NIST Special Publication 1017-1
Sixth Edition

Smokeview, A Tool for Visualizing Fire Dynamics Simulation Data Volume I: User's Guide

Glenn P. Forney

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

NIST Special Publication 1017-1
Sixth Edition

Smokeview, A Tool for Visualizing Fire Dynamics Simulation Data Volume I: User's Guide

Glenn P. Forney
Fire Research Division
Engineering Laboratory

August 21, 2020
Revision: SMV6.7.15-0-g4fa68dd

U.S. Department of Commerce
Wilbur L. Ross, Jr., Secretary

National Institute of Standards and Technology
Walter Copan, NIST Director and Undersecretary of Commerce for Standards and Technology

National Institute of Standards and Technology Special Publication 1017-1
Natl. Inst. Stand. Technol. Spec. Publ. 1017-1, 193 pages (June 2016)
CODEN: NSPUE2

Preface

This guide is part of a three volume set of companion documents describing Smokeview. Volume I, the Smokeview User's Guide [1], describes how to use Smokeview. Volume II, the Smokeview Technical Reference Guide [2], gives technical details of how the visualizations are performed. Volume III, the Smokeview Verification Guide [3] presents example cases verifying the various visualization capabilities of Smokeview. Details on the use and technical background of the Fire Dynamics Simulator is contained in the FDS User's [4] and Technical reference guide [5] respectively. This guide is Volume I the Smokeview User's guide.

Smokeview is a software tool designed to visualize numerical calculations generated by fire models such as the Fire Dynamics Simulator (FDS), a computational fluid dynamics (CFD) model of fire-driven fluid flow or CFAST, a zone fire model. Smokeview visualizes smoke and other attributes of the fire using traditional scientific methods such as displaying tracer particle flow, 2D or 3D shaded contours of gas flow data such as temperature and flow vectors showing flow direction and magnitude. Smokeview also visualizes fire attributes realistically so that one can *experience* the fire. This is done by displaying a series of partially transparent planes where the transparencies in each plane (at each grid node) are determined from soot densities computed by FDS. Smokeview also visualizes static data at particular times again using 2D or 3D contours of data such as temperature and flow vectors showing flow direction and magnitude.

Smokeview and associated documentation for Windows, Linux and Mac OS X may be downloaded from the web site `http://pages.nist.gov/fds` at no cost.

About the Author

Glenn Forney is a computer scientist at the Engineering Laboratory of NIST. He received a bachelor of science degree in mathematics from Salisbury State College and a master of science and a doctorate in mathematics from Clemson University. He joined NIST in 1986 (then the National Bureau of Standards) and has since worked on developing tools that provide a better understanding of fire phenomena, most notably Smokeview, a software tool for visualizing Fire Dynamics Simulator data.

Disclaimer

The US Department of Commerce makes no warranty, expressed or implied, to users of Smokeview, and accepts no responsibility for its use. Users of Smokeview 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 analysis performed using this tools.

Smokeview and the companion program 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. These software packages may or may not have predictive capability when applied to a specific set of factual circumstances. Lack of accurate predictions 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.

Acknowledgements

A number of people have made significant contributions to the development of Smokeview. In trying to acknowledge those that have contributed, we are inevitably going to miss a few people. Let us know and we will include those missed in the next version of this guide.

The original version of Smokeview was inspired by Frames, a visualization program written by James Sims for the Silicon Graphics workstation. This software was based on visualization software written by Stuart Cramer for an Evans and Sutherland computer. Frames used tracer particles to visualize smoke flow computed by a pre-cursor to FDS. Judy Devaney made the multi-screen eight foot Rave facility available allowing a stereo version of Smokeview to be built that can display scenes in 3D. Both Steve Satterfield and Tere Griffin on many occasions helped me demonstrate Smokeview cases on the Rave inspiring many people to the possibility of using Smokeview as a *virtual reality-like* fire fighter training facility.

Many conversations with Nelson Bryner, Dave Evans, Anthony Hamins and Doug Walton were most helpful in determining how Smokeview could be adapted for use in fire fighter training applications.

Smokeview would not be possible without the use of a number of software libraries developed by others. Mark Kilgard while at Silicon Graphics developed GLUT, the basic tool kit for interfacing OpenGL with the underlying operating system on multiple computer platforms. Paul Rademacher while a graduate student at the University of North Carolina developed GLUI, the software library for implementing the user friendly dialog boxes.

Significant contributions have been made by those that have used Smokeview to visualize complex cases; cases that are used to perform both applied and basic research. The resulting feedback has improved Smokeview as a result of their interaction with me, pushing the envelope and not accepting the status quo.

For applied research, Daniel Madrzykowski, Doug Walton and Robert Vettori of NIST have used Smokeview to analyze fire incidents. Steve Kerber has used Smokeview to visualize flows resulting from positive pressure ventilation (PPV) fans. David Stroup has used Smokeview to analyze cases for use in fire fighter training scenarios. Conversations with Doug Walton have been particularly helpful in identifying needed features and clarifying how best to make their implementation user friendly. David Evans, William (Ruddy) Mell and Ronald Rehm used Smokeview to visualize *wildland-urban interface* fires. For basic research, Greg Linteris has used Smokeview to visualize fire simulations involving the cone calorimeter. Anthony Hamins has used Smokeview to visualize the structure of CH_4/air flames undergoing the transition from normal to microgravity conditions and fire suppression in a compartment. Jiann Yang has used Smokeview to visualize smoke or particle number density and saturation ratio of condensable vapor.

This user's guide has improved through the many constructive comments of the reviewers Anthony Hamins, Doug Walton, Ronald Rehm, and David Sheppard. Chuck Bouldin helped port Smokeview to the Macintosh.

Many people have sent in multiple comments and feedback by email, in particular Adrian Brown, Scot Deal, Charlie Fleischmann, Jason Floyd, Simo Hostikka, Bryan Klein, Davy Leroy, Dave McGill, Brian McLaughlin, Derek Nolan, Steven Olenick, Stephen Priddy, Boris Stock, Jason Sutula, Javier Trelles, and Christopher Wood.

Feedback is encouraged and may be sent to glenn.forney@nist.gov .

Contents

List of Figures

List of Tables

Part I

Using Smokeview

Chapter 1

Introduction

1.1 Overview

Smokeview is a scientific software tool designed for visualizing numerical predictions generated by fire models such as the Fire Dynamics Simulator (FDS), a computational fluid dynamics (CFD) model of fire-driven fluid flow [6] and CFAST, a zone model of compartment fire phenomena [7]. This report documents version 6 of Smokeview. For details on setting up and running FDS cases read the FDS User's guide [4].

FDS and Smokeview are primarily used to model and visualize time-varying fire phenomena. FDS and Smokeview are not limited to fire simulation, however. For example, one may use these applications to model other phenomena such as contaminant flow in a building or evacuation flow. Smokeview performs visualizations by displaying time dependent tracer particle flow, animated contour slices of computed gas variables and surface data. Smokeview also presents contours and vector plots of static data anywhere within a simulation scene at a fixed time. Several examples using these techniques to investigate fire incidents are documented in Refs. [8, 9, 10, 11].

Smokeview is used before, during and after model runs. Smokeview is used in a post-processing step to visualize FDS data after a calculation has been completed. Smokeview may also be used during a calculation to monitor a simulation's progress and before a calculation to setup FDS input files more quickly. Figure 1.1 gives an overview of how data files used by FDS, Smokeview and Smokezip, a program used to compress FDS generated data files, are related.

Smokeview is written in both the programming languages C [12] and Fortran 2008 [13]. It consists of about 135 000 lines of code. The C portion visualizes the data, while the Fortran 2008 portion reads in data generated by FDS (also written in Fortran 2008). Smokeview uses the 3D graphics library OpenGL [14] for generating the visualizations and the Graphics Library Utility Toolkit (GLUT) [15] for interacting with the underlying OS. Smokeview uses the GLUT software library so that most of the development effort can be spent implementing the visualizations rather than creating an elaborate user interface. Smokeview uses a number of auxiliary libraries to implement image capture (GD [16, 17], PNG [18], JPEG [19]), image and general file compression (ZLIB [20]) and dialog creation (GLUI [21]). Each of these libraries is portable running under LINUX, OS X and Windows 7 allowing Smokeview to run on these platforms as well.

1.2 Features

Smokeview is a program designed to visualize numerical calculations generated by the Fire Dynamics Simulator. The version of FDS used to run the cases illustrated in this report is given by:

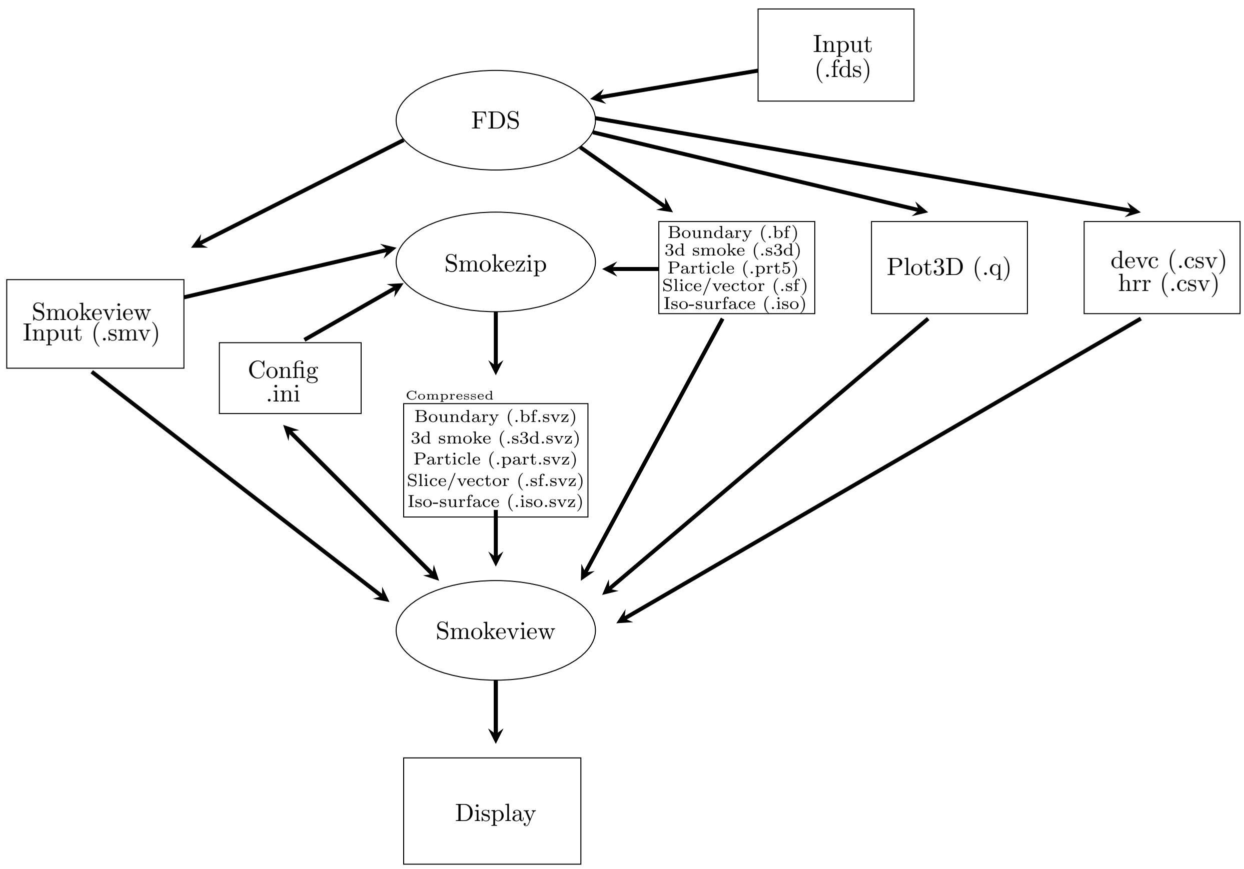


Figure 1.1: Diagram illustrating files used and created by the Fire Dynamics Simulator (FDS), Smokezip and Smokeview.

```
 Fire Dynamics Simulator

 Current Date     : August 21, 2020  10:35:19
 Revision         : FDS6.7.5-0-g71f0256-release
 Revision Date    : Thu Aug 20 16:19:01 2020 -0400
 Compiler         : Intel ifort 19.1.1.217
 Compilation Date : Aug 21, 2020 10:19:48

 MPI Enabled;    Number of MPI Processes:       1
 OpenMP Enabled; Number of OpenMP Threads:      8

 MPI version: 3.1
 MPI library version: Intel(R) MPI Library 2019 Update 7 for Linux* OS

 Consult FDS Users Guide Chapter, Running FDS, for further instructions.
```

The version of Smokeview described here and used to generate most figures in this report is given by:

```
Smokeview  SMV6.7.15-0-g4fa68dd-release - Aug 21 2020

Revision         : SMV6.7.15-0-g4fa68dd-release
Revision Date    : Mon Aug 10 10:50:47 2020 -0400
Compilation Date : Aug 21 2020 10:29:41
Compiler         : Intel C/C++ 19.1.1.217
Checksum(SHA1)   : 8afe6ca569e0f2a5ccded13ad19ecfbcc96697e8
Platform         : LINUX64
```

1.2.1 Visualizing Data

Smokeview visualizes data primarily generated by FDS. Smokeview visualizes data that is both dynamic and static. Dynamic data is visualized by animating particle flow (showing location and *values* of tracer particles), 2D contour slices (both within the domain and on solid surfaces) and 3D iso surfaces. 2D contour slices can also be drawn with colored vectors that use velocity data to show flow direction, speed and value. Static data is visualized similarly by drawing 2D contours, vector plots and 3D level surfaces.

Particles

Lagrangian or moving particles (Section 2.1) may be used to visualize the flow field. Often these particles represent smoke or water droplets. Particles may also be used to represent people when modeling evacuation flow.

Particle data may also be visualized as streak lines (a particle drawn where it has been for a short period of time in the past). Streak lines are a good method for displaying motion using static images.

Volumetric - Realistic Smoke

Smoke and fire (heat release rate per unit volume) are displayed realistically using a series of partially transparent planes (Section 2.2). The smoke transparencies are determined by using smoke densities computed by FDS. The fire and sprinkler spray transparencies are determined by using a heuristic based on heat release rate and water density data, again computed by FDS. Various settings for the 3D smoke option may be set using the 3D Smoke dialog box found in the *Dialogs>Data bounds* menu. The windows version of

Smokeview uses the graphical processing unit (GPU) on the video card to perform some of the calculations required to visualize smoke.

Slices - 2D contours

Animated 2D shaded color contour plots (Section 3.2) are used to visualize gas phase information, such as temperature or density. The contour plots are drawn in horizontal or vertical planes along any coordinate direction. Contours can also be drawn in shades of gray. Shaded contours may also be used to visualize information computed on solid objects (Section 3.3). These contours are known as boundary files.

Animated 2D shaded color contour plots are also used to visualize solid phase quantities such as radiative flux or heat release rate per unit area.

Vector slice files may be visualized if U, V and W velocity slice files are recorded. Though similar to solidly shaded contour animations (the vector colors are the same as the corresponding contour colors), vector animations are better than solid contour animations for highlighting flow features since vectors accentuate the direction that flow is occurring.

A 3D region of data may be visualized using slice files. Slices may be moved from one plane to the next just as with Plot3D files (using up/down cursor keys or page up/page down keys). 3D slices may also be rotated and/or translated by double clicking and moving the mouse. If the `CTRL` or `d` key is also pressed (press and release the `d` key do no hold it down), the slice moves up and down. If the `ALT` or `f` key is pressed, (press and release the `f` key do no hold it down), the slice moves side to side. Otherwise, the slice rotates. Data for 3D slice files are generated by specifying a 3D rather than a 2D region with the `&SLCF` keyword.

Data computed at cell centers rather than interpolated at cell nodes may be visualized. This is useful for investigating numerical algorithms as the data visualized has not been interpolated before being seen.

Surfaces - 3D contours

Isosurface or 3D level surface animations (Section 3.4) may be used to represent flame boundaries, layer interfaces and various other gas phase variables. Multiple isocontours may be stored in one file, allowing one to view several isosurface levels simultaneously.

1.2.2 Exploring Data

Data Mining

The user can analyze and examine the simulated data by altering its appearance to more easily identify features and behaviors found in the simulation data. One may flip or reverse the order of colors in the colorbar and also click in the colorbar and slide the mouse to highlight data values in the scene. These options may be found under the *Options/Shades* menu.

The user may click in the timebar and slide the mouse to change the simulation time displayed. One use for the timebar and colorbar selection modes might be to determine when smoke of a particular temperature enters a room.

Data Filtering

The File/Bounds Settings... dialog box allows one to set bounds, to chop or hide data and in the case of slice file data to time average. (Chapter 5) The data chopping feature is useful for highlighting data. A ceiling jet, for example, may be visualized by hiding ambient temperature data, data below a prescribed temperature. Using time averaging allows one to smooth noisy data over a user selectable time interval.

Data coloring

Multiple colorbars are available for displaying simulation data. New colorbars may be created using the colorbar editor (Section 9.2). Colorbars may then be adapted to best highlight the simulation data visualized. Regions in the simulation with certain data values may be highlighted by clicking on the colorbar.

Data Compression

An option has been added to the *LOAD/UNLOAD* menu to compress 3D smoke and boundary files (Section 15.1). The option shells out to the program Smokezip which runs in the background enabling one to continue to use Smokeview while files are compressing.

Data Comparison

A stand alone utility program named Smokediff may be used to compare two FDS cases (Section 15.2). Smokediff generates the difference between corresponding slice and boundary files for two cases with the same geometry. Smokediff creates a `.smv` of the differenced data which may then be viewed with Smokeview.

1.2.3 Exploring the Scene

Motion/View/Render

The Motion/View/Render dialog box may be used to allow more precise control of scene movement and orientation. Cursor keys have been mapped to scene translation/rotation to allow easy navigation within the scene. Viewpoints may be saved for later access.

