smokeview should use the entered data bounds. The new bounds become activated after the data has been reloaded. The `Plot3D` portion of the dialog box uses radio buttons instead of check boxes in order to implement an additional loading option. Portions of the plot as illustrated in Figure 12 are not drawn when this option, labeled "chop", is selected.

5.5.8 Extracting Numbers from the Output Data Files

Often users want to present the results of calculations in some form other than those offered by Smokeview. In this case, there is a short Fortran 90 program called **fds2ascii.f**, with a PC compiled version called **fds2ascii.exe**. To run the program, just type

```
fds2ascii
```

at the command prompt. The user will be asked a series of questions about which type of output file to process, what time interval to time average the data, and so forth. A single file will be produced with the name **casename.dat**.



0.0 s 1.0 s 2.0 s

FIGURE 10: Particle snapshots of an isolated fire plume at t=(0, 1, 2) s. The particles were added to the calculation by specifying `PARTICLES=.TRUE.` on the `SURF` line defining the fire, and the coloring of the particles with the local gas temperature was achieved by adding the line
`&PART QUANTITY='TEMPERATURE' /`
to the FDS input file.

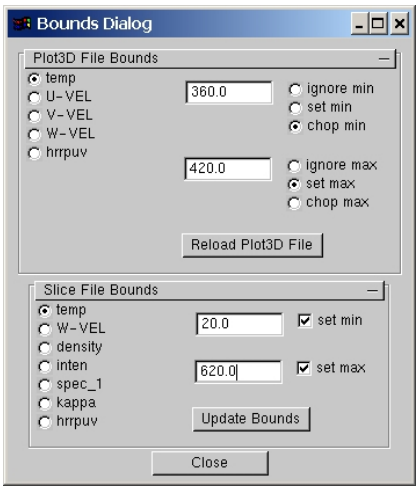


FIGURE 11: Bounds Dialogue Box. Select a variable, select a bounds type "check box/radio button", then enter a bound. Then either reload the data file or update the bounds.

Quantity	Description	Symbol	Units
DENSITY	density	ρ	kg/m^3
TEMPERATURE	gas temperature	T	C
U-VELOCITY	velocity component	u	m/s
V-VELOCITY	velocity component	v	m/s
W-VELOCITY	velocity component	w	m/s
VELOCITY	flow speed	$\sqrt{u^2+v^2+w^2}$	m/s
PRESSURE	perturbation pressure	$\tilde{p}$	Pa
HRRPUV	HRR Per Unit Volume	$\dot{q}'''$	kW/m^3
MIXTURE_FRACTION	mixture fraction	Z	kg/kg
DYNAMIC_VISCOSITY	dynamic viscosity	μ	kg/m/s
KINEMATIC_VISCOSITY	kinematic viscosity	μ/ρ	m^2/s
DIVERGENCE	divergence	$\nabla \cdot \mathbf{u}$	s^{-1}
WMPUV	Water Mass PUV	m_w'''	kg/m^3
WATER VAPOR	water vapor mass frac.	Y_w	kg/kg
oxygen	O_2 volume fraction	$X_{O_2}(Z)$	mol/mol
oxygen mass fraction	O_2 mass fraction	$Y_{O_2}(Z)$	kg/kg
fuel	fuel volume fraction	$X_F(Z)$	mol/mol
nitrogen	N_2 volume fraction	$X_{N_2}(Z)$	mol/mol
water vapor	H_2O volume fraction	$X_{H_2O}(Z)$	mol/mol
carbon dioxide	CO_2 volume fraction	$X_{CO_2}(Z)$	mol/mol
carbon monoxide	CO volume fraction	$X_{CO}(Z)$	ppm
soot volume fraction	soot volume fraction	$\rho Y_s/\rho_s$	ppm
soot density	smoke particulate concentration	ρY_s	μg/m^3
extinction coefficient	light extinction coef.	$K = K_m \rho Y_s$	1/m
visibility	visibility distance	$S = C/K$	m

TABLE 1: Gas Phase Output Quantities for THCP**,** SLCF **or** PL3D**. Note that lower case quantities are appropriate only for calculations involving the mixture fraction. If individual species are listed via** SPEC **namelist lines, the Quantity for mass and volume fractions are** [SPECIES_ID] **and** [SPECIES_ID]_VF**. The quantities** water vapor **and** WATER VAPOR **denote the volume fraction of water vapor generated by combustion and the *mass* fraction of water vapor from evaporated sprinkler droplets, respectively. See Section 7.8 for a discussion of** extinction coefficient **and** visibility**.**

Quantity	Description	Symbol	Units
RADIATIVE_FLUX	radiative flux	$\dot{q}''_r$	kW/m^2
CONVECTIVE_FLUX	convective flux	$\dot{q}''_c$	kW/m^2
HEAT_FLUX	total heat flux	$\dot{q}''_r + \dot{q}''_c$	kW/m^2
WALL_TEMPERATURE	wall temperature	T_s	C
BACK_WALL_TEMPERATURE	inside wall temperature		C
BURNING_RATE	mass loss rate per unit area	$\dot{m}''_f$	kg/m^2/s
WMPUA	Water Mass Per Unit Area	$\dot{m}''_w$	kg/m^2
WCPUA	Water Cooling Per Unit Area	$\dot{q}''_w$	kW/m^2

TABLE 2: Solid Phase Output Quantities for THCP and BNDF.

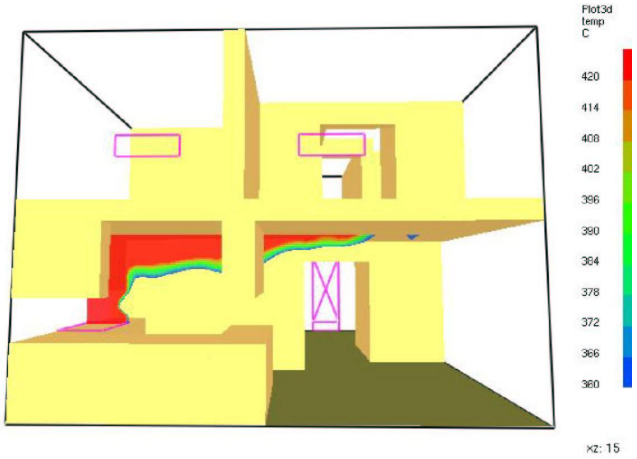


FIGURE 12: Ceiling jet visualization created by "chopping data" below a specified temperature by using the Bounds Dialogue Box as illustrated in Figure 11.

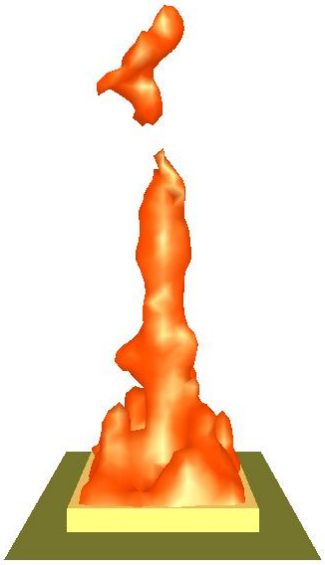


FIGURE 13: Instantaneous snapshot of a pool fire simulation. The isosurface denotes the interface between fuel-rich and fuel-lean regions of the fire. The square burner is 0.4 m by 0.4 m, the dimensions of the computational domain are 0.6 m by 0.6 m by 1.2 m.

6 Sample Calculations

This section documents some calculations performed with version 2 of the FDS model. For each case, the parameters used to perform the calculations will be presented and explained in terms of the physical descriptions of the experiments. A complete listing of the input parameters is included below.

6.1 Pool Fire

The first example is intended to demonstrate the physical model of a fire that has been newly implemented in FDS 2. The input file for the isolated plume calculation is included in Fig. 14. More detail about the input parameters may be found in Section 5. A brief description is provided here.

```
&HEAD CHID='pool2',TITLE='Single Pool Fire' /
```

The first line of the file provides a character string `'pool2'` that will be common to all output files. The `TITLE` is just for identifying the job. Note that single quotes should surround all character string input, and that character strings are case sensitive.

```
&GRID IBAR=24,JBAR=24,KBAR=48 /
&PDIM XBAR0=-.30,XBAR=0.30,YBAR0=-.30,YBAR=0.30,ZBAR=1.2 /
```

The dimensions of the numerical grid are given by the integers `IBAR`, `JBAR` and `KBAR`. The dimensions of the computational domain are given by `XBAR0, XBAR`, `YBAR0, YBAR` and `ZBAR`, all in meters.

```
&TIME TWFIN=10. /
```

The time of the simulation is given by `TWFIN` in seconds.

```
&HEAD CHID='pool2',TITLE='Single Pool Fire' /
&GRID IBAR=24,JBAR=24,KBAR=48 /
&PDIM XBAR0=-.30,XBAR=0.30,YBAR0=-.30,YBAR=0.30,ZBAR=1.2 /
&TIME TWFIN=10. /
&SURF ID='burner',HRRPUA=1000. /
&OBST XB=-.20,0.20,-.20,0.20,0.00,0.05,SURF_IDS='burner','INERT','INERT' /
&VENT CB='XBAR' ,SURF_ID='OPEN' /
&VENT CB='XBAR0',SURF_ID='OPEN' /
&VENT CB='YBAR' ,SURF_ID='OPEN' /
&VENT CB='YBAR0',SURF_ID='OPEN' /
&VENT CB='ZBAR' ,SURF_ID='OPEN' /
&SLCF PBY=0.,QUANTITY='TEMPERATURE',VECTOR=.TRUE. /
&SLCF PBY=0.,QUANTITY='HRRPUV' /
&SLCF PBY=0.,QUANTITY='MIXTURE_FRACTION' /
&BNDF QUANTITY='HEAT_FLUX' /
```

FIGURE 14: Input file for single pool fire calculation.

```
&SURF ID='burner',HRRPUA=1000. /
&OBST XB=-.20,0.20,-.20,0.20,0.00,0.05,SURF_IDS='burner','INERT','INERT' /
```

The SURF line defines the fire as essentially a boundary condition applied at the top of the solid obstruction that is defined by the OBST line. Thus, as far as the calculation is concerned, the burner is a rectangular solid out of whose top fuel is ejected and burns. The total heat release rate of the fire will be (1000 kW/m^2) $\times$ (0.16 m)^2 = 160 kW. There are no walls or ceiling in the calculation, as indicated by the five VENT lines that call for the planes $x = -0.3$ (XBAR0), $x = 0.3$ (XBAR), $y = -0.3$ (YBAR0), $y = 0.3$ (YBAR), and $z = 1.2$ (ZBAR) to be OPEN, that is, open to the atmosphere.

```
&SLCF PBY=0.0,QUANTITY='TEMPERATURE',VECTOR=.TRUE. /
```

These lines designate that the gas temperatures in the plane $y = 0.0$ be saved, along with the three components of the velocity.