The first person or eye view mode for moving allows one to move through a scene more realistically (Section 1.4.2). Using the cursor keys and the mouse, one can move through a scene *virtually*.

Stereo views

A method for displaying stereo/3D images has been implemented that does not require any specialized equipment such as shuttered glasses or quad buffered enabled video cards (Section 5.4.4) . Stereo pair images are displayed side by side after invoking the option with the Stereo dialog box or pressing the “S” key (upper case). A 3D view appears by relaxing the eyes, allowing the two images to merge into one. Pressing the “S” key again results in stereo views generated by displaying red and blue versions of the scene. Glasses with a red left lens and a blue right lens are required to view the image.

Scene Clipping

It is often difficult to visualize data in complicated geometries due to the number of obstructed surfaces. Interior portions of the scene may be seen more easily by clipping part of the scene away. (Section 5.7)

Clipping discussed above occurs in 3D within the scene. A screenshot converted to a PNG or JPEG file may also be clipped or cropped using the Render portion of the Motion/View/Render dialog box.

1.2.4 Automating the Visualization

Virtual Tour

A series of checkpoints or key frames specifying position and view direction may be specified. (Chapter 7) A smooth path is computed using Kochanek-Bartels splines [22] to go through these key frames so that one may control the position and view direction of an observer as they move through the simulation. One can then see the simulation as the observer would. This option is available under the *Tour* menu item. Existing tours may be edited and new tours may be created using the Tour dialog box found in the *Dialogs>View* menu. Tour settings are stored in the local configuration file (casename.ini).

Scripting

Smokeview may be run in an unattended mode using instructions found in a script file. (Chapter 8) These instructions direct Smokeview to load data files, load configuration files, set view points and time values in order to document a case by rendering the Smokeview scene into one or more image files. The script file may be created by Smokeview as a user performs various actions or may be created by editing a text file.

1.2.5 Customizing the Scene

Objects

A method for drawing realistic appearing objects such as a heat detector, smoke detector, sprinkler sensor, etc. has been implemented. (Chapter 6) . Objects are specified in a data file rather than in Smokeview as C code. This allows one to customize the look and feel of the objects (to match the types of detectors/sprinklers that are being used) without requiring code changes in Smokeview.

Texture Mapping

Image files may be drawn over top of a blockage, vent or enclosure boundary (Chapter 11). This is called texture mapping. This allows Smokeview scenes to appear more realistic. These image files may be obtained from the internet, a digital camera, a scanner or from any other source that generates these file formats. Image files used for texture mapping should be seamless. A seamless texture as the name suggests is periodic in both horizontal and vertical directions. This is an especially important requirement when textures are tiled or repeated across a blockage surface.

Annotating Cases

Text may be added to a scene in order to help document Smokeview output. (Chapter 14) It allows one to place colored labels at specified locations at specified times. A second keyword, `TICK` keyword places equally spaced tick marks between specified bounds. These marks along with `LABEL` text may be used to specify length scales in the scene.

The *User Tick Settings* tab of the Display dialog box provides an easier way to place ticks with length annotations along coordinate axes.

1.3 Getting Started

1.3.1 Obtaining Smokeview

Smokeview is available at `http://pages.nist.gov/fds`. This site contains links to installation packages for Windows, Linux and Mac OS X operating systems. It also contains documentation for Smokeview and FDS, sample FDS calculations, software updates and links for requesting feedback about the software.

After obtaining the setup program, install Smokeview (and FDS) on the PC by double-clicking the downloaded setup program. The setup program then steps through the program installation. It copies the

FDS and Smokeview executables, sample cases and documentation to the selected directory. The setup 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. Smokeview uses the OpenGL graphics library which is a part of all Windows distributions.

Most computers purchased today are perfectly adequate for running Smokeview. For Smokeview it is more important to obtain a fast graphics card than a fast CPU. If the computer will run both FDS and Smokeview, then a fast CPU is important as well. For example, the townhouse case used for many examples in this report consists of about 23000 grid cells. This case requires about 10 CPU minutes on a 2.0 GHZ Intel Core i7-2630QM Windows 7 system. Cases with more grid cells and longer simulation times (the townhouse case simulated 60 s of smoke flow) would clearly benefit from a faster CPU and more memory which are now relatively inexpensive.

1.3.2 Running Smokeview

A typical procedure for using FDS and Smokeview is to:

1. Create a file named `casename.fds` describing the fire scenario.
2. Type `fds casename.fds` in a command shell to run the case.
3. Double click on the file named `casename.smv` (if on the PC) or type `smokeview casename` in a command shell (on other platforms) to start Smokeview.
4. Right clicking within the scene and select a file to load within the *Load/Unload* menu.

This report documents step 3 and 4. Steps 1 and 2 are documented in the FDS User's Guide [4].

Menus in Smokeview are activated by clicking the right mouse button anywhere within the Smokeview window. Data files may be visualized by selecting the desired *Load/Unload* menu option. Other menu options are discussed in Appendix B. Many menu commands have equivalent keyboard shortcuts. These shortcuts are listed in Smokeview's *Help* menu and are described in Appendix C. Visualization features not controllable through the menus may be customized by using the Smokeview preference file, `smokeview.ini` , discussed in Appendix D.3.

Smokeview is started on a Windows PC 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. Menus are accessed by clicking with the right mouse button. The *Load/Unload* menu may be used to read in the data files to be visualized. The *Show/Hide* menu 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. Menu items are described in Appendix B.

To use Smokeview from a command line, open a command shell. Then change to the directory containing the FDS case to be viewed and type:

```
smokeview casename
```

where again 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.

Normally Smokeview is run during an FDS run, after the run has completed and 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 located as intended. One may select the color of these elements using color parameters in the `smokeview.ini` to help distinguish one element from another. `smokeview.ini` file entries are described in section D.3.

Although specific video card brands cannot be recommended, they should be high-end due to Smokeview's intensive graphics requirements. These requirements will only increase in the future as more features are added. A video card designed to perform well for *fancy* computer games should do well for Smokeview. Some apparent bugs in Smokeview have been found to be the result of problems found in video cards on older computers.

1.4 Manipulating the Scene

A Smokeview scene may be rotated or moved by using either the mouse or the Motion/View/Render dialog box. The scene may be rotated about a point within the scene, usually the scene center, or rotated about the point where the user is located. In either case, to rotate, click the scene with the left mouse button and drag either horizontally or vertically. Motion about the scene center which is the default is called world or global view while motion about the user location is called eye or first person view. These viewing modes may be swapped by pressing the "e" key or by selecting the appropriate radio button in the Motion/View/Render dialog box.

Similarly, the scene may be translated by clicking the scene with the left mouse button while either the `ALT` or `CTRL` keys are pressed. The `ALT` key results in vertical motion of the scene while the `CTRL` key results in motion in and out.

1.4.1 World View

The scene may be rotated or translated using the mouse or by using controls in the Motion/View/Render dialog box. This dialog box, illustrated in Fig. 1.2, is opened using the *Dialogs>Motion* menu item.

Clicking the left mouse button and dragging horizontally, vertically or a combination of both results in scene rotation or translation depending upon whether the CTRL or ALT modifier keys are pressed or not.[1]

no modifier keys Horizontal mouse movement results in scene rotation about the Z axis. Vertical mouse movement results in scene rotation about the X axis. If the 3-axis rotation option is selected then mouse movement around the periphery of the scene results in clockwise or counter clockwise movement about the Y axis.

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 in and out of the along the Y axis.

ALT key depressed Vertical mouse movement results in scene translation up and down along the Z axis. Horizontal mouse movement has no effect.

1.4.2 First Person View

First person view is entered by either pressing the appropriate radio buttons in the Motion/View/Render dialog box (button labeled eye centered) or by pressing the "e" key until first person view is obtained. When

[1] The Sierra version of the Macintosh OS no longer supports the `ALT` and `CTRL` keys when used with the mouse to move the scene. The `d` key may be used in place of `CTRL` key and the `f` key may be used in place of the `ALT` key. Press and release the `d` and `f` keys do not hold them down.

in *eye center* mode, several key mappings have been added, inspired by popular computer games, to allow for easier movement within the scene. For example, the up and down cursor keys allow one to move forward or backwards. The left and right cursor keys allow one to rotate left or right. Other keyboard mappings are described in Table 1.1.

Table 1.1: Keyboard mappings for *eye centered* or first person scene movement.

Key	Description
up/down cursor w/s	move forward/backward
`ALT` + left/right cursor `a/d`	slide left/right
`ALT` + up/down cursor	move up/down
left/right cursor	rotate left/right
`Page Up/Down`	look up/down
`Home`	look level
Pressing the `SHIFT` key while moving, sliding or rotating results in a 4x speedup of these actions.	

1.4.3 Motion Dialog Box

The Motion dialog box illustrated in Fig 1.2 may also be used to manipulate the scene. Buttons in the Motion region of the dialog box allow one to translate or rotate the scene. The `Horizontal` button is used to translate the scene horizontally within a horizontal plane (left/right or in/out within the scene). The `Vertical` button allows one to translate the scene vertically. Scene translation and rotations may also be specified by using controls to set x, y, z coordinates and azimuth, elevation rotation angles within the *Specify Orientation* panel. Within this same panel one may also specify whether the gravity vector (usually pointed down) is visualized and/or used to draw the scene.

The `Rotate about` selection list allows one to change the center of rotation. Rotation center choices are: the scene center, denoted as *world center* in this list, the center of each mesh and a center specified by the user. Changing the rotation center is useful for cases where the portion of the scene being viewed is far away from the currently used rotation center. To allow for easier changes, the rotation center may be made visible (drawn as a small black sphere) by selecting the *Show* checkbox within the *rotation center panel*. The rotation center may be changed by using the *x, y, z* spinners below this checkbox.

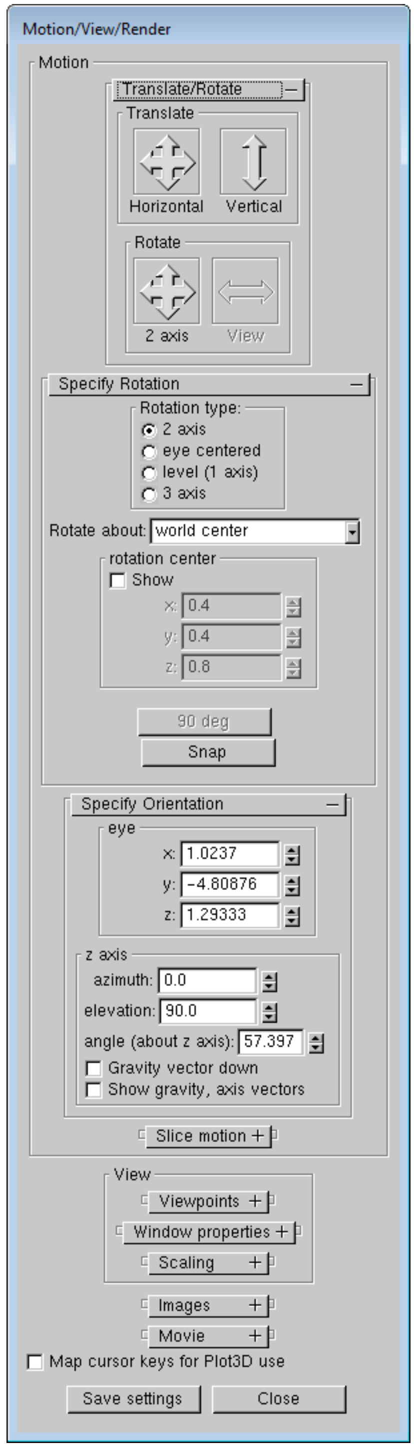


Figure 1.2: Dialog box for controlling scene motion. To rotate the scene, select the rotation type then select and move the mouse in the rotation control. One may also select where to rotate about (scene center, any mesh center or user specified center). To translate the scene, select and move the mouse in the horizontal or vertical control.

Chapter 2

Visualizing Smoke

2.1 Tracers and Streaklines

Particle files contain the locations of tracer particles used to visualize the flow field. Figure 2.1 shows several snapshots of a developing kitchen fire visualized by using particles where particles are colored black. If present, sprinkler water droplets would be colored blue. Particles are stored in files ending with the extension `.prt5` and are displayed by selecting the particle file entry from the *Load/Unload* menu.

Streaklines are a technique for showing motion in a still image. Figure 2.2 shows a snapshot of the same kitchen fire using streak lines instead of particles. The streaks begin at 9 s and end at 10 s.

Particle file data may be converted to an isosurface using Smokezip. The isosurface location is defined in terms of particle density and the isosurface color is defined in terms of averaged particle values. See Chapter 15.1 for more details on using Smokezip for generating isosurface files from particle files and Section 3.4 for some examples.

2.2 Realistic

FDS generates several data files visualized by Smokeview. Each file type may be loaded or unloaded using the *Load/Unload* menu described in Appendix B.1. Visualizations produced by these data files are described in this and the following sections. The format used to store each of the data files is given in the FDS User's Guide [4].

Visualizing smoke realistically is a daunting challenge for at least three reasons. First, the storage requirements for describing smoke can easily exceed the disk capacities of present 32 bit operating systems such as Linux, i.e., file sizes can easily exceed 2 gigabytes. Second, the computation required both by the CPU and the video card to display each frame can easily exceed 0.1 s, the time corresponding to a 10 frame/s display rate. Third, the physics required to describe smoke and its interactions with itself and surrounding light sources is complex and computationally intensive. Therefore, approximations and simplifications are required to display smoke rapidly.

Smoke visualization techniques such as tracer particles or shaded 2D contours are useful for quantitative analysis but not suitable for virtual reality applications, where displays need to be realistic and fast as well as accurate. The approach taken by Smokeview is to display a series of parallel planes. Each plane is colored black (for smoke) with transparency values pre-computed by FDS using time dependent soot densities also computed by FDS corresponding to the grid spacings of the simulation. The transparencies are adjusted in real time by Smokeview to account for differing path lengths through the smoke as the view direction

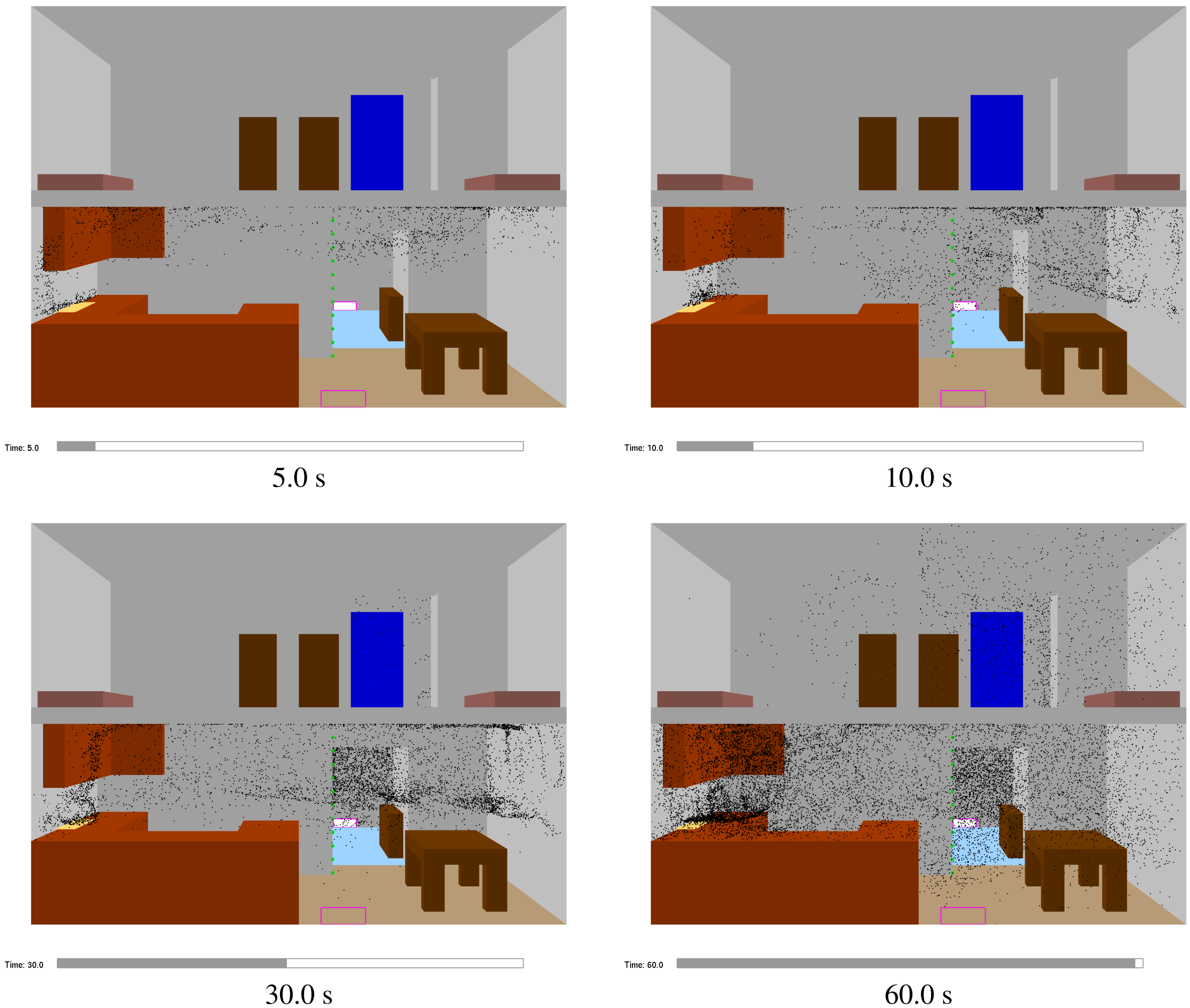


Figure 2.1: Townhouse kitchen fire visualized using tracer particles.

Figure 2.2: Townhouse kitchen fire visualized using streak lines. The *pin heads* shows flow conditions at 10 s, the corresponding *tails* shows conditions 1.0 s earlier.

changes. The graphics hardware then combines the planes together to form one image.

Fire by default is colored a dark shade of orange wherever the computed heat release rate per unit volume exceeds a user-defined cutoff value. The visual characteristics of fire are not automatically accounted for. The user though may use the 3D Smoke dialog box to change both the color and transparency of the fire for fires that have non-standard colors and opacities.

The windows version of Smokeview has the option of using the GPU or graphics programming unit to perform some of the calculations required to visualize realistic smoke. These calculations consist of adjusting the smoke opaqueness as pre-computed in FDS to account for off-axis viewing directions. The GPU performs the computations in parallel while the former method using the CPU performs them sequentially. For many (but not all) cases, the use of the GPU results in a smoke drawing speed up of 50 % or more. This option is turned on or off by pressing the `G` key.

Figure 2.3 illustrates a visualization of realistic smoke.

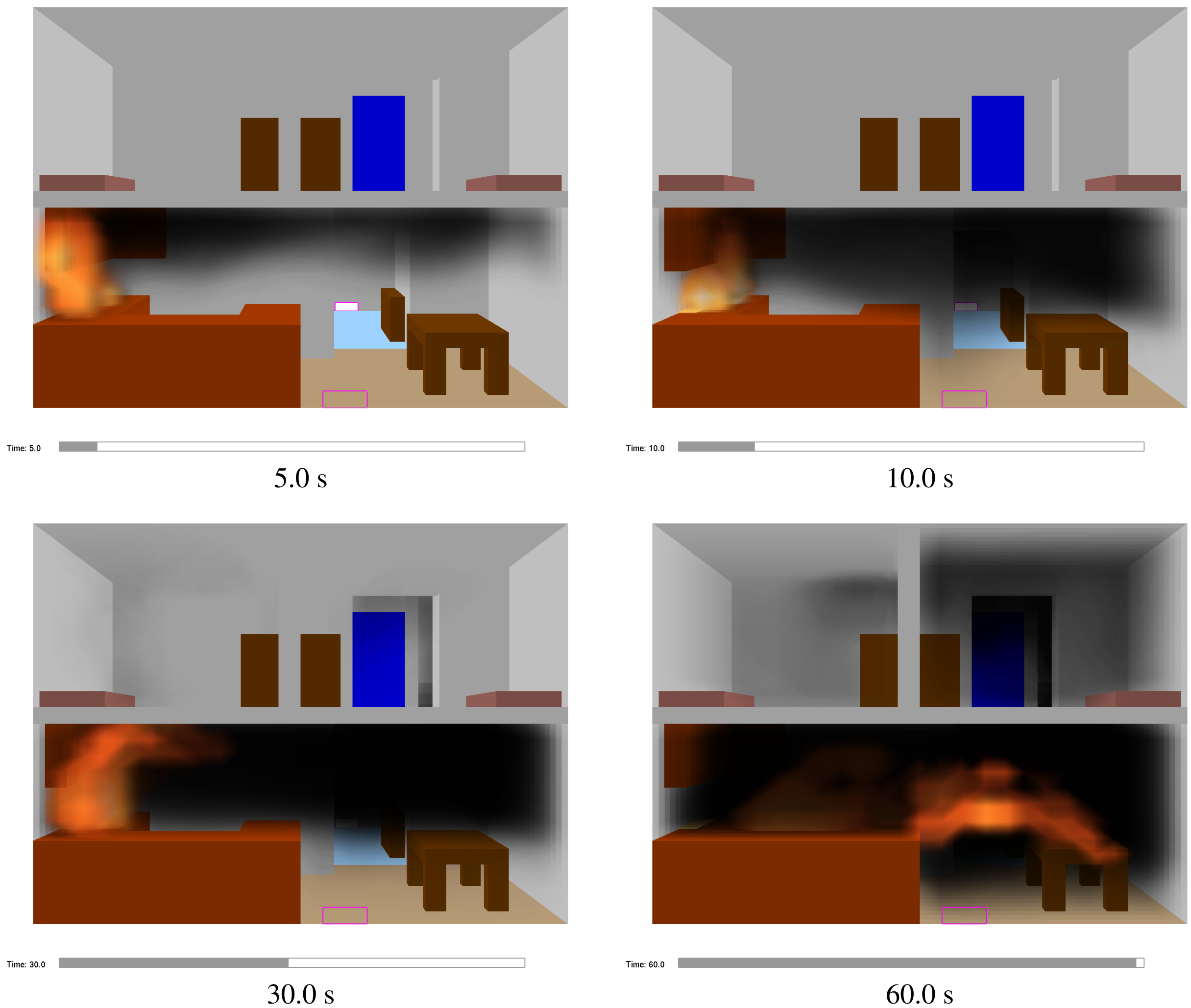


Figure 2.3: Smoke3d file snapshots at various times in a simulation of a townhouse kitchen fire.

Chapter 3

Visualizing Data Quantitatively

3.1 Coloring data

Smokeview uses a 1D texture map for coloring data occurring in slice, boundary and Plot3D files. 1D texture maps or colorbars may be selected using the Data coloring dialog box illustrated in Fig. 3.1. This dialog box is used to control how data is colored. One may select the mapping used to associate data with color (a colorbar). One may also select the colors used to color extreme data, data with values greater the maximum or smaller than the minimum (smallest and greatest colorbar data labels). When the colorbar is selected with the mouse a portion of it changes color to black. Data in the scene with the same values are also colored black. The width of the selection region (default 5 pixels) may be selected with this dialog box. Other data coloring properties such as transparency, order of the colorbar may also be selected.

A colorbar defined with a split may also be specified with this dialog box. One range of colors are specified between the minimum data value and the split value and another range of colors are specified between the split value and the maximum value. This is useful when highlighting data that has a special property, for example a tenability temperature criteria or where flow velocity reverses. Figure 3.2 illustrates a scene using a colorbar with a split at 0.0 m/s. Velocities greater than 0.0 m/s are colored with shades of red. Velocities less than 0.0 m/s are colored with shades of blue.

Note, due to the way that transparent objects are drawn (from back to front), 3D Smoke/Fire and transparent slices may not display properly when shown at the same time.

3.2 2D Shaded Contours and Vector Slices - Slice Files

3.2.1 Axis aligned slices

Slice files contain results recorded within a rectangular array of grid points at each recorded time step. Continuously shaded contours are drawn for simulation quantities such as temperature, gas velocity and heat release rate. Figure 3.3 shows several snapshots of a vertical animated slice where the slice is colored according to gas temperature. Slice files have file names with extension `.sf` and are displayed by selecting the desired entry from the *Load/Unload* menu.

All slice files oriented along the same plane (x, y and/or z directions) may be loaded with one mouse click by choosing the desired orientation from the *Slice* portion of the *Load/Unload* menu. These menu entries do not exist in the Multi-Slice menu. However, when selected from the *Slice* menu, Smokeview will load all x/y/z oriented multi-slices.

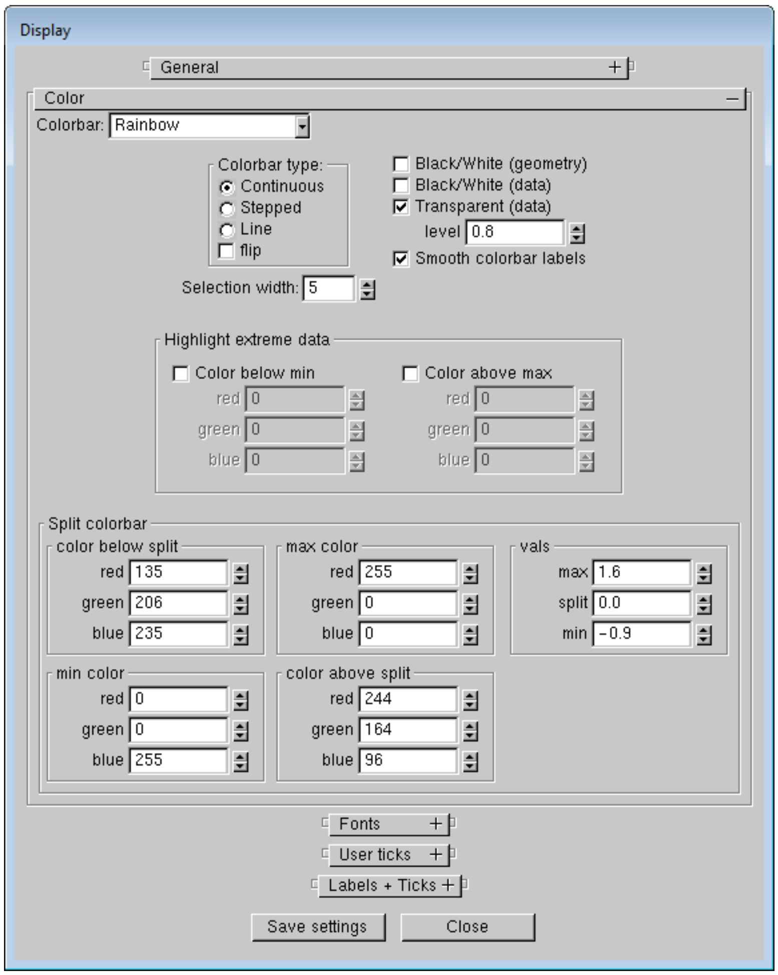


Figure 3.1: Dialog box for selecting colorbars. The selected colorbar may be modified by shading it continuously, stepped or as a series of discrete lines. The colorbar may also be converted to shades of gray. Colors for extreme data (data outside of specified bounds) may be specified. A split colorbar may be specified using one range of colors between the minimum value and the split and another range of colors between the split and the maximum value.

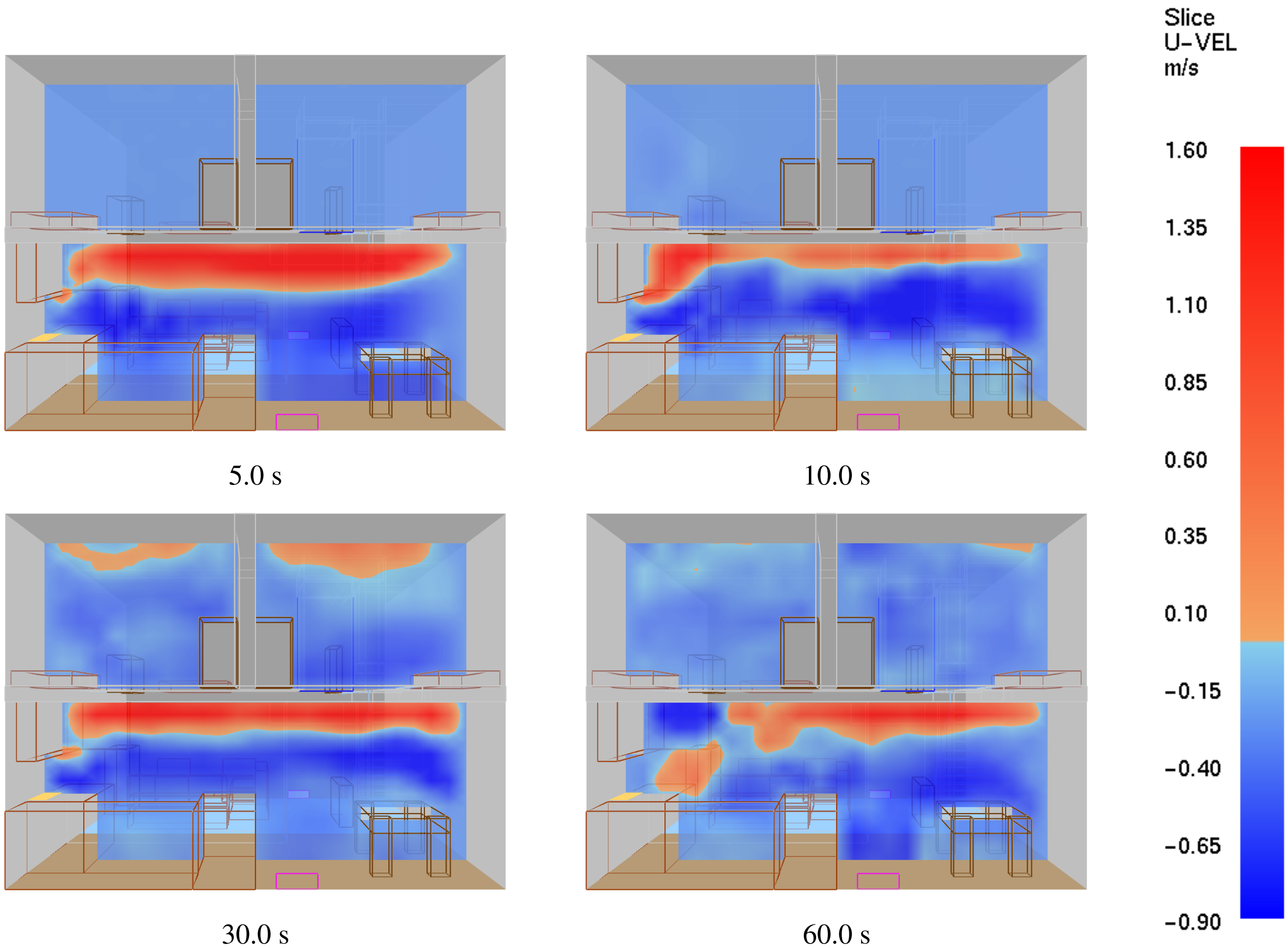


Figure 3.2: Slice file snapshots of shaded U velocity contours at various times in a simulation using a colorbar with a split at 0.0 m/s. Velocities greater than 0.0 m/s are colored with shades of red. Velocities less than 0.0 m/s are colored with shades of blue.

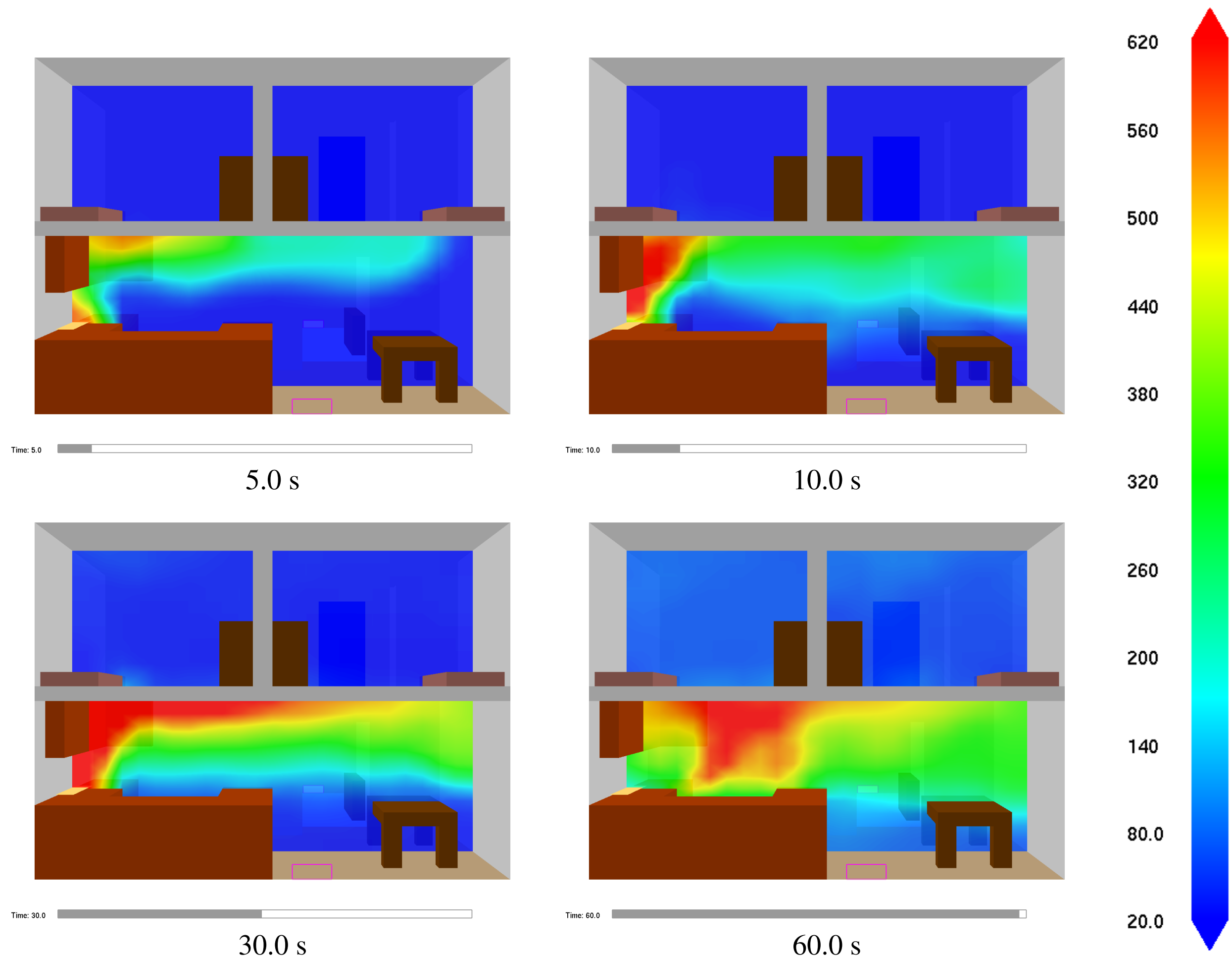


Figure 3.3: Slice file snapshots of shaded temperature contours at various times in a simulation. These contours were generated by adding "`&SLCF PBY=1.5, QUANTITY='TEMPERATURE' /`" to the FDS input file.

To specify in FDS a vertical slice 1.5 m from the $y = 0$ boundary colored by temperature, use the line:

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

A more complete list of output quantities may be found in Ref. [4].

Vector slices 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. The length of the vectors can be adjusted by pressing the `'a'` key. For cases with a fine grid, the number of vectors may be overwhelming. Vectors may be skipped by pressing the `'s'` key. Figure 3.4 shows a sequence of vector slices corresponding to the shaded temperature contours found in Fig. 3.3.