6.2 Fire Spread in a Townhouse

The next example demonstrates the spread of a fire in a two-level townhouse, starting with a small fire that starts on a stove top. The overall geometry of the house, plus a snapshot from the simulation is shown in Fig. 15. The effort in designing the simulation lies in both prescribing the geometry and in describing the solid and gas phase reactions. The former task is relatively straightforward, although somewhat tedious. The latter task is more difficult because it often involves parameters that are unknown or at the very least grossly approximated. These parameters will be described first.

For simplicity, all of the walls have been designated as wood via the entry on the MISC line,

```
SURF_DEFAULT='PINE'
```

Of course, for actual simulations many different types of materials may be included. The thermal properties of PINE are given as

```
&SURF ID                    = 'PINE'
      HEAT_OF_VAPORIZATION = 2500.
      KS                    = 0.14
      ALPHA                 = 8.3E-8
      DELTA                 = 0.020
      TMPIGN                = 390. /
```

These properties are taken from Ref. [7]. KS is the thermal conductivity (k_s) in units of W/m/K, ALPHA is the thermal diffusivity ($k_s/(\rho_s c_s)$) in units of m^2/s, DELTA (δ) is the thickness of the material in units of m, TMPIGN is the ignition temperature in units of degrees C. These parameters govern the heat up of the wood. Once the ignition temperature is reached, the wood begins to pyrolyze according to the prescribed HEAT_OF_VAPORIZATION, which is expressed in units of kJ/kg. In words, every 2500 kJ of energy that is not conducted through the wall is used to pyrolyze a kg of fuel. Note that for wood there is a wide range of values for the HEAT_OF_VAPORIZATION. Plus, wood is a char former, meaning that a single value of the HEAT_OF_VAPORIZATION may not be appropriate. Research is continuing in this area.

In addition to specifying the wall material, one must also specify the characteristics of the gas phase reaction. Unlike the material properties, only one set of gas phase reaction parameters may be prescribed via the MISC namelist group; in this case, REACTION='WOOD'. The parameters corresponding to this entry are as follows

```
&REAC ID='WOOD'
      SOOT_YIELD = 0.01
      NU_O2      = 3.7
      NU_CO2     = 3.4
      NU_H2O     = 3.1
      MW_FUEL    = 87.
      EPUMO2     = 8850. /
```

These parameters are taken from a paper by Kashiwagi and co-workers [8]. It is assumed that the very complex set of reactions describing the combustion of wood can be reduced to a single, infinitely-fast reaction between a hydrocarbon fuel whose molecular weight, MW_FUEL, is 87 g/mol. Each fuel molecule consumed produces 3.7 molecules of O_2, 3.4 molecules of CO_2, and 3.1 molecules of H_2O. Energy is released at a rate of 8850 kJ/kg of oxygen consumed. In addition, 1 % of the fuel mass is converted into soot. There is no soot growth or oxidation model present; it is assumed that the soot generated in the reaction is transported along with the other combustion products throughout the house.

In practice, the parameters described above are stored in a text file along with various other SURFace conditions and REACtion characteristics. The user would invoke these parameters via keywords in the data file, **townhouse2.data**:

```
&HEAD CHID='townhouse2',TITLE='Townhouse Kitchen Fire Example' /
&GRID IBAR=48,JBAR=64,KBAR=48 /
&PDIM XBAR=6.4,YBAR=8.0,ZBAR=4.8 /
&TIME TWFIN=300.0 /
&MISC REACTION='WOOD',SURF_DEFAULT='PINE',
      DATABASE='c:\nist\fds\database2.data' /

&SURF ID='BURNER',HRRPUA=2000. /
&VENT XB=0.0,0.3,1.00,1.50,0.9,0.9,SURF_ID='BURNER' /

&VENT XB=3.4,4.2,0.00,0.00,3.2,4.0,SURF_ID='OPEN' /
&VENT XB=1.2,2.0,0.00,0.00,3.2,4.0,SURF_ID='OPEN' /

                        .
                        .

&OBST XB=0.00,1.10,0.00,0.25,1.50,2.30 /
&OBST XB=0.00,0.20,0.25,2.75,1.50,2.30 /
&OBST XB=4.00,5.06,4.25,4.37,2.50,4.40,BLOCK_COLOR='BLUE',T_REMOVE=240. /

                        .
                        .

&SLCF PBY=1.5,QUANTITY='TEMPERATURE',VECTOR=.TRUE. /
&SLCF PBZ=2.2,QUANTITY='TEMPERATURE' /

&BNDF QUANTITY='HEAT_FLUX',DTSAM=0.3 /
&BNDF QUANTITY='WALL_TEMPERATURE' /
```

Note that it is assumed that the user has put the DATABASE file in the directory **c:\nist\fds** and called it **database2.data**. This file can be put anywhere and called anything, as long as its name and location are conveyed to the program via the DATABASE entry on the MISC line. It can't be assumed that the program knows it is to open a database file, even if it happens to be in the same directory as the data file **townhouse2.data**.

There are two uses of the namelist group VENT. The first is to establish the location and size of the initial fire. Keep in mind that although the whole house is made of wood, something has to start the fire. The fire is prescribed over a rectangular area atop the stove with a Heat Release Rate Per Unit Area, HRRPUA, of 2000 kW/m^2. The total heat release rate (initially) should be 0.3 m $\times$ 0.5 m $\times$ 2000 kW/m^2 or 300 kW.

The second use of the VENT namelist group is to designate openings to the atmosphere, in this case windows and doors located on the exterior boundary of the computational domain. The entry SURF_ID='OPEN' on these VENT lines means that the windows are, literally, open. Note that this boundary condition can only be used at exterior boundaries. Doorways and windows within the computational domain must be carved out using OBSTruction lines.

The geometry of the house is specified via the OBST namelist lines. These lines establish all of the solid obstructions within the house, in this case, these obstructions are mostly walls. Note that there is no need to explicitly introduce external walls in the file – the boundary of the overall rectangular domain is assumed to be a solid wall, unless designated otherwise. Internal walls must be explicitly specified with OBST lines. Each OBST line represents a rectangular block, thus the walls and the floor separating the first and second

levels must be entered as separate chunks. It is permissible to overlap these blocks, but it is not permissible to open a door within a wall with a VENT statement. A wall with an open door in it must be specified with 3 OBST lines – a block to the left and right of the door opening, plus a block over the door. Note that obstructions can be removed using the parameter T_REMOVE, a time in seconds after which the obstruction is to disappear. This parameter is used to simulate the opening of a door or window. See Section 7.3 for details.

Finally, the user must decide before starting the calculation which results to save in various output files. In this case, by default an isosurface of the fire will be produced automatically, as well as five Plot3D files at five equally spaced instants in time. In this example, the user has also prescribed slice planes of temperature at $y = 1.5$ and $z = 2.2$. The attribute VECTOR=.TRUE. instructs the program to also save (in separate files) the three components of velocity in the same plane so that an animation of the flow vectors, colored by temperature, can be produced. This attribute need not be repeated on other SLCF lines that designate the same plane, but must be repeated on other SLCF lines that designate different planes.

Information about bounding surfaces is saved via the BNDF namelist group. In this case, the WALL_TEMPERATURE and the HEAT_FLUX to the wall will be saved and shown as an animation. Note that it is often prudent to save boundary files less often than other file formats because the BNDF output files can become quite large. The parameter DTSAM is used for this purpose, and it applies to both BNDF lines.

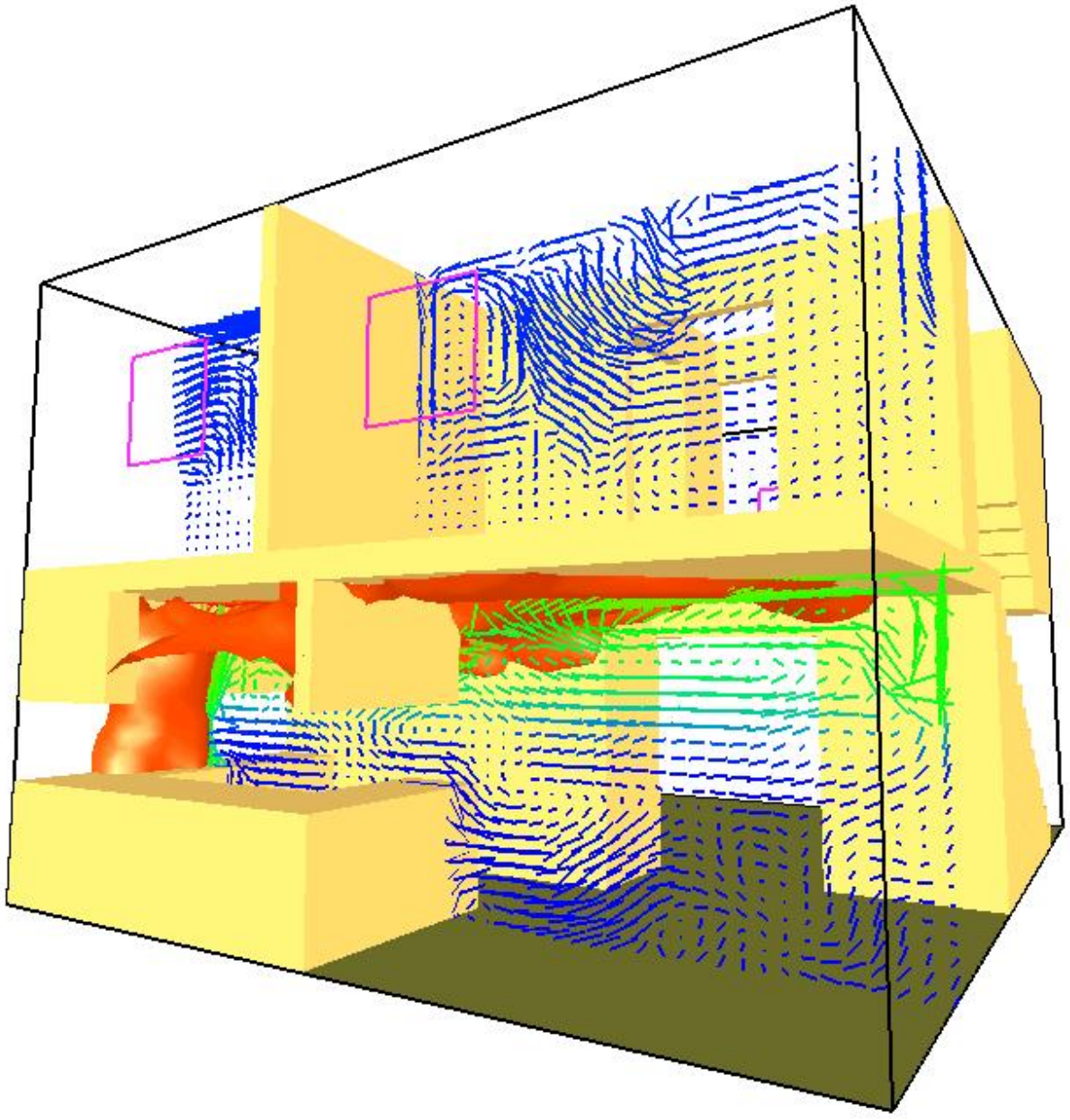


FIGURE 15: Snapshot of townhouse simulation showing an isosurface of the reaction zone and a vertical slice of flow vectors.

7 Special Features

The following sections describe some special features of the FDS model that are not usually invoked for engineering applications. Some of these features represent current research efforts and are to be considered fragile in the sense that unexpected results can occur if the user is not familiar with numerical combustion.

7.1 Stopping and Restarting Calculations

Normally, a simulation consists of a sequence of events starting from ambient conditions. However, there are occasions where the user might want to stop a calculation, make a few limited adjustments, and then restart the calculation from that point in time. To do this, the user must first bring the calculation to a halt gracefully by creating a file called **casename.stop** in the directory where the output files are located. FDS checks for the existence of this file at each time step, and if it finds it, gracefully shuts down the calculation after first creating a final Plot3D file and a file called **casename.restart**. To restart a job, the file **casename.restart** should be designated by the parameter `RESTART_FILE` which needs to be added to the `MISC` line of the data file controlling the continued job. If the user desires that the old job simply be continued, and all the output files be appended with new data, then use the same "casename" when designating the `RESTART_FILE`. For example, suppose that the job whose casename is "plume" is halted by the user who creates a dummy file called **plume.stop** in the directory where all the output files are being created. To restart this job from where it left off, add `RESTART_FILE='plume.restart'` to the `MISC` line of **plume.data**. The existence of a restart file with the same character ID as the original job tells the code to save the new data in the same files as the old. If it is desired that the data from the restarted job be saved in new output files, rename the job by changing the character ID (`CHID`), for example, `CHID='plume2'` on the `HEAD` line.

There may be times when the user wants to save restart files periodically during a run as insurance against power outages or system crashes. If this is the case, at the start of the original run set `DTCORE=50.` on the `MISC` line to save restart files every 50 s, for example. The default for `DTCORE` is infinity, meaning no restart files will be created unless the user gracefully stops a job by creating a dummy file called **casename.stop**.

Note that between job stops and restarts, the user cannot make major changes in the calculation like adding or removing vents and obstructions. The changes are limited to those parameters which will not instantly alter the existing flow field. Since the restart capability has been used infrequently at NIST, it should be considered a fragile construct, and the user should make sure by examining the output data that no sudden or unexpected events occur during the stop and restart.

7.2 Stretching the Grid: The `TRNX`, `TRNY` and/or `TRNZ` Namelist Groups

By default the grid cells that fill the computational domain are uniform in size. However, it is possible to specify that the cells be non-uniform in one or two of the three coordinate directions. For a given coordinate direction, *x*, *y* or *z*, a function can be prescribed that maps the uniformly-spaced computational grid to a non-uniformly spaced physical grid. Take the *x* direction as an example. A function $x = f(\xi)$ will map the uniformly-spaced Computational Coordinate (CC) ($\texttt{XBAR0} \le \xi \le \texttt{XBAR}$) to the Physical Coordinate (PC) ($\texttt{XBAR0} \le x \le \texttt{XBAR}$). The function has three mandatory constraints: it must be monotonic, it must map `XBAR0` to `XBAR0`, and it must map `XBAR` to `XBAR`. The default transformation function is $f(\xi) = \xi$ (uniform grid). If uniform gridding is desired in the *x* direction, then nothing need be specified, and no `&TRNX` lines should be written. The same is true for the *y* and *z* directions.

Two types of transformation functions are allowed. The first, and simplest, is a piecewise linear function. Figure 16 gives an example of a piecewise linear transformation, in this case applied to the *y* coordinate

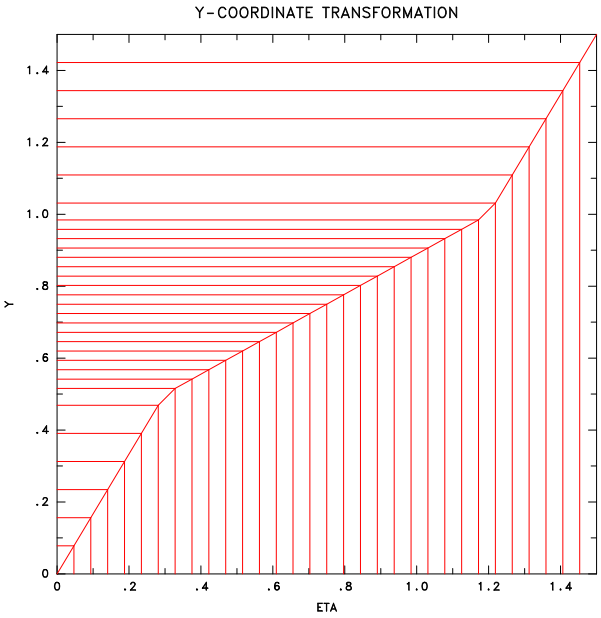


FIGURE 16: Piecewise Linear Transformation.

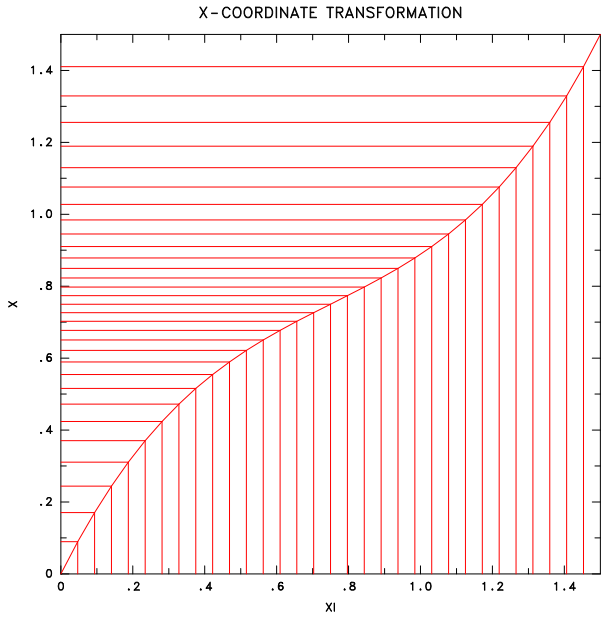


FIGURE 17: Polynomial Transformation.

direction. The graph indicates how the uniformly-spaced computational grid (horizontal axis) is mapped to the non-uniformly-spaced physical grid (vertical axis). In this case, the function is made up of straight line segments connecting user-prescribed points (CC,PC). Note that the points should be given in increasing order. The parameters used in this example are

```
&TRNY CC=0.30,PC=0.50 /
&TRNY CC=1.20,PC=1.00 /
```

The parameter CC refers to the Computational Coordinate, located on the horizontal axis; PC is the Physical Coordinate, located on the vertical axis. In this example, the uniformly-spaced grid cells between 0 and 0.3 on the horizontal axis are mapped to larger, but still uniformly-spaced, cells on the vertical axis between 0 and 0.5. Then the segment between 0.3 and 1.2 on the horizontal axis is mapped to a segment between 0.5 and 1.0 on the vertical axis. Finally, the segment between 1.2 and 1.5 on the horizontal axis is mapped to the segment between 1.0 and 1.5 on the vertical axis. The slopes of the line segments indicate whether the grid is being stretched (slopes greater than 1) or shrunk (slopes less than 1). The tricky part about this process is that the user usually has a desired shrinking/stretching strategy for the physical coordinate (vertical axis), and must work backwards to determine what the corresponding points are in computational space (horizontal axis).