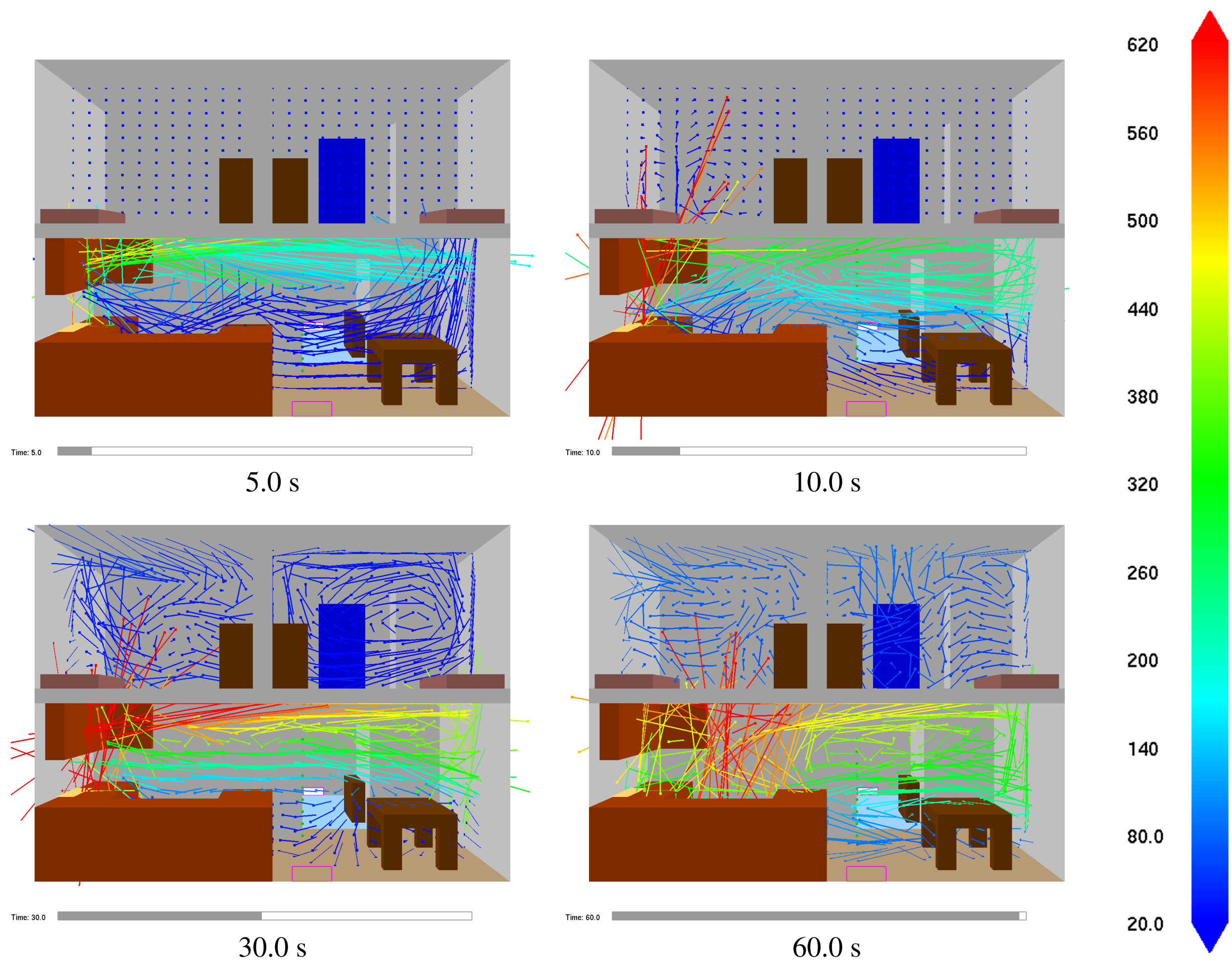


Figure 3.4: Vector slice file snapshots of shaded vector plots. These vector plots were generated by using "`&SLCF PBY=1.5,QUANTITY='TEMPERATURE',VECTOR=.TRUE. /`".

Similar to slice files, all vector slice files oriented along the same plane (x, y and/or z directions) may be loaded with one mouse click by choosing the desired orientation from the *Vector Slice* portion of the *Load/Unload* menu. These menu entries do not exist in the Vector Multi-Slice menu. However, when selected from the *Vector Slice* menu, Smokeview will load all x/y/z oriented multi-slices.

To generate the extra velocity files needed to view vector animations, add `VECTOR=.TRUE.` to the above `&SLCF` line to obtain:

```
&SLCF PBY=1.50,QUANTITY='TEMPERATURE',VECTOR=.TRUE. /
```

3.2.2 3D slices

The user may visualize a 3D region of data using slice files. To specify a cube of data from 1.0 to 2.0 in each of the X, Y and Z directions in FDS, use the line:

```
&SLCF XB=1.0,2.0,1.0,2.0,1.0,2.0 QUANTITY='TEMPERATURE' /
```

A slice from the resulting slice file may be moved from one plane to the next just as with Plot3D files (using left/right, up/down cursor keys or page up/page down keys). 3D slices and 3D vector slices may also be oriented arbitrarily. To make these slices visible press the `w` key. Examples of these slices are illustrated in Figs. 3.5 and 3.6. These slice may also be oriented in arbitrary positions and directions by double clicking within the scene. While holding down the mouse after double clicking, move the mouse from side to side or up and down to rotate the general slice. Double clicking and moving the mouse vertically while holding down the `ALT` key causes the center of rotation for the general slice to move up and down. Double clicking and moving the mouse horizontally and vertically while holding down the `SHIFT` key causes the center of rotation for the general slice to move along the `X` and `Y` axis respectively.

The position and orientation of 3D slices may be manipulated using the Slice motion portion of the Motion/View/Render dialog box as illustrated in Fig. 3.7.

3.2.3 Wind Roses

A wind rose displays a 2D summary of how flow velocities are distributed at a point over some period of time. Data for a wind rose is generated by specifying devices for U, V and W components of velocity using `&DEVC` keywords such as

```
&DEVC XYZ=1.2,0.8,1.0 QUANTITY='U-VELOCITY' /
&DEVC XYZ=1.2,0.8,1.0 QUANTITY='V-VELOCITY' /
&DEVC XYZ=1.2,0.8,1.0 QUANTITY='W-VELOCITY' /
```

in an FDS input file. Smokeview creates a two dimensional histogram recording the distribution of wind speeds and direction. The user can then set various viewing options using the wind rose dialog box illustrated in Figure 3.8. Figure 3.9 illustrates wind roses at several locations drawn in horizontal (xy) and vertical (xz) planes.

3.2.4 Fractional effective dose (FED) slices

The fractional effective dose (FED), developed by Purser [23], is an estimate of human incapacitation due to a limited set of combustion gases. FED index data is computed by Smokeview using CO, CO_2 and O_2 gas

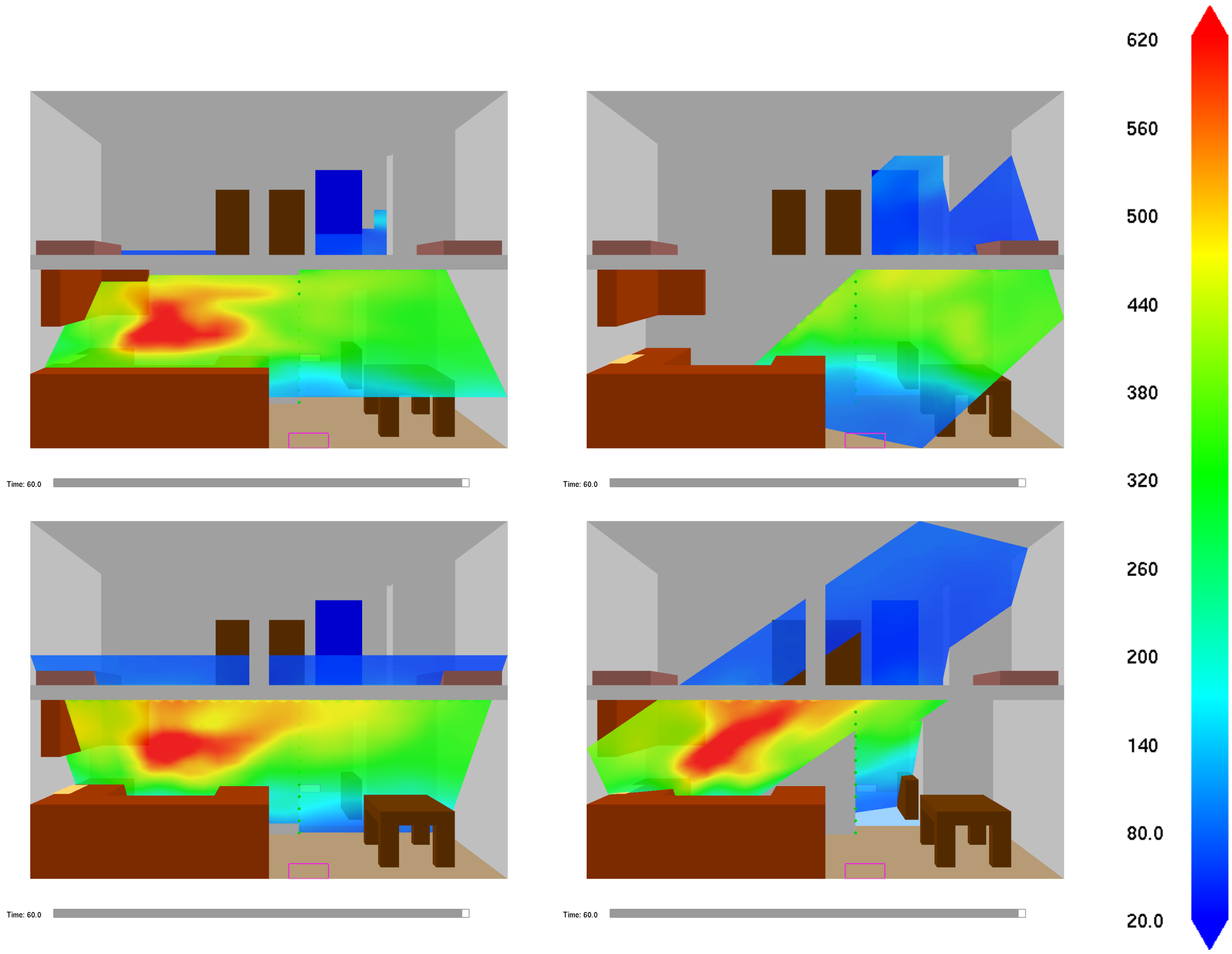


Figure 3.5: Slices from a 3D temperature slice file at 60 s displayed using four orientations. 3D slices may be re-oriented by double clicking and dragging the mouse or by changing settings in the Motion/View/Render dialog box. These images were generated using "`&SLCF XB=0.0,6.4,0.0,8.0,0.0,4.8, QUANTITY='TEMPERATURE' /`" in an FDS input file.

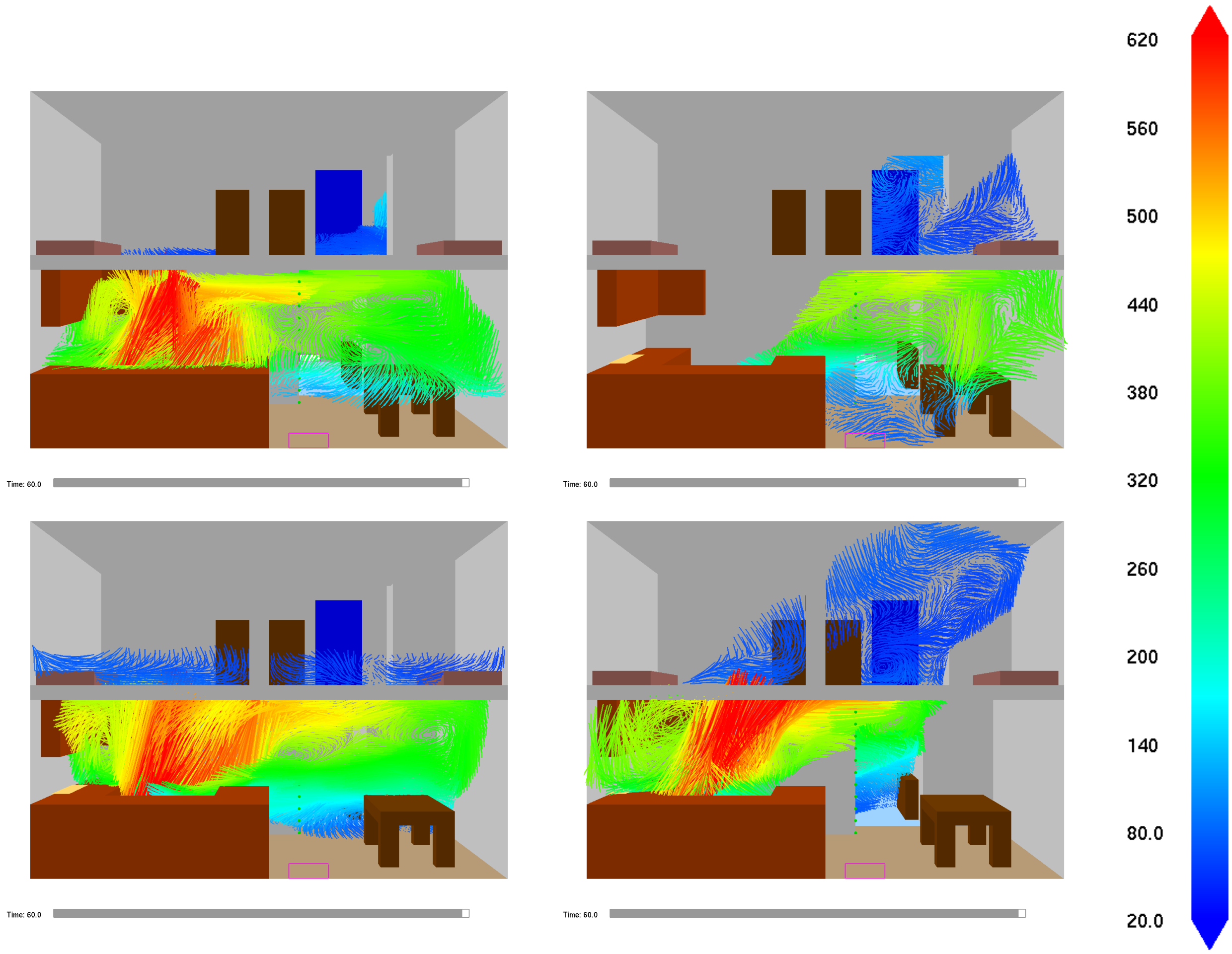


Figure 3.6: Vector 3D temperature slices at 60 s displayed using four orientations. 3D vector slices may be re-oriented by double clicking and dragging the mouse or by changing settings in the Motion/View/Render dialog box. These images were generated using "`&SLCF XB=0.0,6.4,0.0,8.0,0.0,4.8, QUANTITY='TEMPERATURE', VECTOR=.TRUE./`" in an FDS input file.

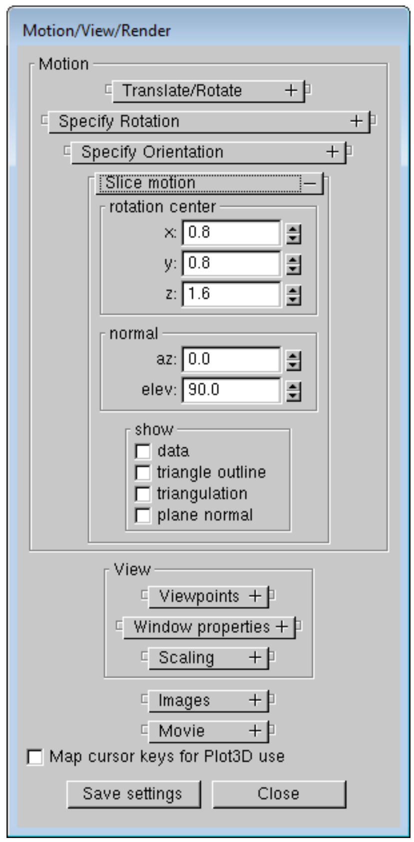


Figure 3.7: Dialog box for controlling the orientation of a 3D slice file.

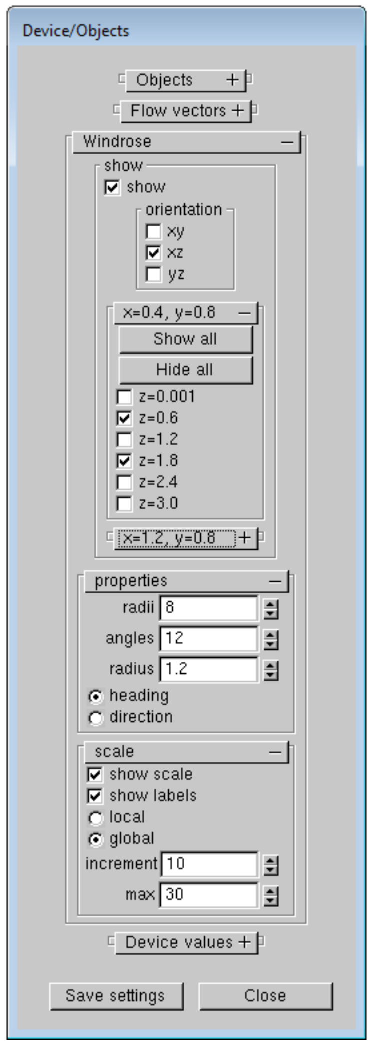
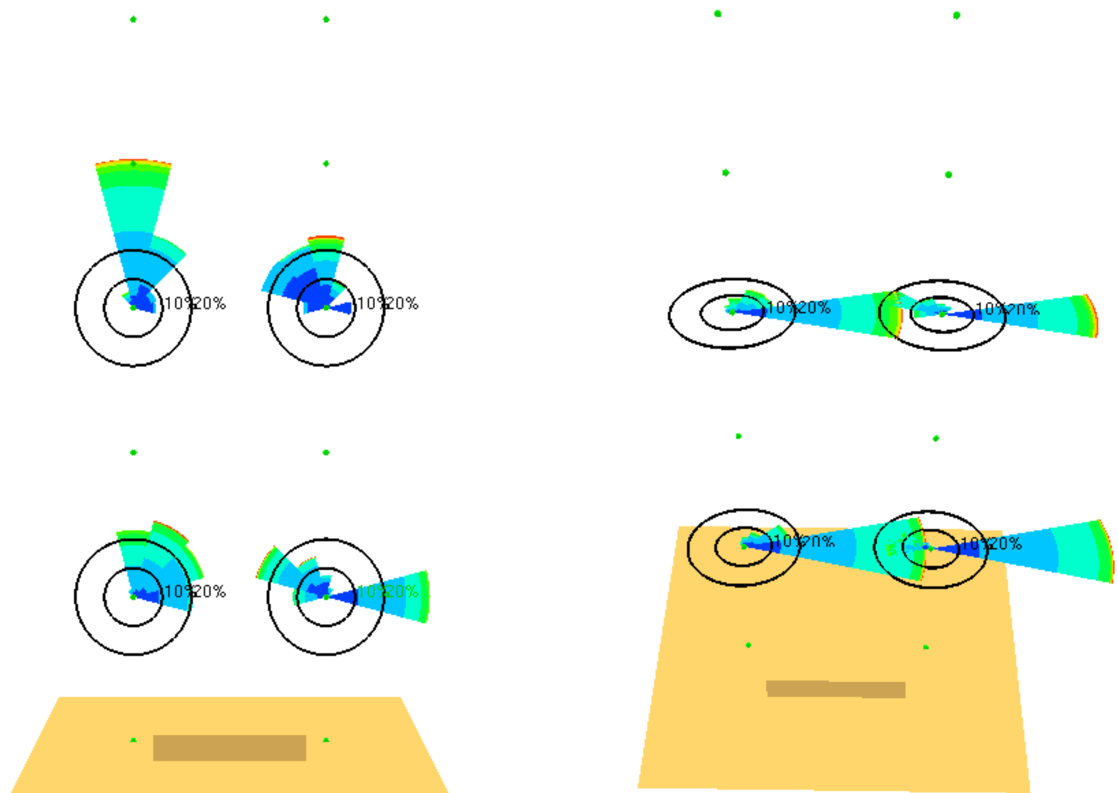


Figure 3.8: Dialog Box for setting wind rose options



XZ XY

Figure 3.9: Wind roses visualizing velocity flow distributions in XZ and XY planes at several locations

concentration data computed by FDS. This data is made available to Smokeview in the form of slice files. Smokeview computes FED data using

$$\mathrm{FED}_{\mathrm{tot}} = \mathrm{FED}_{\mathrm{CO}} \times \mathrm{HV}_{\mathrm{CO}_2} + \mathrm{FED}_{\mathrm{O}_2} \tag{3.1}$$

where $\mathrm{FED}_{\mathrm{tot}}$ is the total FED, $\mathrm{FED}_{\mathrm{CO}}$ is the FED due to CO, $\mathrm{HV}_{\mathrm{CO}_2}$ is a hyper-ventilating factor applied to CO and $\mathrm{FED}_{\mathrm{O}_2}$ is the FED due to O_2. The species data slices used to compute an FED slice needs to be specified at the same location. To generate an FED slice at $y = 1.6$, specify the following species slices in the input file

```
&SLCF PBY=1.6,QUANTITY='VOLUME FRACTION' SPEC_ID='CARBON DIOXIDE' /
&SLCF PBY=1.6,QUANTITY='VOLUME FRACTION' SPEC_ID='CARBON MONOXIDE' /
&SLCF PBY=1.6,QUANTITY='VOLUME FRACTION' SPEC_ID='OXYGEN' /
```

FED computations are stored by Smokeview in slice files for subsequent use. Since this computation is performed in Smokeview using data only found in the slices files, time step intervals should be chosen to ensure accuracy. Figure 3.10 illustrates an FED slice file at several times. The FED colorbar is split at values of 0.3, 1.0 and 3.0. When FED slices are displayed using the FED colorbar (colorbar illustrated in Figure 3.10), Smokeview computes color levels assuming a minimum FED level of 0.0 and a maximum level of 3.0. To display FED data using other data bounds, a different colorbar needs to be chosen.

3.2.5 Duplicate Slices

FDS outputs duplicate slices whenever a `&SLCF` entry is specified where two meshes coincide. One set of slices is output for each mesh. Using the Slice/Duplicates panel of the File/Bounds dialog box, illustrated in Figure 3.11, one may specify whether to keep all duplicate slices, keep the finely gridded slices or keep the coarsely gridded slices. One may similarly specify preferences for vector slice files. By default smokeview keeps only finely gridded slices and keeps all vector slices (so one may diagnose possible flow problems).

FDS also outputs duplicate slices when two or more identical `&SLCF` entries are specified in the input (`.fds`) file. Smokeview ignores these duplicate slices when processing the `.smv` file.

3.3 2D Shaded Contours on Solid Surfaces - Boundary Files

Boundary files contain simulation data recorded at blockage or wall surfaces. Continuously shaded contours are drawn for quantities such as wall surface temperature, radiative flux, etc. Figure 3.12 shows several snapshots of a boundary file animation where the surfaces are colored according to their temperature. Boundary files have file names with extension `.bf` and are displayed by selecting the desired entry from the *Load/Unload* menu. Figure 3.13 shows the same snapshots as in Fig. 3.12 except that data below 200 °C is chopped.

A boundary file containing wall temperature data may be generated by using:

```
&BNDF QUANTITY='WALL TEMPERATURE' /
```

Loading a boundary file is a memory intensive operation. The entire boundary file is read in to determine the minimum and maximum data values. These bounds are then used to convert four byte floats to one byte

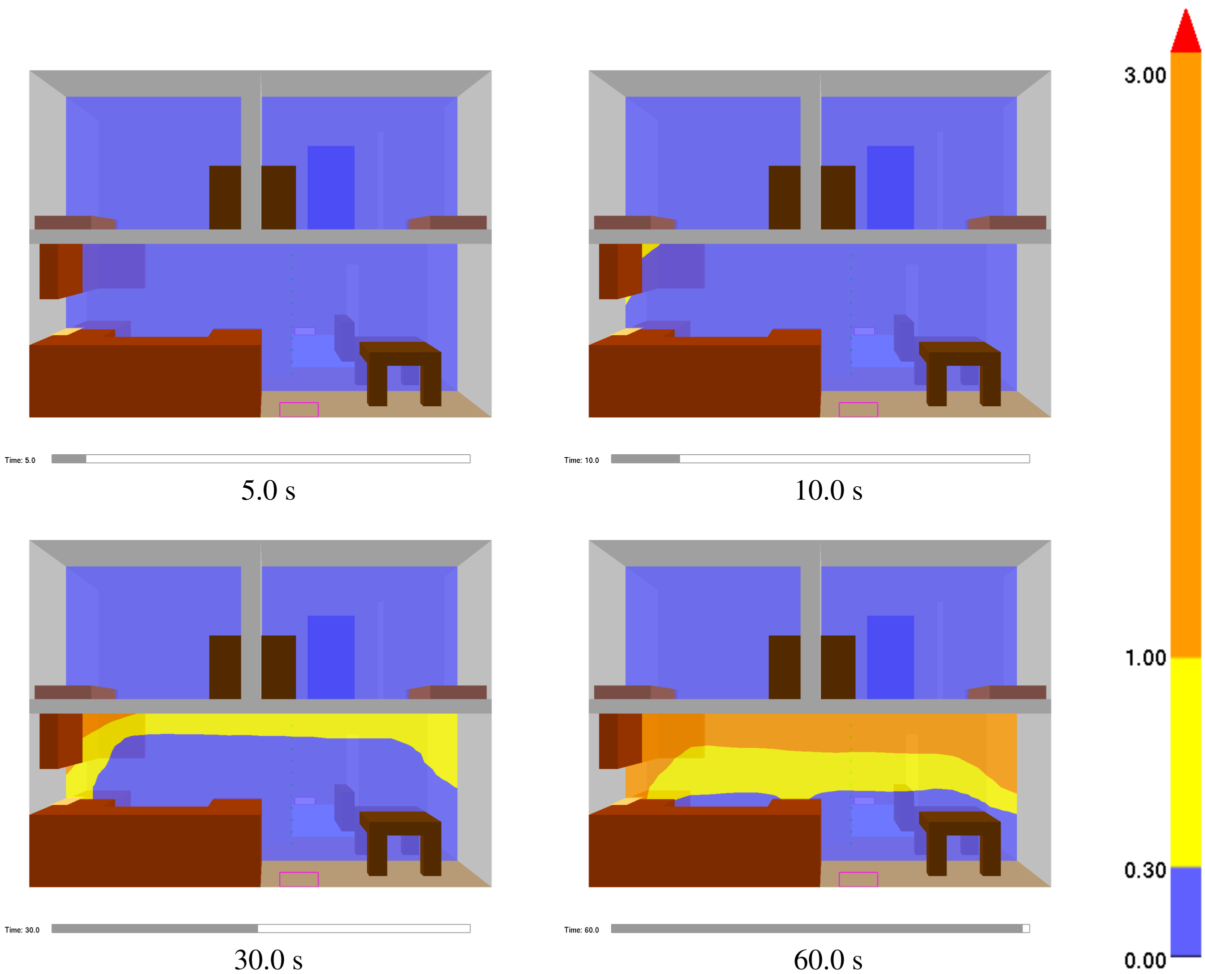


Figure 3.10: FED slices. These contours were generated using CO, CO_2 and O_2 data slices.

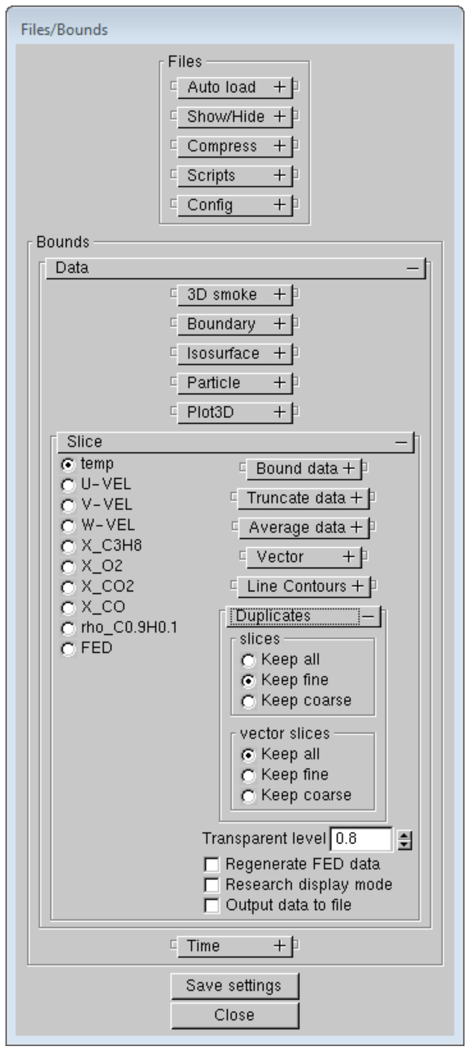


Figure 3.11: Dialog box for specifying duplicate slice visibility.

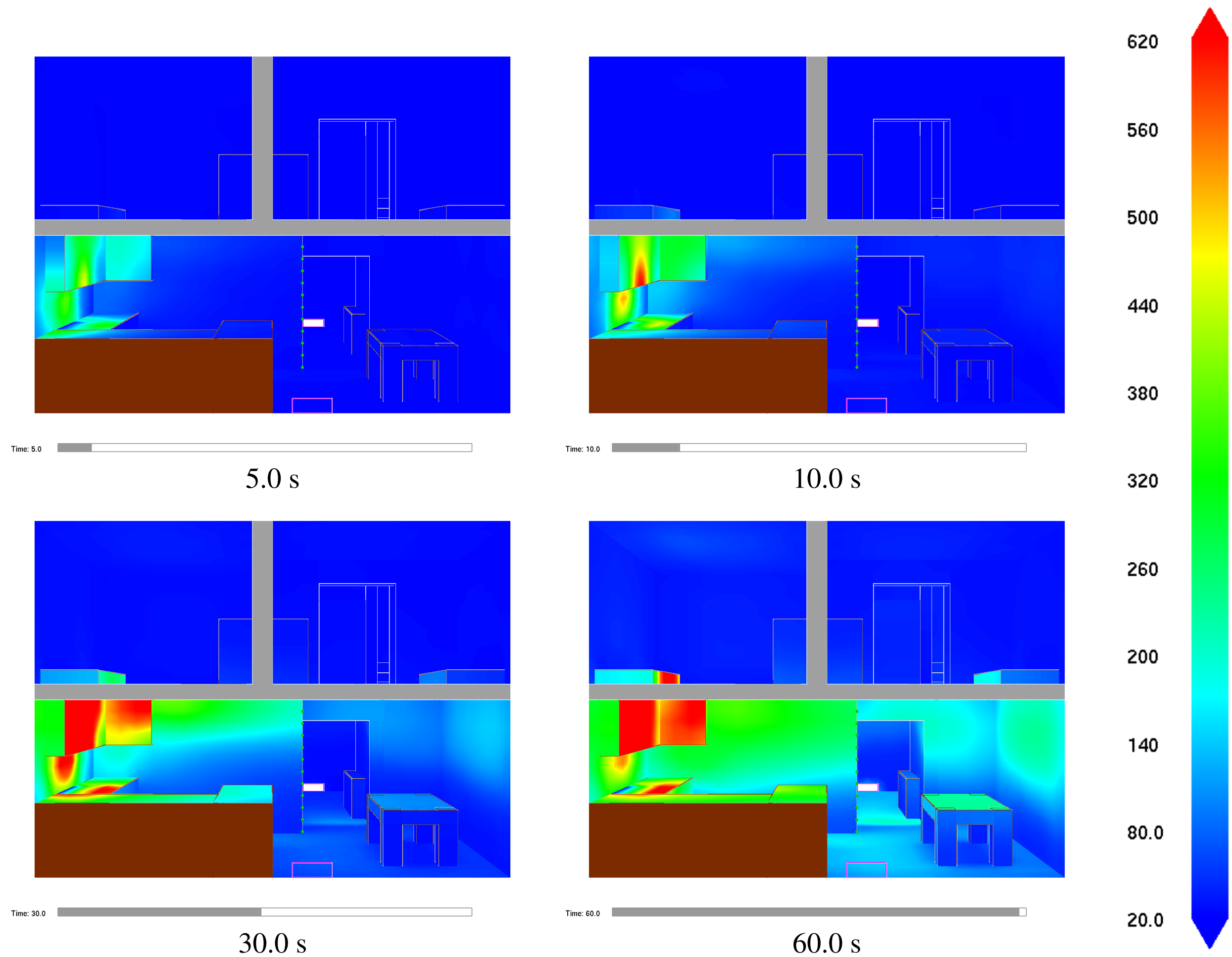


Figure 3.12: Boundary file snapshots of shaded wall temperatures (cell averaged data). These snapshots were generated by using "`&BNDF QUANTITY='WALL_TEMPERATURE'/`".

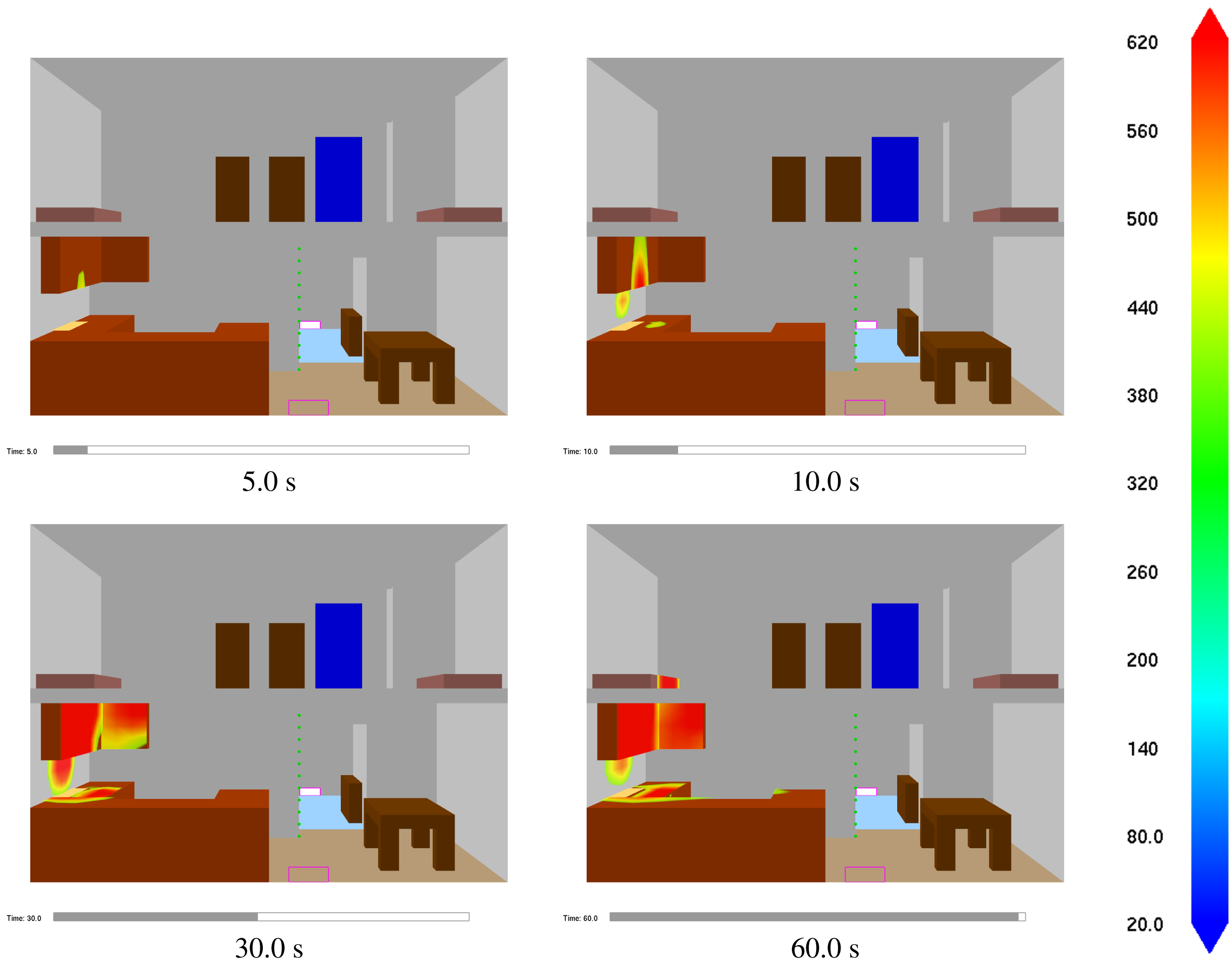


Figure 3.13: Boundary file snapshots of truncated shaded wall temperatures (cell averaged data). Data values are truncated or chopped below 200 °C. These snapshots were generated by using "`&BNDF QUANTITY='WALL_TEMPERATURE'/`".