The second type of transformation is a polynomial function whose constraints are of the form

$$\frac{d^{\texttt{IDERIV}} f(\texttt{CC})}{d\xi^{\texttt{IDERIV}}} = \texttt{PC}$$

Figure 17 gives an example of a polynomial transformation. The parameters used in this example are

```
&TRNX IDERIV=0,CC=0.75,PC=0.75 /
&TRNX IDERIV=1,CC=0.75,PC=0.50 /
```

which correspond to the constraints $f(0.75) = 0.75$ and $\frac{df}{d\xi}(0.75) = 0.5$, or, in words, the function maps 0.75 into 0.75 and the slope of the function at $\xi = 0.75$ is 0.5 . Note that these constraints are optional. The two

mandatory constraints $f(\texttt{XBAR0}) = \texttt{XBAR0}$ and $f(\texttt{XBAR}) = \texttt{XBAR}$ need not, and should not, be prescribed. The reason for this is that the mandatory constraints are already built into the linear system of equations that is solved to yield the coefficients of the *n*th order polynomial. In the above example, two optional constraints plus the two mandatory constraints will yield a unique cubic polynomial $f(\xi) = c_0 + c_1\xi + c_2\xi^2 + c_3\xi^3$. More optional constraints lead to higher order polynomial functions. The monotonicity of the function will be checked by the program and an error message will be produced if it is not monotonic.

The vertical and horizontal lines on Figs. 16 and 17 indicate how the uniformly spaced computational grid on the horizontal axis is transformed into a non-uniformly spaced physical grid on the vertical axis. The slope of the transformation function indicates the degree to which the computational cells will be stretched (slope greater than 1) or shrunk (slope less than 1). Note that shrinking cells in one region will necessarily lead to stretching cells elsewhere. When one or two coordinate directions are transformed, the aspect ratio of the grid cells in the 3D mesh will vary. To be on the safe side, transformations that alter the aspect ratio of cells beyond 2 or 3 should be avoided. Keep in mind that the large eddy simulation technique is based on the assumption that the numerical grid should be fine enough to allow the formation of eddies that are responsible for the mixing. In general, eddy formation is limited by the largest dimension of a grid cell, thus shrinking the grid in one or two directions may not necessarily lead to a better simulation if the third dimension is large. Also note that transformations, in general, reduce the efficiency of the computation, with two coordinate transformations impairing efficiency more than a transformation in one coordinate direction.

7.3 Creating or Removing Obstructions; Opening or Closing Vents

In many fire scenarios, the opening or closing of a door or window can lead to dramatic changes in the course of the fire. Sometimes these actions are taken intentionally, sometimes as a result of the fire. Within the framework of an FDS calculation, these actions are represented by the creation or removal of solid obstacles, or the opening or closing of exterior vents.

The user can remove or create a solid obstruction by entering a time via the parameter `T_REMOVE` or `T_CREATE` on the `OBST` line that defines the obstruction in question. For example, the line

```
&OBST XB=...,SURF_ID='whatever',T_CREATE=39. /
```

will instruct the code to create the given obstruction 39 s after the start of the simulation. Note that once created, the blockage can never be removed. Similarly for a blockage that is removed, it can never be created again. One additional feature is to time the removal or creation of an obstruction with a thermally-responsive device; in FDS the device being a `HEAT` detector. If one sets either `T_REMOVE` or `T_CREATE` to a *negative* value, then the absolute value of that number is the index of the `HEAT` detector that controls the action. For example, the lines

```
&OBST XB=...,SURF_ID='whatever',T_REMOVE=-2 /
                    .
                    .
&HEAT XYZ=...,RTI=...,ACTIVATION_TEMPERATURE=87. /
&HEAT XYZ=...,RTI=...,ACTIVATION_TEMPERATURE=110. /
```

will cause the given obstruction to be removed when the *second* `HEAT` detector listed in the input file activates[7].

The removal and creation of an obstruction can be used to model the opening of a door that connects one part of the domain to another. If one wants to open or close a window or door to the outside of the computational domain (*i.e.* the atmosphere), a similar set of parameters can be used on the `VENT` line. For example, the line

[7] Note that the parameter `IDEVICE` from FDS 1 is no longer used to link `HEAT` detectors to the removal of obstructions.

```
&VENT XB=...,SURF_ID='OPEN',T_CLOSE=39. /
```

instructs the code to close a vent to the exterior of the domain at 39 s. The line

```
&VENT XB=...,SURF_ID='OPEN',T_OPEN=-2 /
                    .
                    .
&HEAT XYZ=...,RTI=...,ACTIVATION_TEMPERATURE=87. /
&HEAT XYZ=...,RTI=...,ACTIVATION_TEMPERATURE=110. /
```

instructs the code to open an initially closed vent to the outside when the second HEAT detector activates.

One additional feature associated with a VENT is to specify a time or a thermal device that controls when the VENT's SURF_ID is to be applied. For example, suppose it is desired that when the second HEAT detector activates, a fan should be turned on. Set

```
&SURF ID='FAN',VOLUME_FLUX=5. /
                    .
                    .
&VENT XB=...,SURF_ID='FAN',T_ACTIVATE=-2 /
                    .
                    .
&HEAT XYZ=...,RTI=...,ACTIVATION_TEMPERATURE=87. /
&HEAT XYZ=...,RTI=...,ACTIVATION_TEMPERATURE=110. /
```

Note that T_ACTIVATE has the same functionality as T_OPEN or T_CLOSE, only it is specified for a VENT that does not have the OPEN attribute. Note that none of these control parameters should be applied to a MIRROR.

7.4 Extra Species

Normally when one specifies a fire via either HRRPUA or HEAT_OF_VAPORIZATION, the mixture fraction combustion model is applied. A single scalar variable, Z, represents the state of the combustion process from pure fuel ($Z = 1$) to pure air ($Z = 0$). The major reactants and products of combustion – fuel, O_2, CO_2, H_2O, N_2, CO and soot – are all pre-tabulated functions of the mixture fraction, Z. In other words, the value of Z in any given grid cell determines the mass fraction of all the gases listed. The user-defined stoichiometry listed under the REAC namelist group is used to generate the table associating the mass fractions with Z. The user need not, *and should not*, explicitly list the reactants and products of combustion.

Suppose however that gases are introduced into the domain that are neither reactants nor products of combustion. This gas can be tracked separately from the mixture fraction via a second scalar transport equation[8]. In fact, there need not be any fire at all – the FDS code can be used to transport a mixture of non-reacting ideal gases.

The namelist group SPEC is used to specify each additional species. Each SPEC line should include at the very least the name of the species via a character string called (ID). Next, if the ambient (initial) mass fraction of the gas is something other than 0, then the parameter MASS_FRACTION_0 is used to specify it. Several gases that can be included in a calculation are listed in Table 3. The physical properties of these gases are known and need not be specified. However, if a desired gas is not included in Table 3, its molecular

[8]Often an extra gas introduced into a calculation is the same as a product of combustion, like water vapor from a sprinkler or carbon dioxide from an extinguisher. These gases will be tracked separately, thus water vapor generated by the combustion is tracked via the mixture fraction variable and water vapor generated by evaporating sprinkler droplets is tracked via its own transport equation. In the case of sprinklers, the user need not specify WATER VAPOR as an extra species, it will be done automatically.

weight MW must be specified in units of g/mol. In addition, if a DNS calculation is being performed, either the Lennard-Jones potential parameters σ (SIGMALJ) and ε/k (EPSILONKLJ) should be specified; or the VISCOSITY (kg/m/s), THERMAL_CONDUCTIVITY (W/m/K), and DIFFUSION_COEFFICIENT (m^2/s) between the given species and the background species should be specified.

```
&SPEC ID='ARGON',MASS_FRACTION_0=0.1,MW=40. /
```

There are three types of boundary conditions for the listed species. Reference to a given species is via its place in the input file, so, for example, the second listed species is N=2. If a simple no-flux condition is desired at a solid wall, do not set anything. If the mass fraction of the Nth species is to be some value at a forced flow boundary, set MASS_FRACTION(N) equal to the desired mass fraction on the appropriate SURF line. If the mass flux of the Nth species is desired, set MASS_FLUX(N) instead of MASS_FRACTION(N). If MASS_FLUX(N) is set, no VEL should be set. It will automatically be calculated based on the mass flux. The mass flux is in units of $kg/m^2/s$.

Use TAU_MF(N) or RAMP_MF(N) to control the ramp-ups for either the mass fraction or mass flux of species N. The mass fraction of species N at the surface is given by

$$Y_N(t) = Y_N(0) + f(t)\,(Y_N - Y_N(0))$$

where $Y_N(0)$ is the ambient mass fraction of species N (MASS_FRACTION_0 in the Nth SPEC namelist line is used to prescribe $Y_N(0)$), Y_N is the desired mass fraction to which the function $f(t)$ is ramping (MASS_FRACTION(N) specified in the SURF line is used to prescribe Y_N). The function $f(t)$ is either a tanh, t^2, or user defined function. For a user defined function, indicate the name of the ramp function with RAMP_MF(N), a character string (see Time Dependent Boundary Conditions in Section 5.3.1).

As an example, the lines

```
&SPEC ID='ARGON',MASS_FRACTION_0=0.1,MW=40. /
&SPEC ID='HELIUM' /
       .
       .
&SURF ID='INLET',MASS_FRACTION(2)=0.2,VEL=-0.3,TAU_MF(2)=0.5,TAU_V=0.5 /
```

specify that ARGON and HELIUM, in addition to the default BACKGROUND_SPECIES='AIR', be included in the calculation. At the INLET, a mixture of helium (0.2 by mass), argon (0.1 by mass because nothing different is specified), and air (0.7 by mass making up the rest) flows out at a velocity of 0.3 m/s *into* the flow domain. The mass fraction of helium and the velocity are both ramped up according to the function $\tanh(t/0.5)$.

7.5 Finite-Rate or Premixed Combustion

By default, FDS assumes that the fire is essentially an infinitely-fast reaction between fuel and oxygen, and this reaction is not dependent on the surrounding gas temperature. It also assumes that the reaction zone is an infinitely thin sheet with fuel on one side and oxygen on the other. If a finite-rate or a pre-mixed reaction is desired, the following steps must be taken by the user:

1. It is strongly recommended that finite-rate reactions be invoked only when FDS is running in DNS mode. Set DNS=.TRUE. on the MISC line. Note: one may attempt to use the finite-rate reaction scheme in an LES calculation, but because the temperature in a large scale calculation is smeared out over a grid cell, some of the reaction parameters may need to be modified to account for the lower temperatures.

TABLE 3: Optional Gas Species [9]

Species	Mol. Wgt. (g/mol)	σ (Å)	k/ε (K)
AIR	29	3.711	78.6
CARBON DIOXIDE	44	3.941	195.2
CARBON MONOXIDE	28	3.690	91.7
HELIUM	4	2.551	10.22
METHANE	16	3.758	148.6
NITROGEN	28	3.798	71.4
OXYGEN	32	3.467	106.7
PROPANE	44	5.118	237.1
WATER VAPOR	18	2.641	809.1

2. The BACKGROUND_SPECIES on the MISC line is normally set to be 'NITROGEN'.

3. The namelist group SPEC is used to specify each additional species. Do not enter a SPEC line for the background species. Each SPEC line should include the name of the species (ID) and its ambient (initial) mass fraction, MASS_FRACTION_0. Several gases that can be included in a calculation are listed in Table 3. The physical properties of these gases are known and need not be specified. However, if a desired gas is not included in Table 3, its molecular weight MW must be specified in units of g/mol. In addition, if a DNS calculation is being performed, either the Lennard-Jones potential parameters σ (SIGMALJ) and ε/k (EPSILONKLJ) should be specified; or the VISCOSITY (kg/m/s), THERMAL_CONDUCTIVITY (W/m/K), and DIFFUSION_COEFFICIENT (m^2/s) between the given species and the background species should be specified. If the listed species is to be consumed or generated by the finite-rate reaction, its stoichiometric coefficient, NU, needs to be specified. Note that the background species is not allowed to participate in the reaction.

4. Read Section 7.4 for a description of the boundary conditions for the gas species.

5. The REAC namelist group is used to designate the fuel and the reaction rate parameters.

   FUEL Character string indicating which of the listed optional gas species is the fuel.

   BOF Pre-exponential factor in one-step chemical reaction in units of cm^3/mole/s.

   EACT Activation energy for one-step chemical reaction in units of kcal/mole.

   XNO Exponent for oxygen concentration in one-step chemical reaction. (Default 1.)

   XNF Exponent for fuel concentration in one-step chemical reaction. (Default 1.)

   DELTAH The effective heat of combustion for one-step chemical reaction in units of kJ/kg. (Default 40,000 kJ/kg)

7.6 Liquid Fuels

An experimental feature in the code is to specify the phase of the fuel as solid, liquid, or gas. Only PHASE='LIQUID' on the SURF line directs the code to do something different than what is normally done. For a liquid, the evaporation rate of the fuel is governed by the Clausius-Clapeyron equation. The

only drawback of this approach is that the fuel gases burn regardless of any ignition source. Thus, if PHASE='LIQUID' is specified, the fuel will begin burning at once. An example of a liquid fuel is

```
&SURF ID='METHANOL'
      HEAT_OF_VAPORIZATION=1101.
      PHASE='LIQUID'
      DELTA=0.01
      KS=0.20
      ALPHA=8.85E-8
      TMPIGN=65. /
```

7.7 Suppression by Water (Mixture Fraction Model Only)

Modeling suppression of a fire by a water spray is challenging because the relevant physical mechanisms occur at length scales smaller than a single grid cell. In the gas phase, flames are extinguished due to lowered temperatures and dilution of the oxygen supply. With the mixture fraction model, only the dilution effect is accounted for because it is assumed that fuel and oxygen burn regardless of the temperature. The user controls a parameter called X_O2_LL that can be put on the REAC line. It is the volume fraction of oxygen below which combustion cannot occur. Its default value is 0.15 .

For the solid phase, water reduces the fuel pyrolysis rate by cooling the fuel surface and also changing the chemical reactions that liberate fuel gases from the solid. If the fuel has been assigned a HEAT_OF_VAPORIZATION, there is no need to set any additional suppression parameters. It is assumed that water impinging on the fuel surface will take energy away from the pyrolysis process and thereby reduce the burning rate of the fuel. If the fuel has been assigned a HRRPUA (Heat Release Per Unit Area), it falls upon the user to also specify a parameter that governs the suppression of the fire by water. An empirical way to account for fire suppression by water is to characterize the reduction of the pyrolysis rate in terms of an exponential function. The local mass loss rate of the fuel is expressed in the form

$$\dot{m}''_f(t) = \dot{m}''_{f,0}(t)\, e^{-\int k(t)\, dt} \tag{4}$$

Here $\dot{m}''_{f,0}(t)$ is the user-prescribed burning rate per unit area when no water is applied and k is a function of the local water mass per unit area, m''_w, expressed in units of kg/m^2.

$$k(t) = \texttt{E\_COEFFICIENT}\, m''_w(t) \quad \mathrm{s}^{-1} \tag{5}$$

The parameter E_COEFFICIENT must be obtained experimentally, and it is expressed in units of m^2/kg/s. Usually, this type of suppression algorithm is invoked when the fuel is complicated, like a cartoned commodity.

Another parameter used in water suppression calculations is the fuel POROSITY which is the fraction of water that does not cascade down the side of the fuel package, either because of absorption or because it is trapped within the interior of the fuel. Both POROSITY and E_COEFFICIENT are 0 by default and both are prescribed on the SURF line.

7.8 Visibility

If one is performing a fire calculation using the mixture fraction approach (default in FDS 2), the smoke is tracked along with all other major products of combustion. The most useful quantity for assessing visibility in a space is the *light extinction coefficient*, K [10]. The intensity of monochromatic light passing a distance L through smoke is attenuated according to

$$I/I_0 = e^{-KL} \tag{6}$$

The light extinction coefficient, K, is a product of the density of smoke particulate, ρY_s, and a mass specific extinction coefficient that is fuel dependent

$$K = K_m \, \rho Y_s \tag{7}$$

Estimates of visibility through smoke can be made by using the equation

$$S = C/K \tag{8}$$

where C is a nondimensional constant characteristic of the type of object being viewed through the smoke, *i.e.* $C = 8$ for a light-emitting sign and $C = 3$ for a light-reflecting sign [10]. Since K will vary from point to point in the domain, the visibility S will as well. Keep in mind that FDS can only track smoke whose production rate and composition are dictated by the user. Predicting either is beyond the capability of the present version of the model.

The user controls three parameters related to smoke production and visibility; each parameter is input on the `REAC` line. The first parameter is `SOOT_YIELD`, which is the fraction of fuel mass that is converted to soot. The second parameter is called the `MASS_EXTINCTION_COEFFICIENT`, and it is the K_m in Eq. (7). The default value is 7600 m^2/kg, a value suggested for flaming combustion of wood and plastics. The third parameter is called the `VISIBILITY_FACTOR`, the constant C in Eq. (8). It is 3 by default.

The slice, Plot3D or thermocouple output quantity `extinction coefficient` is K. The visibility S is output via the keyword `visibility`. Note that each is tied to the mixture fraction formulation of combustion.

7.9 Fires and Flows in the Outdoors

Simulating a fire in the outdoors is not much different than a fire indoors, but there are several useful parameters that can easily be invoked. First, it is easy to prescribe the wind with a realistic increase in velocity with altitude. By default the velocity (wind) profile at any vent is a top hat, but the parameter `PROFILE` on the `SURF` line can yield other profiles. For example, `PROFILE='PARABOLIC'` produces a parabolic profile with `VEL` being the maximum velocity, and `'ATMOSPHERIC'` produces a typical atmospheric wind profile of the form $u = u_0(z/z_0)^p$. If an atmospheric profile is prescribed, also prescribe `Z0` for z_0 and `PLE` for p. `VEL` specifies the reference velocity u_0.

Another useful parameter for outdoor simulations is the temperature lapse rate of the atmosphere. Typically, in the first few hundred meters of the atmosphere, the temperature decreases several degrees Celsius per kilometer. These few degrees are important when considering the rise of smoke since the temperature of the smoke decreases rapidly as it rises. `DT0DZ` is the lapse rate of the atmosphere in units of °C/m. This need only be set for outdoor calculations where the height of the domain is tens or hundreds of meters. The default value of `DT0DZ` is -g/c_p ≈-0.0097 °C/m.

Figure 18 gives an example of an outdoor fire scenario, with the corresponding input data file given in Figure 19. The tanks are built from rectangular blocks forming a cylindrical shape. The fire is specified on the top of one tank and a wind is prescribed at one of the computational boundaries.

7.10 2D and Axially-Symmetric Calculations

The governing equations solved in FDS are written in terms of a three dimensional Cartesian coordinate system. However, the user can invoke a two dimensional Cartesian or two dimensional cylindrical (axisymmetric) calculation by setting `JBAR=1` on the `&GRID` line, and for axisymmetry replacing the parameter `XBAR` with `RBAR` on the `&PDIM` line. No grid stretching is allowed when doing a 2D calculation. No boundary conditions should be set at the planes $y =$ `YBAR0` or $y =$ `YBAR`, nor at $x =$ `RBAR0` in an axisymmetric calculation in which `RBAR0`=0. For better visualizations, the difference of `YBAR` and `YBAR0` should