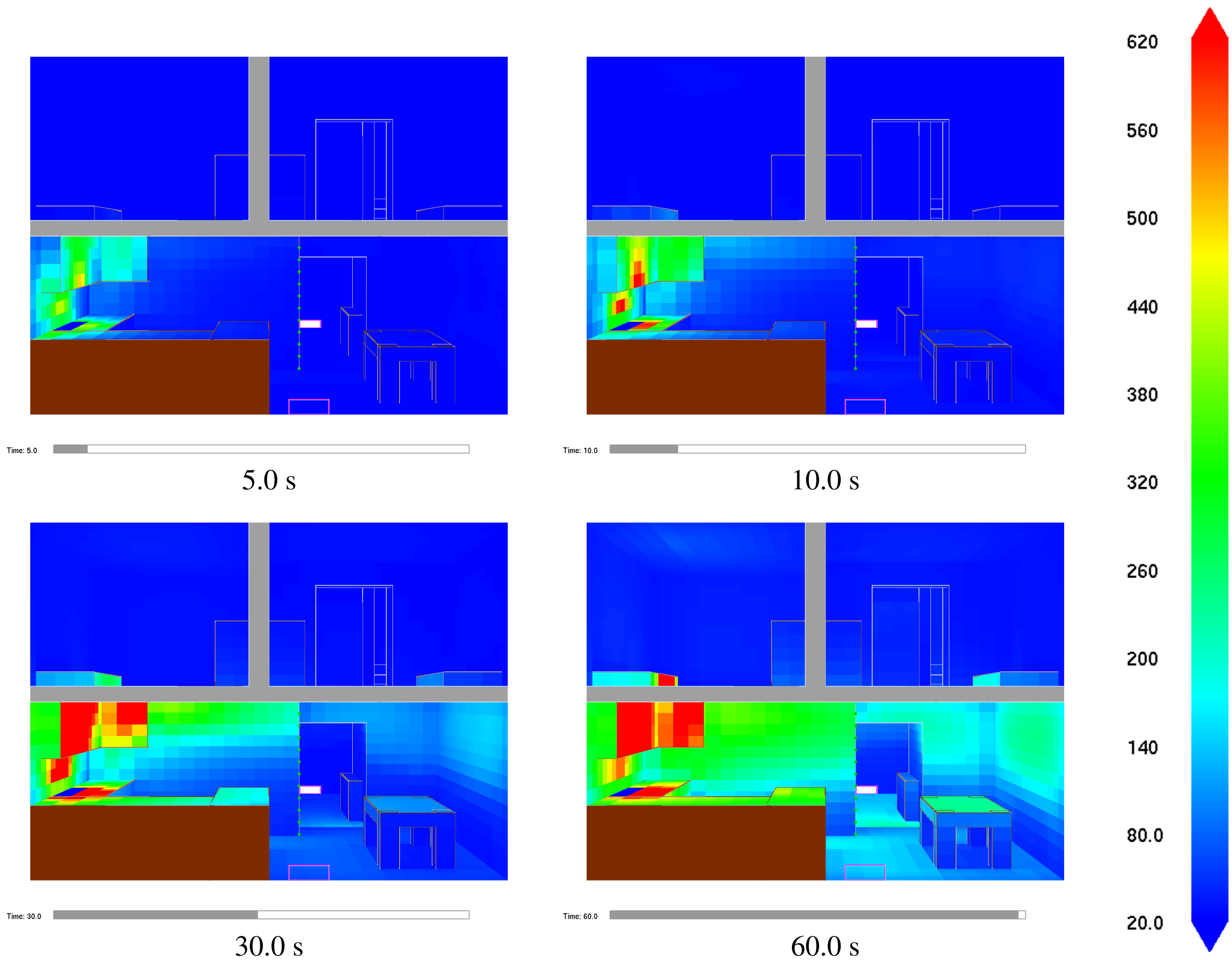
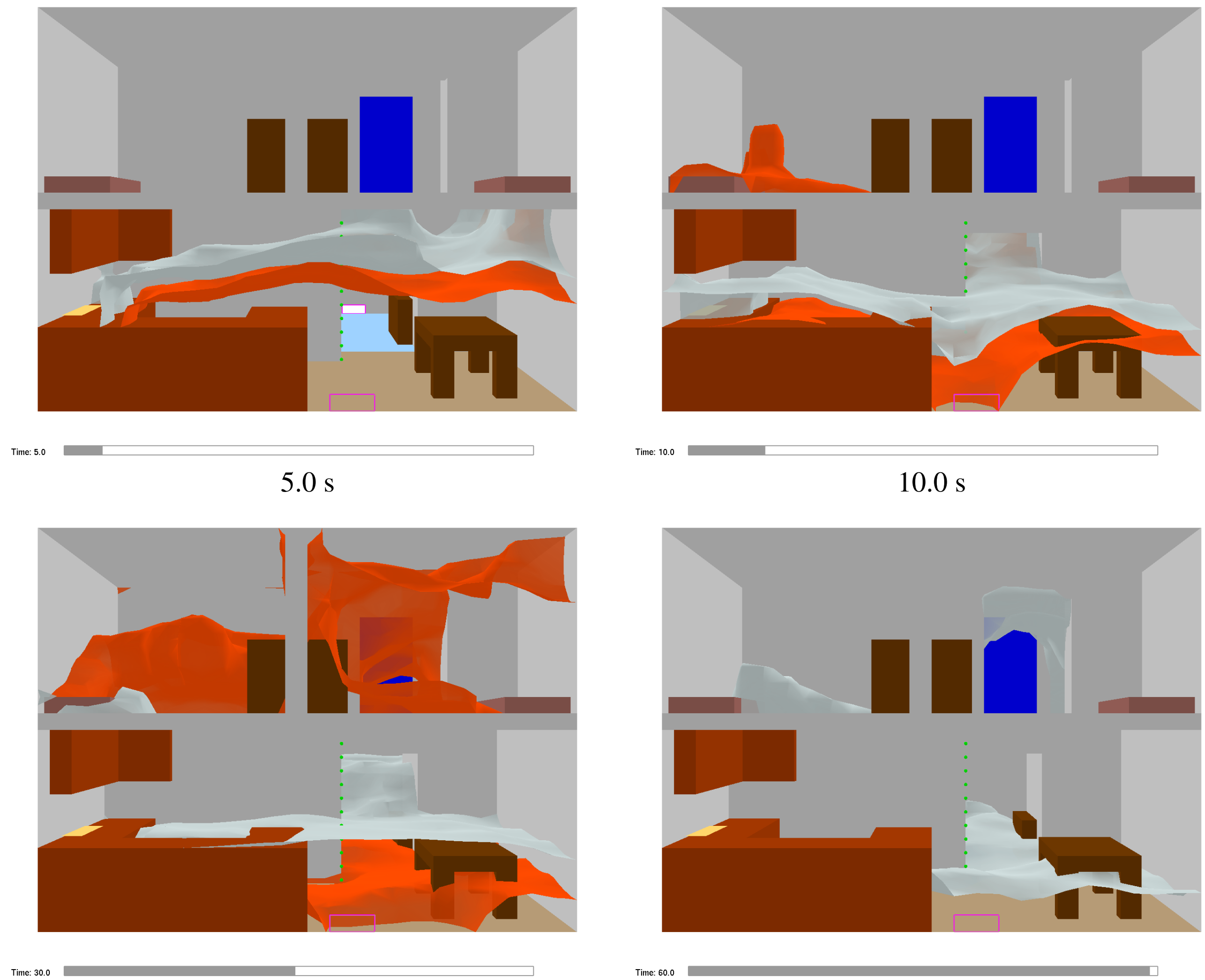


Figure 3.14: Boundary file snapshots of shaded wall temperatures (cell centered data) These snapshots were generated by using "`&BNDF QUANTITY='WALL_TEMPERATURE' CELL_CENTERED=.TRUE. /`".

color indices. To drastically reduce the memory requirements, simply specify the minimum and maximum data bounds using the Set Bounds dialog box. This should be done before loading the boundary file data. When this is done, memory for the boundary file data is allocated for only one time step rather than for all time steps.

3.4 3D Contours - Isosurface Files



5.0 s 10.0 s

30.0 s 60.0 s

Figure 3.15:
Isosurface file snapshots of temperature levels. The orange surface is drawn where the air/smoke temperature is 30 °C and the white surface is drawn where the air/smoke temperature is 100 °C. These snapshots were generated by adding "`&ISOF QUANTITY='TEMPERATURE',VALUE(1)=30.0,VALUE(2)=100.0 /`" to the FDS input file.

The surface where a quantity such as temperature attains a given value is called an isosurface. An isosurface is also called a level surface or 3D contour. Isosurface files contain data specifying isosurface locations for a given quantity at one or more levels. These surfaces are represented as triangles. Isosurface files have file names with extension .iso and are displayed by selecting the desired entry from the *Load/Unload* menu.

Isosurfaces are specified in the FDS input file with the `&ISOF` keyword. To specify isosurfaces for temperatures of 30°C and 100°C as illustrated in Fig. 3.15 add the line:

```
&ISOF QUANTITY='TEMPERATURE', VALUE(1)=30.0, VALUE(2)=100.0 /
```

to the FDS input file. A complete list of isosurface quantities may be found in Ref. [4]

3.4.1 Isosurfaces from particle files

The Smokezip -part2iso option may be used to generate isosurfaces from particle data. Isosurface locations indicate a boundary separating particle and no-particle regions, i.e., wherever particle density is 0.5 particles per grid cell. Isosurface coloring is determined using averaged particle data. Representing particle data with an isosurface is useful when particles are used to model objects such as trees especially when the objects are viewed up close. See Chapter 15.1 for more details on generating isosurface files from particle files. Figure 3.16 shows a snapshot of a fire plume generated using particles and the command

```
smokezip -part2iso plumeiso
```

The plume is visualized using both particles and an isosurface generated from these same particles.

3.4.2 Isosurfaces from fractional effective dose data (generated by Smokeview)

As with 2D slices, Smokeview computes the fractional effective dose (FED) for isosurfaces if 3D slices for CO_2, CO and O_2 are specified in the FDS input file. 3D slices are required to compute isosurfaces. Again, these slices need to be specified at the same location as in

```
&SLCF XB=0.0,1.6,0.0,1.6,0.0,3.2,QUANTITY='VOLUME FRACTION' SPEC_ID='CARBON DIOXIDE' /
&SLCF XB=0.0,1.6,0.0,1.6,0.0,3.2,QUANTITY='VOLUME FRACTION' SPEC_ID='CARBON MONOXIDE'
    /
&SLCF XB=0.0,1.6,0.0,1.6,0.0,3.2,QUANTITY='VOLUME FRACTION' SPEC_ID='OXYGEN' /
```

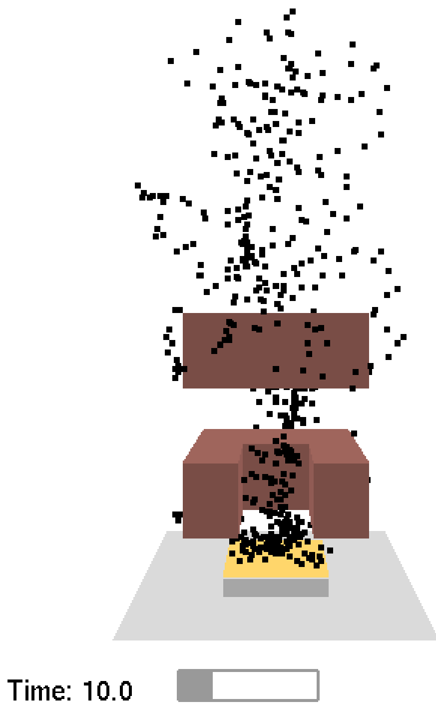
Figure 3.17 illustrates an FED isosurfaces where the three levels are at 0.3 (blue), 1.0 (yellow) and 3.0 (red).

3.5 Device data - .csv files

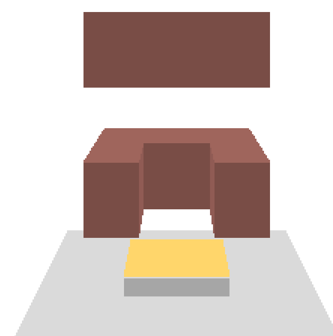
Spreadsheet data, generated by either FDS or CFAST or imported from some other source, may be visualized by Smokeview. Version 6 of both FDS and CFAST both generate spreadsheet files using the same file format given by

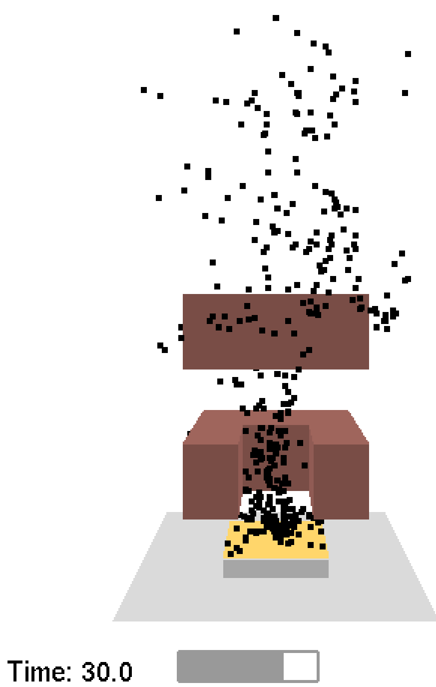
```
unit1,unit2, ..., unitN
label1,label2, ..., labelN
data11,data12, ..., data1N
data21,data22, ..., data2N
....
datam1,datam2, ..., dataMN
```

where the `unit` and `label` entries are character strings and the `data` entries are floating point numbers.



particles at 10.0 s


particle isosurface at 10.0 s



particles at 30.0 s

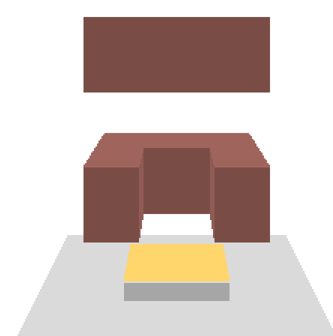

particle isosurface at 30.0 s

Figure 3.16: Fire plume visualized using particles and isosurfaces generated from particles.

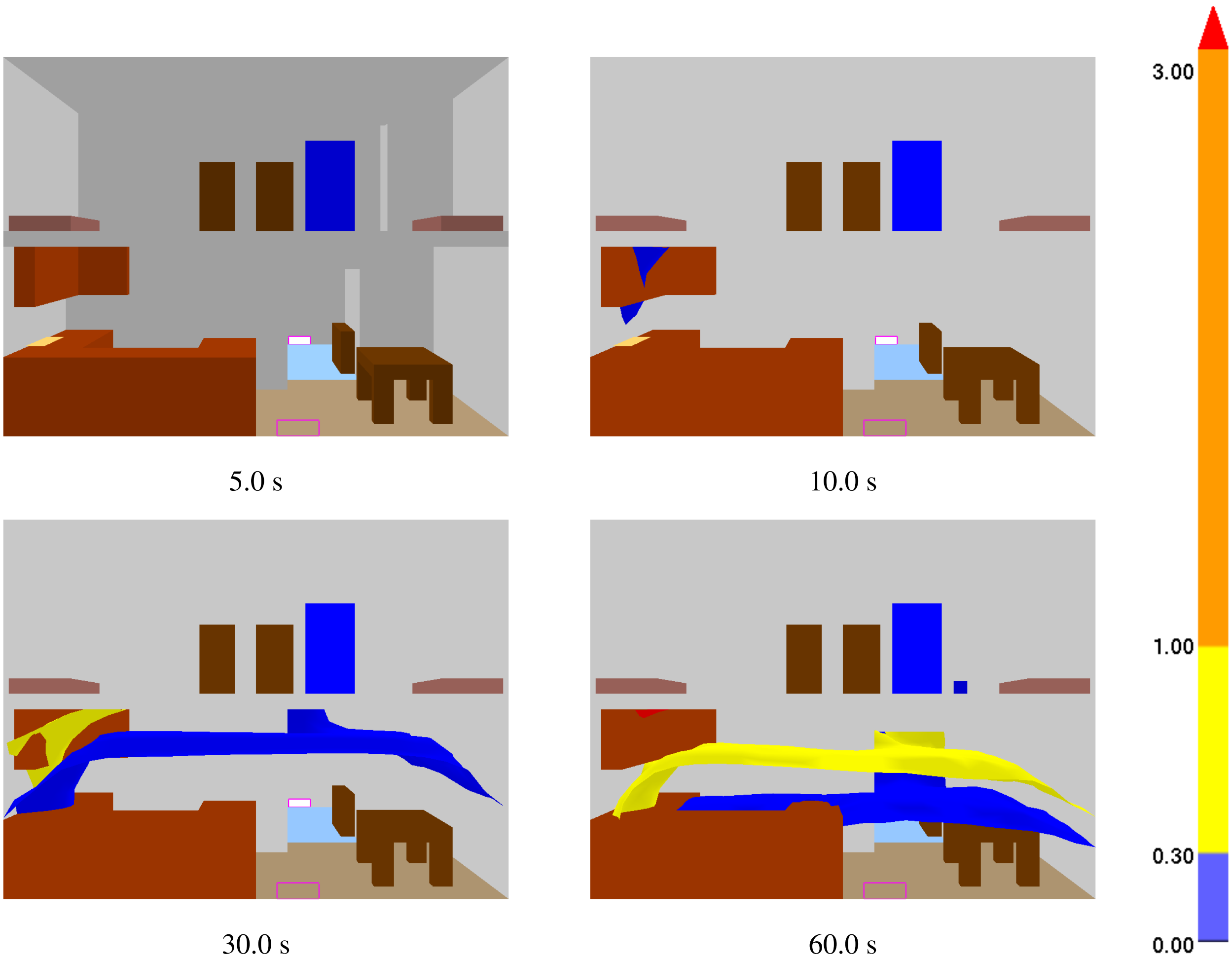


Figure 3.17: FED isosurfaces. These level surfaces were generated using CO, CO2 and O2 3D data slices. The blue, yellow and red surfaces represent where the fed values are 0.3, 1.0 and 3.0 respectively.