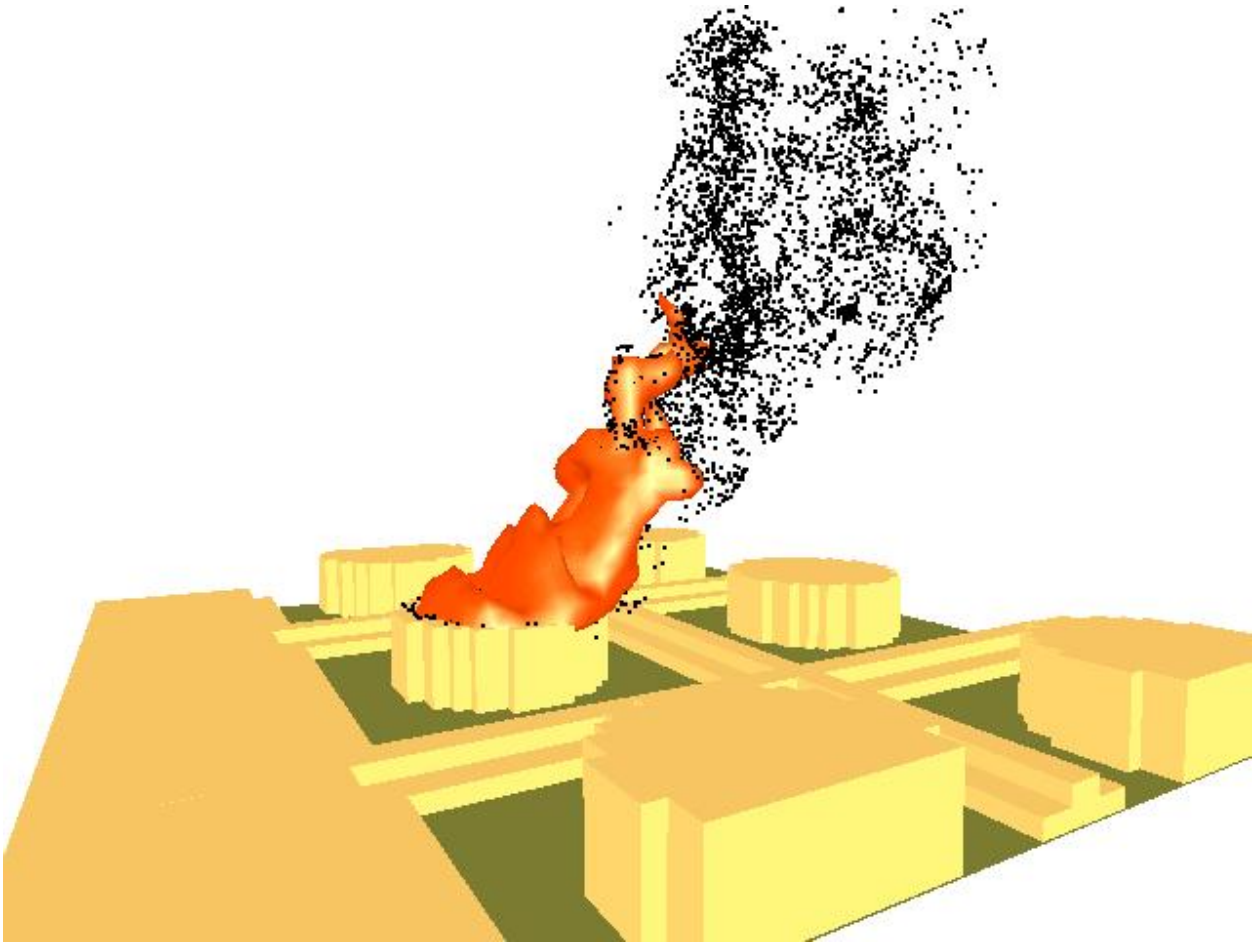


FIGURE 18: Instantaneous snapshot of a large oil fire simulation. The domain is 384 m by 384 m by 288 m. Each tank is 84 m in diameter and 27 m tall. The grid is uniform with 6 m grid cells.

```
&HEAD CHID='tankfarm2',TITLE='Japanese Oil Storage Tank Farm' /
&GRID IBAR=64,JBAR=64,KBAR=48 /
&PDIM XBAR0=64.,XBAR=448.,YBAR0=192.,YBAR=576.,ZBAR=288. /
&TIME TWFIN=60. /

&MISC DT0DZ=0. /

&SURF ID='WIND',VEL=-5.,PROFILE='ATMOSPHERIC',Z0=27.,PLE=0.15  /
&SURF ID='FIRE',HRRPUA=1500.,PARTICLES=.TRUE. /

&OBST XB=  0.,138.,  0.,768.,  0.,  9. /
&OBST XB=138.,144.,  0.,768.,  0.,  6. /
&OBST XB=144.,150.,  0.,768.,  0.,  3. /
                     .
                     .
                     .
&OBST XB=174.,180.,372.,396.,  0., 27.,SURF_IDS='FIRE','INERT','INERT' /
&OBST XB=180.,186.,360.,408.,  0., 27.,SURF_IDS='FIRE','INERT','INERT' /
                     .
                     .
&VENT CB='XBAR0',SURF_ID='WIND' /
&VENT CB='YBAR' ,SURF_ID='OPEN' /
&VENT CB='YBAR0',SURF_ID='OPEN' /
&VENT CB='XBAR' ,SURF_ID='OPEN' /
&VENT CB='ZBAR' ,SURF_ID='OPEN' /

&SLCF PBY=384.,QUANTITY='TEMPERATURE',VECTOR=.TRUE. /
&SLCF PBY=384.,QUANTITY='HRRPUV' /
&SLCF PBY=384.,QUANTITY='MIXTURE_FRACTION' /

&BNDF QUANTITY='HEAT_FLUX' /

&PL3D DTSAM=60. /
```

FIGURE 19: Input data file for tankfarm scenario.

be small so that the Smokeview rendering will appear to be in 2D. An example of an axially-symmetric helium plume is shown in Fig. 20 with the corresponding input data file in Fig. 21.

7.11 Restoring the Baroclinic Vorticity

There is an approximation made when solving for the pressure where it is assumed that

$$\nabla \cdot \frac{1}{\rho} \nabla \tilde{p} = \frac{1}{\rho_0} \nabla^2 \tilde{p} \tag{9}$$

The consequence of this approximation is that the vorticity generated due to the non-alignment of the density and pressure gradients, or the baroclinic torque, is neglected. For most large scale applications, the assumption is justified by the fact that the vorticity generated by buoyancy is the dominant source of vorticity. By neglecting the baroclinic torque the solution of the elliptic partial differential equation obtained by taking the divergence of the momentum equation is greatly simplified. However, an option exists in the code to restore the baroclinic torque by decomposing the relevant term in the pressure equation

$$\nabla \cdot \frac{\nabla \tilde{p}}{\rho} = \nabla \cdot \frac{\nabla \tilde{p}}{\bar{\rho}} + \nabla \cdot \left(\frac{1}{\rho} - \frac{1}{\bar{\rho}} \right) \nabla \tilde{p} \tag{10}$$

and evaluating the second term on the right hand side with values of pressure from the previous time step. The expression $\bar{\rho}$ is an average density, equal to $2\rho_{\min}\rho_{\max}/(\rho_{\min}+\rho_{\max})$. To make this correction, the user must simply include the statement

```
BAROCLINIC=.TRUE.
```

on the `MISC` line. In a DNS calculation (`DNS=.TRUE.`), the correction is made by default. However, for an LES calculation (default), the correction must be explicitly invoked by the user. The cost of the correction is not prohibitive, and the user is encouraged to try calculations with and without the correction to determine if its inclusion is warranted.

7.12 Fine-Tuning the Radiation Transport Model

There are several ways to improve the performance of the Finite Volume Method in solving the radiation transport equation (RTE), all of which increase the computational time. The solver has two main modes of operation – a gray gas model (default) and a wide band model. Modifications to these models can be made via a namelist group called `RADI`. If running in gray gas mode, the user can increase the number angles from the default 100 with the integer parameter `NUMBER_RADIATION_ANGLES`. The frequency of calls to the radiation solver can be reduced from every 3 time steps with integer `TIME_STEP_INCREMENT`. The increment over which the angles are updated can be reduced from 5 with the integer `ANGLE_INCREMENT`. Briefly, if `TIME_STEP_INCREMENT` and `ANGLE_INCREMENT` are both set to 1, the radiation field will be completely updated in a single time step, but the cost of the calculation will increase significantly.

If the optional wide band model is desired, then the user need only set `WIDE_BAND_MODEL=.TRUE.`. It is recommended that this option only be used when the fuel is relatively non-sooting.

Note that for now, the radiation solver can only be used with the mixture fraction combustion model, which is the default. Note also that it is possible to turn off the radiation transport solver (saving roughly 20 % in CPU time) by adding the statement `RADIATION=.FALSE.` to the `MISC` line. For isothermal calculations, the radiation is turned off automatically. If burning is taking place and the user turns off radiation, then `RADIATIVE_FRACTION` from the `REAC` line will be subtracted from the total heat release rate. This radiated energy will completely disappear from the calculation.

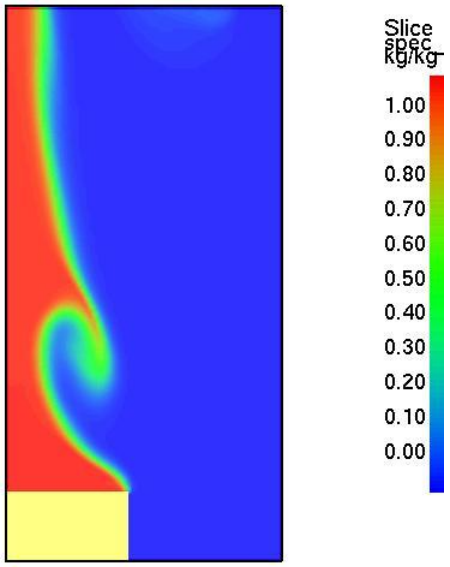


FIGURE 20: Instantaneous snapshot of an axially-symmetric helium plume.

```
&HEAD CHID='helium',TITLE='Axisymmetric Helium Plume' /
&GRID IBAR=72,JBAR=1,KBAR=144 /
&PDIM RBAR=0.08,YBAR=0.001,ZBAR=0.16 /
&TIME TWFIN=10.0 /
&MISC DNS=.TRUE.,ISOTHERMAL=.TRUE. /
&SPEC ID='HELIUM'  /
&SURF ID='HELIUM',VEL=-0.673,MASS_FRACTION(1)=1.0,TAU_MF(1)=0.3 /
&VENT CB='XBAR' ,SURF_ID='OPEN' /
&VENT CB='ZBAR' ,SURF_ID='OPEN' /
&OBST XB= 0.0,0.036, 0.00, 0.01,0.00,0.02,SURF_IDS='HELIUM','INERT','INERT' /
&PL3D DTSAM=2.0,QUANTITIES(1)='DENSITY',QUANTITIES(5)='HELIUM' /
&SLCF PBY=0.000,QUANTITY='DENSITY',VECTOR=.TRUE. /
&SLCF PBY=0.000,QUANTITY='HELIUM' /
```

FIGURE 21: Input data file for helium plume calculation.

7.13 Defying Gravity

Most users of FDS assume that the acceleration of gravity points toward the negative end of the z axis, or more simply, downward. However, the user may want to change the direction of gravity to model a sloping roof or tunnel, for example. To do this, specify the gravity vector on the `MISC` line with a triplet of numbers of the form `GVEC=0.0,0.0,-9.81` (units are m/s^2). This is the default, but it can be changed to be any direction.

Note: if sprinklers are specified, the gravity vector must not be changed. Much of the logic governing the trajectories of water droplets over solid objects assumes that gravity points in the negative z direction.

7.14 Isothermal and Salt Water Simulations

Although FDS was designed specifically for fire simulations, it can be used for other fluid flow simulations that do not include fire or heat addition of any kind. The first thing the user ought to do in situations like this is set `ISOTHERMAL=.TRUE.` on the `MISC` line. This logical parameter indicates that the calculation does not involve any change in temperature and no radiation heat transfer, thus reducing the number of equations that must be solved, simplifying those that are, and reducing the computation time.

A valuable model validation exercise is to simulate the mixing of fresh and salt water. These types of experiments have traditionally been performed to mimic the movement of smoke in a building. Although FDS is rarely used for this purpose now, the capability still exists in the code to handle fresh/salt water interactions. To invoke this feature, the user needs to include the statements

```
&MISC ISOTHERMAL=.TRUE.,BACKGROUND_SPECIES='FRESH WATER',DENSITY=1000.,
      VISCOSITY=1.0E-3,SC=1. /
&SPEC ID='SALT WATER',DENSITY=1052.,VISCOSITY=1.0E-3 /
```

in the input file, along with appropriate boundary conditions for the salt water. Note that the names `'FRESH WATER'` and `'SALT WATER'` have no particular meaning as far as the code is concerned. What matters is the designation of `DENSITY` and `VISCOSITY` for both, plus a global Schmidt number `SC`. Without the designation `DNS=.TRUE.`, the calculation will be performed as a Large Eddy Simulation (LES), in which case the `VISCOSITY` will serve as a lower bound for the Smagorinsky viscosity, and the Schmidt number will be used to relate the viscosity to the material diffusivity.

7.15 Non-rectangular Geometry

The efficiency of FDS is due to the simplicity of its numerical grid. However, there are situations in which certain geometric features do not conform to the rectangular grid. In these cases, the user must construct the curved geometry using rectangular obstructions, a process sometimes called "stair stepping". The concern by many users is that the stair stepping will change the flow pattern near the wall. To lessen the impact of stair stepping on the flow field near the wall, the user can prescribe the parameter

```
SAWTOOTH=.FALSE.
```

on each `OBST` line that makes up the stair stepped obstruction. The effect of this parameter is to prevent vorticity from being generated at sharp corners, in effect smoothing out the jagged steps that make up the obstruction. This is not a complete solution of the problem, but it does provide a simple way of ensuring that the flow field around a non-rectangular obstruction is not inhibited by extra drag created at sharp corners.

8 Conclusion

The equations and numerical algorithm described in this document form the core of an evolving fire model. As research into specific fire-related phenomena continues, the relevant parts of the model can be improved. Because the model was originally designed to analyze industrial scale fires, it can be used reliably when the fire size is specified and the building is relatively large in relation to the fire. In these cases, the model predicts flow velocities and temperatures to within 20 % of the experimental measurements. Improvements in version 2 of FDS address many issues associated with large fires in relatively small spaces, but the uncertainties of these types of calculations is greater due both to the lack of input data for material properties and combustion chemistry and to greater numerical error in combustion and radiation transport. Version 2 takes a step closer to a truly predictive model where less of the fire behavior is predetermined by the user.

Any user of the numerical model must be aware of the assumptions and approximations being employed. There are two issues for any potential user to consider before embarking on calculations. First, for both real and simulated fires, the growth of the fire is very sensitive to the thermal properties of the surrounding materials. Second, even if all the material properties are known, the physical phenomena of interest may not be simulated due to limitations in the model algorithms or numerical grid. Except for those few materials that have been studied to date at NIST, the user must supply the thermal properties of the materials, and then validate the performance of the model with experiments to ensure that the model has the necessary physics included. Only then can the model be expected to predict the outcome of fire scenarios that are similar to those that have actually been tested.

References

[1] K.B. McGrattan, H.R. Baum, R.G. Rehm, G.P. Forney, J.E. Floyd, and S. Hostikka. Fire Dynamics Simulator (Version 2), Technical Reference Guide. Technical Report NISTIR 6783, National Institute of Standards and Technology, Gaithersburg, Maryland, August 2001.

[2] Thomas Boutell. *CGI Programming in C & Perl*. Addison-Wesley Publishing Co., Reading, Massachussets, 1996.

[3] C.M. Fleischmann and F.F. Chen. Radiant Ignition of Upholstered Furniture. In *Proceedings of the International Conference on Engineered Fire Protection Design*, 2001. Society of Fire Protection Engineers.

[4] U.O. Koylu and G.M. Faeth. Carbon Monoxide and Soot Emissions from Liquid-Fueled Buoyant Turbulent Diffusion Flames. *Combustion and Flame*, 87:61–76, 1991.

[5] T. Ma. Numerical Simulation of an Axi-symmetric Fire Plume: Accuracy and Limitations. Master's thesis, University of Maryland, 2001.

[6] Pamela P. Walatka and Pieter G. Buning. PLOT3D User's Manual, version 3.5. NASA Technical Memorandum 101067, NASA, 1989.

[7] J.G. Quintiere. *Principles of Fire Behavior*. Delmar Publishers, Albany, New York, 1998.

[8] S.J. Ritchie, K.D. Steckler, A. Hamins, T.G. Cleary, J.C. Yang, and T. Kashiwagi. The Effect of Sample Size on the Heat Release Rate of Charring Materials. In *Proceedings of the 5th International Symposium on Fire Safety Science*, pages 177–188. International Association For Fire Safety Science, 1996.

[9] R.C. Reid, J.M. Prausnitz, and B.E. Poling. *Properties of Gases and Liquids*. McGraw-Hill, Inc., New York, 4th edition, 1987.

[10] G.W. Mulholland. *SFPE Handbook*, chapter Smoke Production and Properties. National Fire Protection Association, Quincy, Massachusetts, 2nd edition, 1995.

A Compiling the Source Code for FDS

If a compiled version of **fds2** exists for the machine on which the calculation is to be run, there is no need to compile the code, and this section may be ignored. For example, the file **fds2.exe** is the compiled program for a Windows-based PC; thus PC users do not need a Fortran compiler and do not need to compile the source code. For machines for which an executable has not been compiled, the user must compile the code. Fortran 90/95 and C compilers are needed for compilation. Table 4 lists the files that make up the source code. The files with suffix ".f" contain fixed form Fortran 90 instructions conforming to the ANSI and ISO standards, with the exception of those instructions that pass information between the C and Fortran routines. The source files should be compiled in the order in which they are listed in Table 4 because some routines are dependent on others. For UNIX users, **Makefiles** for various platforms have been provided that will assist in the compilation. Compiler options differ from platform to platform. The only compiler options that need to be prescribed are those that optimize performance, specify fixed format Fortran, and define the C to Fortran argument passing convention.

TABLE 4: Source Code Files

File Name	Description
mods.f	Global arrays and constants
misc.f	Miscellaneous routines
radi.f	Radiation
read.f	Read input parameters
divg.f	Compute the flow divergence
pres.f	Spatial discretization of pressure (Poisson) equation
mass.f	Mass equation(s) and thermal boundary conditions
init.f	Initialize variables and Poisson solver
pois.f	Poisson solver
velo.f	Momentum equations
part.f	Particle transport
sprk.f	Sprinkler activation and spray dynamics
dump.f	Dumps output data into files
isob.c	Routine for computing isosurface triangles
main.f	Main program, calls subroutines

B Output File Formats

The output from the code consists of the file **casename.out**, plus various data files that will be described below. Most of these output files are written out by the routine **dump.f**, and can easily be modified to accommodate various plotting packages.

B.1 Diagnostic Output

A file called **casename.out** is created if the program is run in the background and the file **casename.out** is prescribed as standard out. The file **casename.out** consists of a list of the input parameters, and an accounting of various important quantities, including CPU usage. Typically, diagnostic information is printed out every 100 time steps

```
                          .
                          .

200 CPU/step:    9.381 s, Total CPU:    32.04 min
    Time step: 0.15545 s, Total time:   32.54 s
    Max divergence: 0.23E+00 at ( 28, 30,  7)
    Max CFL number: 0.80E+00 at ( 39, 36, 47)
    Total Heat Release Rate:       4751471.032 kW
    Net Radiative Loss:             674522.107 kW
    Radiative Fraction:                  0.142
    Fire Resolution Index:               0.054
    Number of Tracer Particles:           5664

300 CPU/step:    9.489 s, Total CPU:    47.86 min
    Time step: 0.15654 s, Total time:   47.72 s
    Max divergence: -.30E+00 at ( 32, 35, 30)
    Max CFL number: 0.85E+00 at ( 44, 34, 46)
    Total Heat Release Rate:       4537902.900 kW
    Net Radiative Loss:             577468.871 kW
    Radiative Fraction:                  0.127
    Fire Resolution Index:               0.042
    Number of Tracer Particles:           6669

                          .
                          .
```

The number on the left indicates the number of the time step. The quantity `CPU/step` is the amount of CPU time required to complete a time step; `Total CPU` is the amount of CPU time elapsed since the start of the run; `Time step` is the time step size of the simulation; `Total time` is the time of the simulation; `Max divergence` is the maximum value of the function $\nabla \cdot \mathbf{u}$ and is used as a diagnostic when the flow is incompressible (*i.e.* no heating); and `Max CFL number` is the maximum value of the CFL number. The `Net Radiative Loss` is the amount of the total energy that is being radiated to the boundaries, the `Radiative Fraction` is the corresponding fraction of the total. The `Fire Resolution Index` is an indicator of well resolved the calculation is – it is the fraction of the ideal stoichiometric value of the mixture fraction that is being used in the calculation. Finally, `Number of Tracer Particles` indicates how many passive particles are being tracked at that time.