FDS uses spreadsheet files to store device and heat release data. CFAST uses spreadsheet files to store the results of the simulation (room pressures, layer heights, layer temperatures, etc.). To view spreadsheet data generated by FDS, open the Devices/Objects dialog box illustrated in Fig. 3.18 and select the *Show values* checkbox. If U, V and/or W velocity data is contained in the spreadsheet file then velocity vectors may also be displayed. Figures 3.19 and 3.20 illustrates velocity visualization using arrows and continuous profiles.

Flow vectors may be visualized as lines, arrows, smokeview objects or continuous profiles. The length and diameter of vector lines may be specified. In addition, if arrows are selected, the length and diameter of the arrow head may be specified. The following `&DEVC` lines give an example of defining device flow vectors.

```
&DEVC XYZ=3.7,2.0,0.2 QUANTITY='U-VELOCITY' /
&DEVC XYZ=3.7,2.0,0.2 QUANTITY='W-VELOCITY' /
&DEVC XYZ=3.7,2.0,0.2 QUANTITY='TEMPERATURE' /
```

3.6 Static Data - Plot3D Files

Data stored in Plot3D files use a format developed by NASA [24] and are used by many CFD programs for representing simulation results. Plot3D files store five data values at each grid cell. FDS uses Plot3D files to store temperature, three components of velocity (U, V, W) and heat release rate. Other quantities may be stored if desired.

An FDS simulation will automatically create Plot3D files at several specified times throughout the simulation. Plot3D data is visualized in three ways: as 2D contours, vector plots and isosurfaces. Figure 3.21a 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 3.21b shows vectors. The vector lengths may be adjusted by depressing the “a” key. Figure 3.22 gives an example of isosurfaces. Plot3D data are stored in files with extension `.q`.

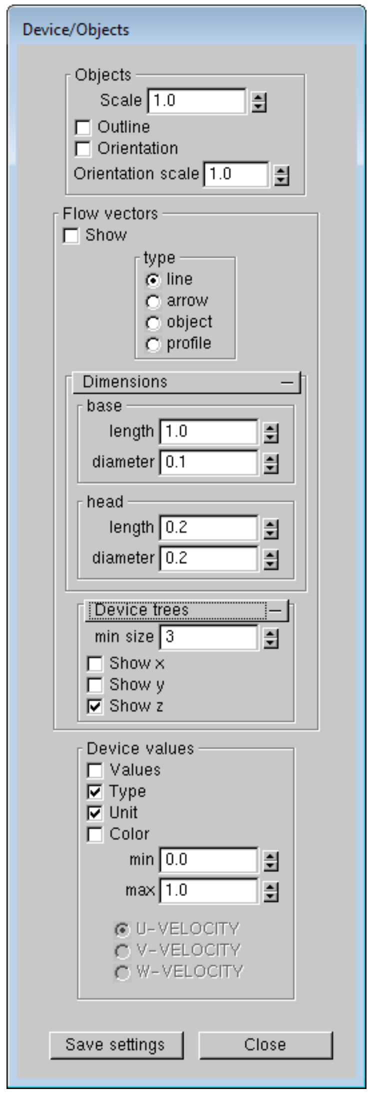


Figure 3.18: A dialog box for displaying device data values stored in FDS formatted spreadsheet files.

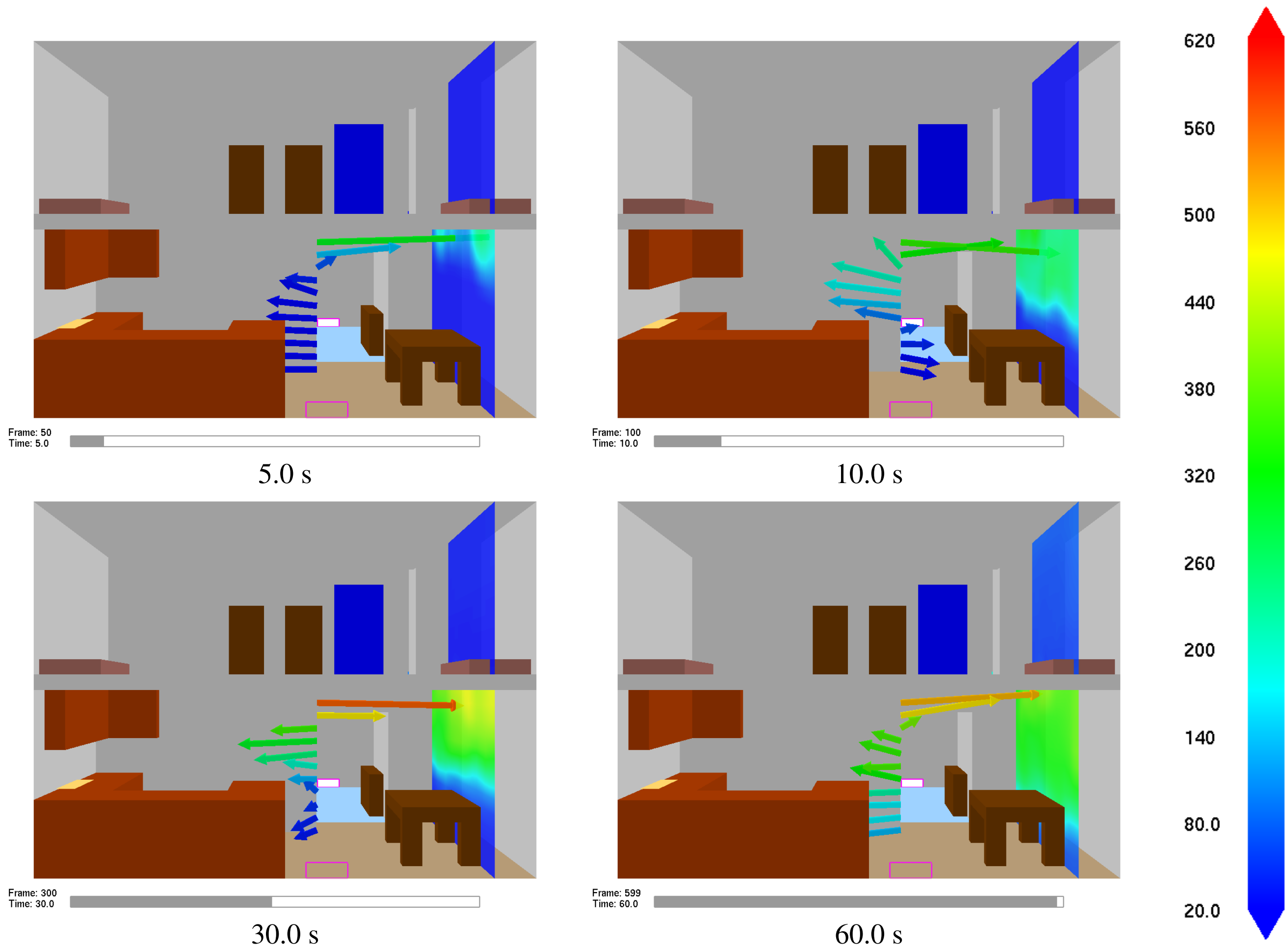
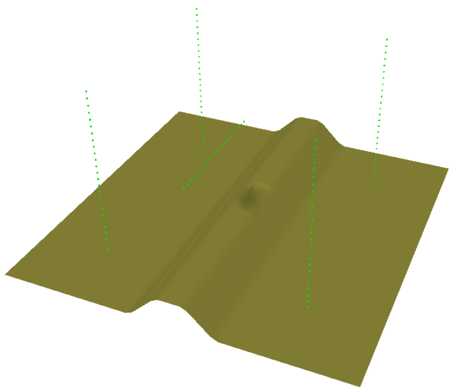
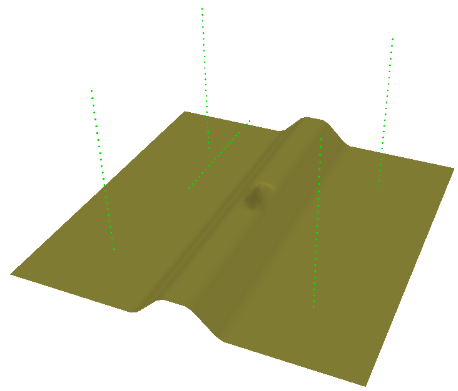


Figure 3.19: Visualization of device flow vectors. Device data may be visualized as colored flow vectors by defining QUANTITY='U-VELOCITY', 'V-VELOCITY' and/or 'W-VELOCITY' keywords on &DEVC namelists each placed at the same XYZ location. The flow vectors may be colored by using QUANTITY='TEMPERATURE' (or some other quantity).

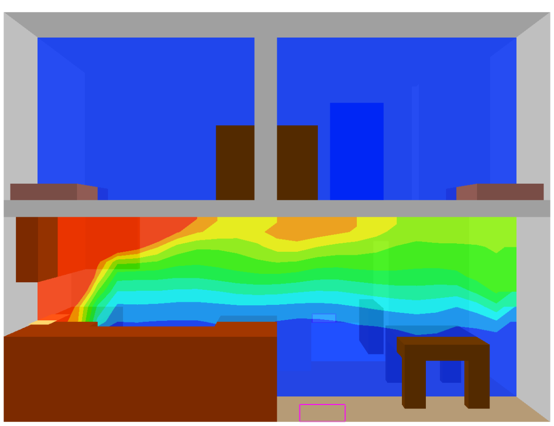


arrows

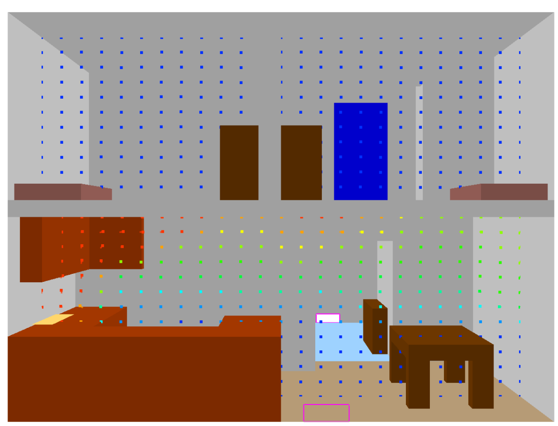


profile

Figure 3.20: Velocity visualization using arrows and continuous profiles.

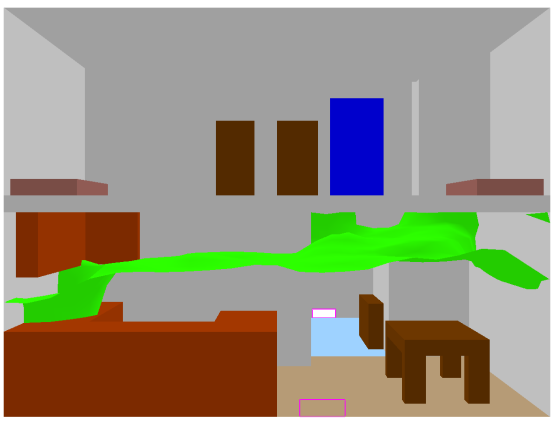


a) shaded 2D temperature contour plots in a vertical plane through the fire

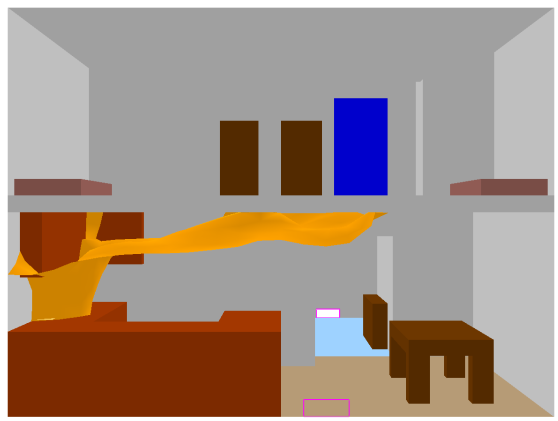


b) shaded temperature vector plot in a vertical plane through the fire. The “a” key may be depressed to alter the vector sizes. The “s” key may be depressed to alter the number of vectors displayed.

Figure 3.21: Plot3D contour and vector plot examples.



a) temperature isosurface at 350 °C



b) temperature isosurface at 530 °C

Figure 3.22: Plot3D isocontour example.

Chapter 4

Visualizing Zone Fire Data

Smokeview may be used to visualize data simulated by a zone fire model. The zone fire model, CFAST [7], creates data files containing geometric information such as room dimensions and orientation, vent locations, etc. It also outputs modeling quantities such as pressure, layer interface heights, and lower and upper layer temperatures. Smokeview visualizes the geometric layout of the scenario. It also visualizes the layer interface heights, upper layer temperature and vent flow. Vent flow is computed internally in Smokeview using the same equations and data as used by CFAST. For a given room, pressures , P_i, are computed at a number of elevations, h_i using

$$P_i = P_f - \rho_L g \min(h_i, y_L) - \rho_U g \max(h_i - y_L, 0) \qquad (4.1)$$

where P_f is the pressure at the floor (relative to ambient), ρ_L and ρ_U are the lower and upper layer densities computed from layer temperatures using the ideal gas law and g is the acceleration of gravity. When densities vary continuously with height, this becomes $P_i = P_f - \int_0^h \rho(z) g \, \mathrm{d}z$. A pressure difference profile is then determined using pressures computed on both sides of the given vent.

In the visualization, colors represent the gas temperature of the vent flow. The colors change because the flow may come from either the lower (cooler) or upper (hotter) layer. The length and direction of the colored vent flow region represents a vent flow speed and direction. Plumes are represented as inverted cones with heights calculated in Smokeview using the same correlation as CFAST and heat release rate data computed by CFAST. A Smokeview view of the one room sample case that comes with the CFAST installation is illustrated in Figs. 4.1, 4.2 4.3 and 4.4.

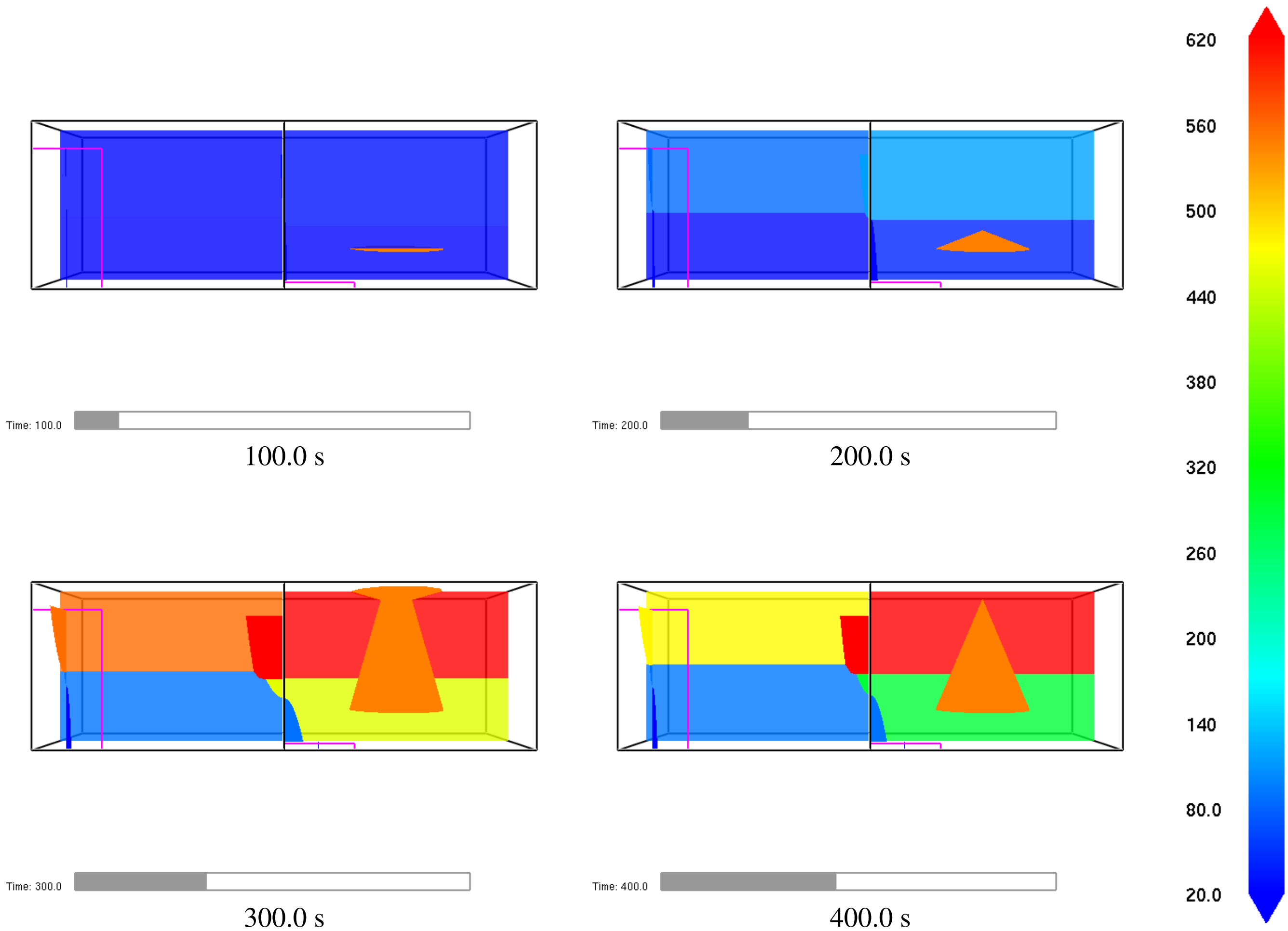


Figure 4.1: CFAST test showing upper/lower layer temperatures and vent flow visualized using color.