Following the completion of a successful run, a summary of the CPU usage per subroutine is listed. This is useful in determining where most of the computational effort is being put.

B.2 Plot3D Data

Presently, field data is output every DTSAM seconds in a format used by the graphics package **Plot3D**. The Plot3D data sets are single precision (32 bit reals), whole and unformatted. Note that there is blanking, that is, blocked out data points are not plotted. The grid data is written out to a file called **casename.xyz**

```
WRITE(LU13) IBAR+1,JBAR+1,KBAR+1
WRITE(LU13) (((X(I),I=0,IBAR),J=0,JBAR),K=0,KBAR),
.           (((Y(J),I=0,IBAR),J=0,JBAR),K=0,KBAR),
.           (((Z(K),I=0,IBAR),J=0,JBAR),K=0,KBAR),
.     (((IBLK(I,J,K),I=0,IBAR),J=0,JBAR),K=0,KBAR)
```

where X, Y and Z are the coordinates of the cell corners, and IBLK is an indicator of whether or not the cell is blocked. If the point (X,Y,Z) is completely embedded within a solid region, then IBLK is 0. Otherwise, IBLK is 1. The flow variables are written to a file called **casename_****_**.q**, where the stars indicate a time at which the data is output. The file is written with the lines

```
WRITE(LU14) IBAR+1,JBAR+1,KBAR+1
WRITE(LU14) ZERO,ZERO,ZERO,ZERO
WRITE(LU14) ((((QQ(I,J,K,N),I=0,IBAR),J=0,JBAR),K=0,KBAR),N=1,5)
```

The five channels N=1,5 are by default the temperature (C), the *u*, *v* and *w* components of the velocity (m/s), and the heat release rate per unit volume (kW/m^3). Alternate variables can be specified with the input parameter QUANTITIES under the PL3D namelist group. Note that the data is interpolated at cell corners, thus the dimensions of the Plot3D data sets are one larger than the dimensions of the computational grid.

Smokeview can display the Plot3D data. In addition, the Plot3D data sets can be read into whatever graphics routine the user has available, as long as the format of the data is specified. This particular format is very convenient, and recognized by a number of graphics packages, including AVS, IRIS Explorer and Tecplot [9].

B.3 Thermocouple Data

Temperature (or other scalar quantity) data at discrete points specified in the input file under the namelist group THCP is output in comma delimited format in a file called **casename_tc.csv**. The format of the file is as follows

```
NTC
    , LABEL(1)   , LABEL(2)   , ... , LABEL(NTC)
TIME, QUANTITY(1), QUANTITY(2), ... , QUANTITY(NTC)
s   , UNITS(1)   , UNITS(2)   , ... , UNITS(NTC)
T   , TC(1)      ,    TC(2)   , ... , TC(NTC)
                .
                .
                .
```

[9] With the exception of Smokeview, the graphics packages referred to in this document are not included with the source code, but are commercially available.

where NTC is the number of thermocouples, LABEL is a user-defined tag, QUANTITY is the physical quantity represented, UNITS the units, T the time of the dump, and TC(I) the value at the Ith thermocouple, in the order that they are listed in the input file. The files can be imported into **Microsoft Excel** or other spread sheet programs.

B.4 Sprinkler Data

If sprinklers are prescribed in the calculation, then a file called **casename_spk.csv** will be generated. The file lists the temperature of each sprinkler link.

```
NSPR
TIME,TEMP ,TEMP ,...,TEMP
s   ,C    ,C    ,...,C
T   ,TL(1),TL(2),...,TL(NSPR)
                .
                .
                .
```

where NSPR is the number of sprinklers, T is the time of the dump, TL(I) is the link temperature of the Ith sprinkler. The data will be ordered the same way as the sprinklers are listed in the input file.

B.5 Heat Release Rate

The total heat release rate of the fire is reported in the comma delimited file **casename_hrr.csv**. The file consists of three columns, the first column containing the time in seconds, the second contains the total heat release rate in units of kW, the third contains the amount of the total energy that is radiated to the boundaries.

B.6 Mixture Fraction State Relations

The functional dependence of the mass fraction of the reactants and products of combustion on the mixture fraction is reported in the comma delimited file **casename_state.csv**. The file consists of nominally 10 columns, the first column containing the mixture fraction, the rest the mass fractions of the various gases.

B.7 Slice Files

The slice files defined under the namelist group SLCF are named **casename_*n*.sf** (*n*=01,02...), and are written out unformatted, unless otherwise directed. These files are written out from **dump.f** with the following lines:

```
WRITE(LUSF) QUANTITY
WRITE(LUSF) SHORT_NAME
WRITE(LUSF) UNITS
WRITE(LUSF) I1,I2,J1,J2,K1,K2
WRITE(LUSF) TIME
WRITE(LUSF) (((QQ(I,J,K),I=11,I2),J=J1,J2),K=K1,K2)
        .
        .
        .
WRITE(LUSF) TIME
WRITE(LUSF) (((QQ(I,J,K),I=11,I2),J=J1,J2),K=K1,K2)
```

QUANTITY, SHORT_NAME and UNITS are character strings of length 30. The sextuple (I1,I2,J1,J2,K1,K2) denotes the bounding grid cell nodes. The sextuple indices correspond to grid cell nodes, or corners, thus the entire grid would be represented by the sextuple (0,IBAR,0,JBAR,0,KBAR).

There is a short Fortran 90 program provided, called **fds2ascii.f**, that can convert slice files into text files that can be read into a variety of graphics packages. The program combines multiple slice files corresponding to the same "slice" of the computational domain, time-averages the data, and writes the values into one file, consisting of a line of numbers for each node. Each line contains the physical coordinates of the node, and the time-averaged quantities corresponding to that node. In particular, the graphics package Tecplot will read this file and produces contour, streamline and/or vector plots. See Section 5.5.8 for more details about the program **fds2ascii**.

B.8 Boundary Files

The boundary files defined under the namelist group BNDF are named **casename_*n*.bf** (*n*=01,02...), and are written out unformatted. These files are written out from **dump.f** with the following lines:

```
WRITE(LUBF) QUANTITY
WRITE(LUBF) SHORT_NAME
WRITE(LUBF) UNITS
WRITE(LUBF) NPATCH
WRITE(LUBF) I1,I2,J1,J2,K1,K2
WRITE(LUBF) I1,I2,J1,J2,K1,K2
        .
        .
        .
WRITE(LUBF) TIME
WRITE(LUBF) (((QQ(I,J,K),I=11,I2),J=J1,J2),K=K1,K2)
WRITE(LUBF) (((QQ(I,J,K),I=11,I2),J=J1,J2),K=K1,K2)
        .
        .
        .
WRITE(LUBF) TIME
WRITE(LUBF) (((QQ(I,J,K),I=11,I2),J=J1,J2),K=K1,K2)
WRITE(LUBF) (((QQ(I,J,K),I=11,I2),J=J1,J2),K=K1,K2)
        .
        .
        .
```

QUANTITY, SHORT_NAME and UNITS are character strings of length 30. NPATCH is the number of planes (or "patches") that make up the solid boundaries plus the external walls. The sextuple (I1,I2,J1,J2,K1,K2) defines the cell nodes of each patch. The user does not prescribe these. Note that the data is planar, thus one pair of nodes will be the same.

Presently, Smokeview is the only program available to view the boundary files.

B.9 Particle Data

The tracer particles and sprinkler droplets are animated using a file called **casename.part**. It contains the particle positions. There is a one line header followed by particle locations output every DTSAM seconds. The structure of the data file is given in the file **dump.f** by a subroutine called PDMP3D.

```
WRITE(LU10) ARX,ARY,DTSAM,IPART,NPPS
WRITE(LU10) IBAR,JBAR,KBAR
WRITE(LU10) (X(I),I=0,IBAR),(Y(J),J=0,JBAR),(Z(K),K=0,KBAR)
WRITE(LU10) NB
DO N=1,NB
WRITE(LU10) IB1(N),IB2(N),JB1(N),JB2(N),KB1(N),KB2(N),1
END DO
WRITE(LU10) NV
DO N=1,NV
WRITE(LU10) IV1(N),IV2(N),JV1(N),JV2(N),KV1(N),KV2(N),2
END DO
WRITE(LU10) NSPR
DO N=1,NSPR
WRITE(LU10) XSP0(N),YSP0(N),ZSP0(N)
END DO
```

where ARX=ZBAR/XBAR and ARY=ZBAR/YBAR are the aspect ratios of the overall domain, IPART is the index of the scalar quantity associated with the particles, NPPS is the maximum number of particles per frame, IB1, IB2, *etc.* are the indices of blocked grid cells, IV1, IV2, *etc.* indicate vent cell nodes, and XSP0, YSP0, ZSP0 are the coordinates of the sprinklers. The arrays X(I), Y(J), Z(K) contain the coordinates of the grid. Every DTSAM seconds the coordinates of the tracer particles and sprinkler droplets are output

```
WRITE(LU10) TIME,NP,NN,(ISPR(N),N=1,NSPR)
WRITE(LU10) (XP(I),I=1,NP),
.           (YP(I),I=1,NP),
.           (ZP(I),I=1,NP),
.           (BP(I),I=1,NP)
IF (NASPR.GT.0) THEN
WRITE(LU10) NSP
WRITE(LU10) (XSP(I),I=1,NSP),(YSP(I),I=1,NSP),(ZSP(I),I=1,NSP)
ENDIF
```

where NP is the number of tracer particles, NN is a dummy integer kept for compatibility reasons, ISPR denotes whether the sprinkler has activated, NSPR is the number of sprinklers, XP, YP, ZP are the coordinates of the element, BP is a the quantity referenced by IPART, NASPR is the number of active sprinklers, NSP is the number of sprinkler droplets, and XSP, YSP, ZSP are the droplet coordinates.