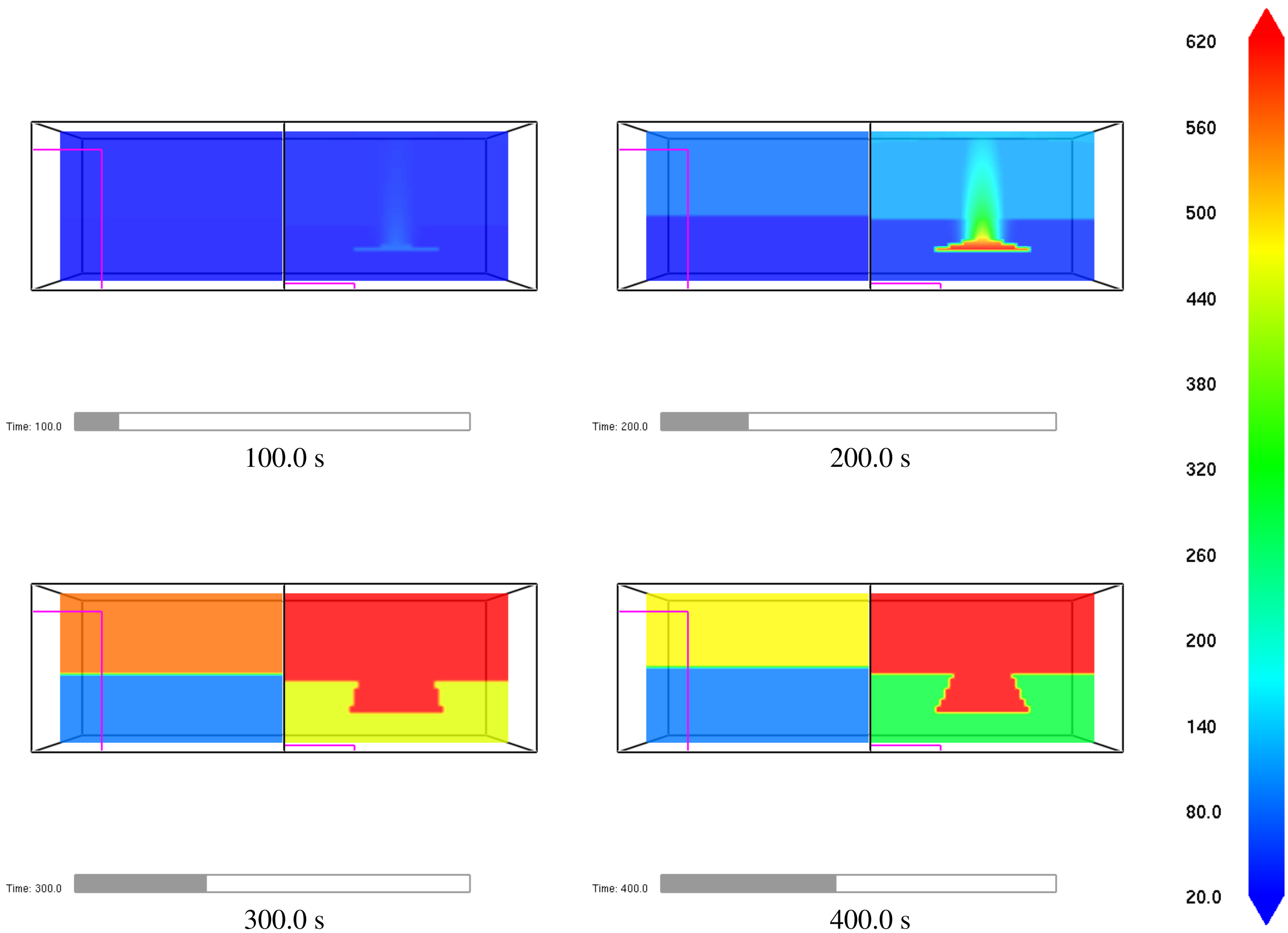


Figure 4.2: CFAST test showing upper/lower layer, plume and ceiling jet temperatures, and vent flow visualized using color.

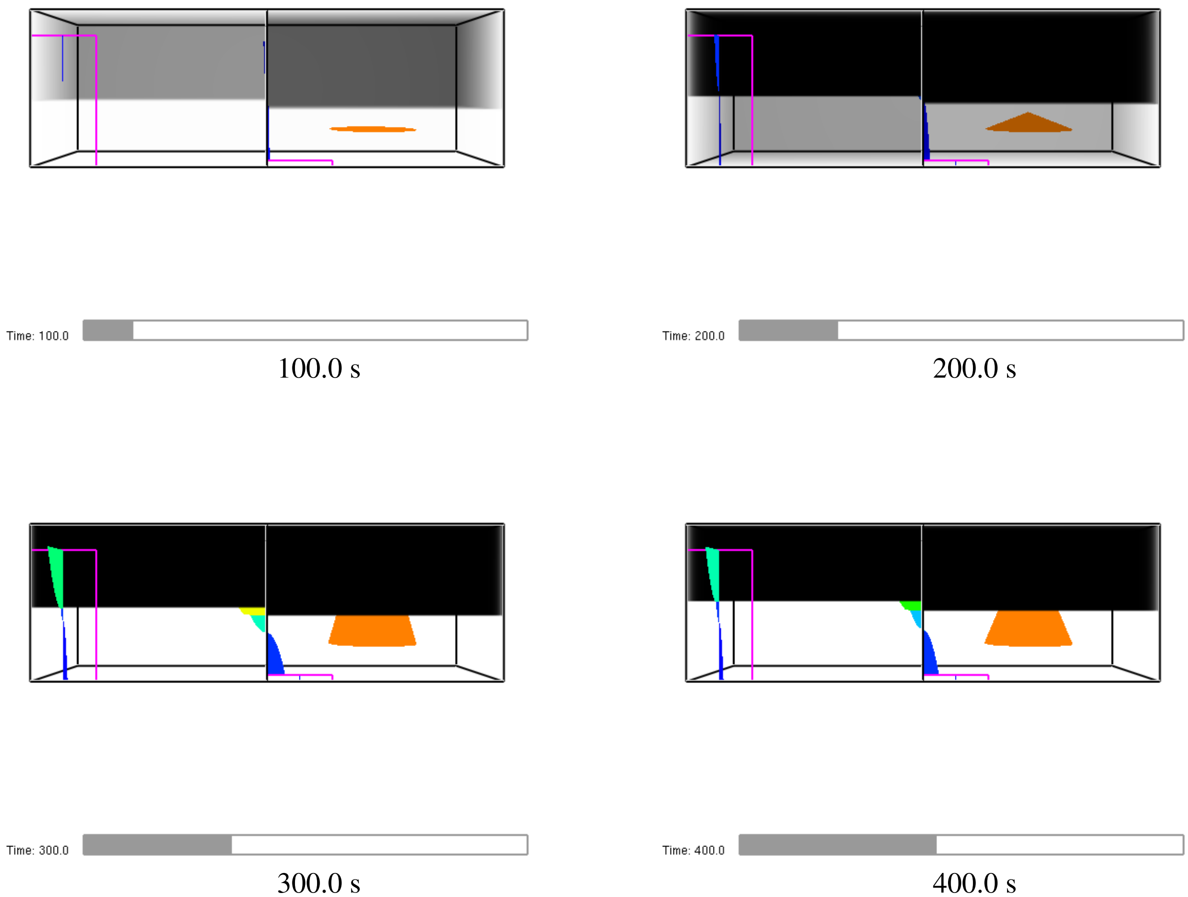


Figure 4.3: CFAST test showing upper/lower layer temperatures and vent flow. Layers are visualized realistically and vent flow is visualized using color.

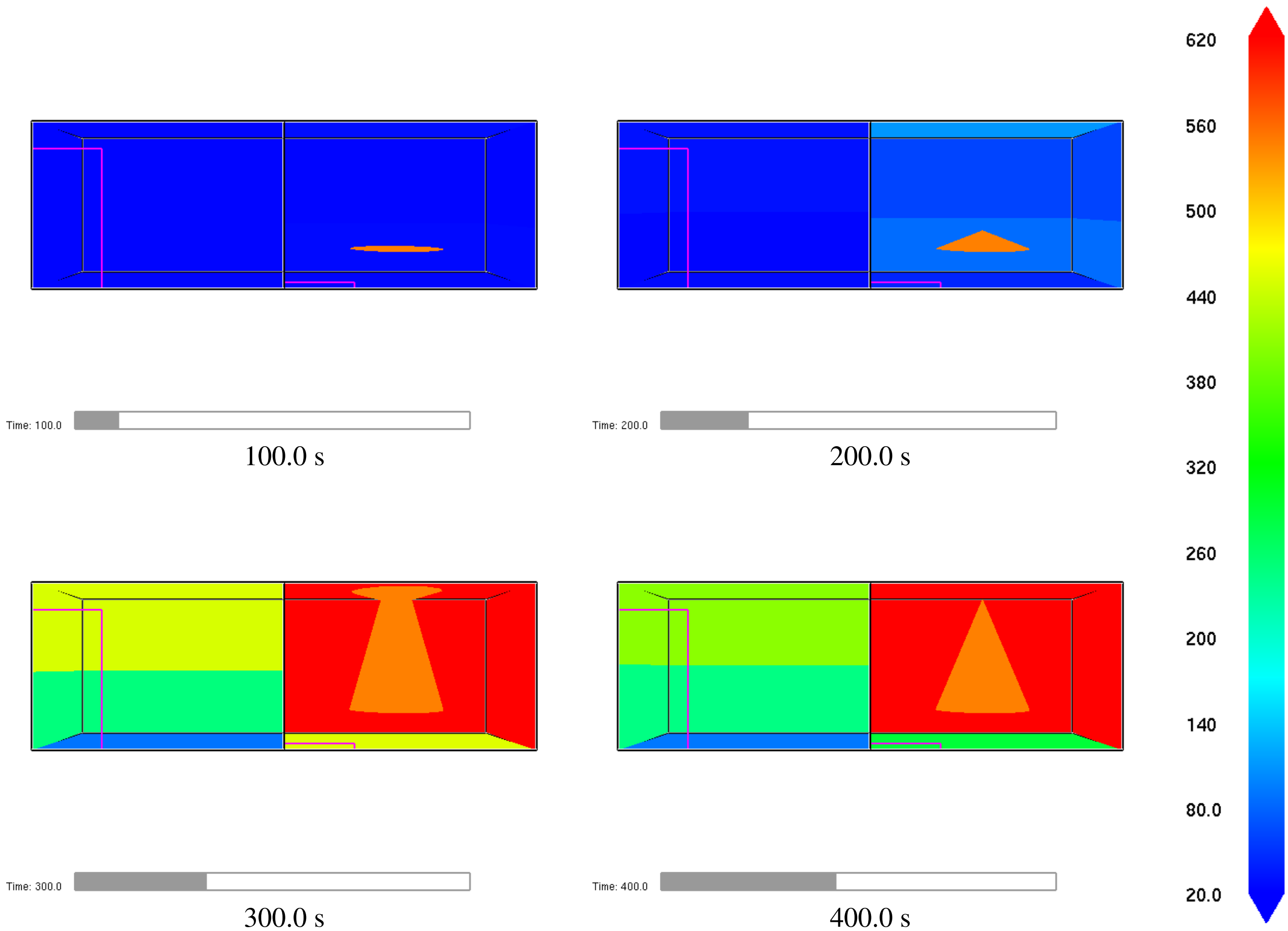


Figure 4.4: CFAST test showing wall temperatures visualized using color.

Part II

Controlling and Customizing Smokeview

Chapter 5

Setting Options

5.1 Data Bounds

In order to visualize data, Smokeview maps data values to colorbar indices ranging from 0 to 255 using a mapping of the form

$$c = 255\frac{v - v_{\min}}{v_{\max} - v_{\min}}$$

where v is a data value and c is a colorbar index (values less than $v_{\min}$ are mapped to 0 and values greater than $v_{\max}$ are mapped to 255). The terms $v_{\min}$ and $v_{\max}$ are set to be either global min/max data values (0'th and 100'th percentile values from a data histogram computed by Smokeview), percentile min/max values (1st and 99th percentile values) or min/max values specified by the user. This choice is set in the Data bounds dialog box as illustrated in Figure 5.1. Creating and modifying colorbars are discussed in Chapter 9. User specified min/max values may be used to ensure consistent color shading when displaying several data files simultaneously. To quickly set bounds for actual data, press `ALT r` and reload or update data files. This puts Smokeview into *research mode* which uses actual global min/max bounds when mapping data to color and does not smooth colorbar labels.

The Data bounds dialog box is opened from the *Dialogs>Data bounds* menu. Each file type in Fig. 5.1 (slice, particle, Plot3D, etc.) has a set of *radio buttons* for selecting the variable type and radio buttons for bounding and truncating data when converting data to color. 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. Radio buttons adjacent to the edit boxes determine what type of bounds should be applied. The `Update` button for slice files and `Reload` button for boundary and PLOT3D files are pressed to make new bounds take effect.

The `Plot3D` and `Slice File` portions of the File/Bounds dialog box have additional controls used to chop or hide data. The settings used in Fig. 5.1 were used to generate the ceiling jet visualized in Fig. 5.2. Data values less than 140 °C are chopped or not drawn in the figure.

Slice file data may be time averaged or smoothed over a user selectable time interval. This option is also implemented from the Slice File section of the File/Bounds dialog box (see Fig. 5.1).

The `Plot3D` portion of the File/Bounds dialog box as illustrated in Figure 5.3 has controls for specifying how Plot3D vectors and isosurfaces appear.

The bounds dialog for Plot3D display allows one to select between three different types of contour plots: shaded, stepped and line contours.

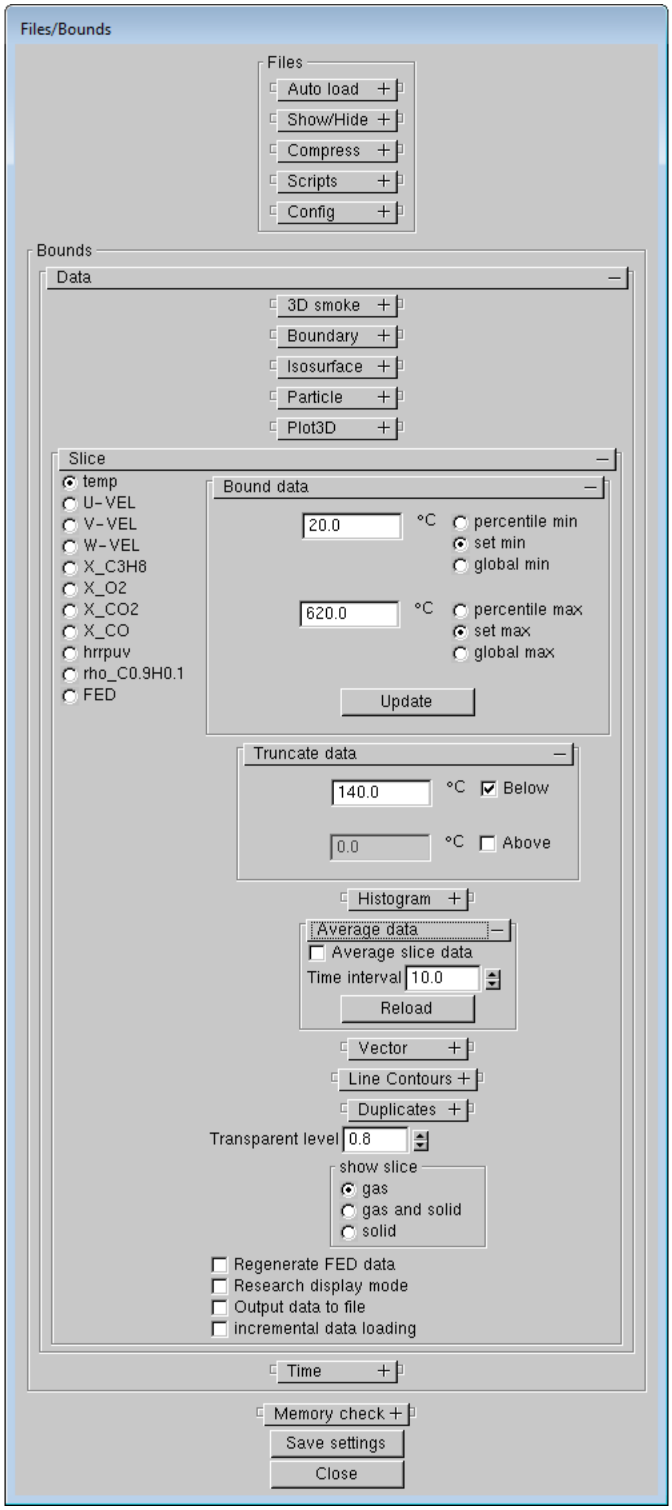


Figure 5.1: Dialog box for setting Slice file data bounds. Select a variable and bound type (percentile, set or global). Enter a lower and/or upper bound if set bound type was selected. Data may be excluded from the plot by selecting a *Truncate data* bound. Press the *Update* button for the new bounds to take effect.

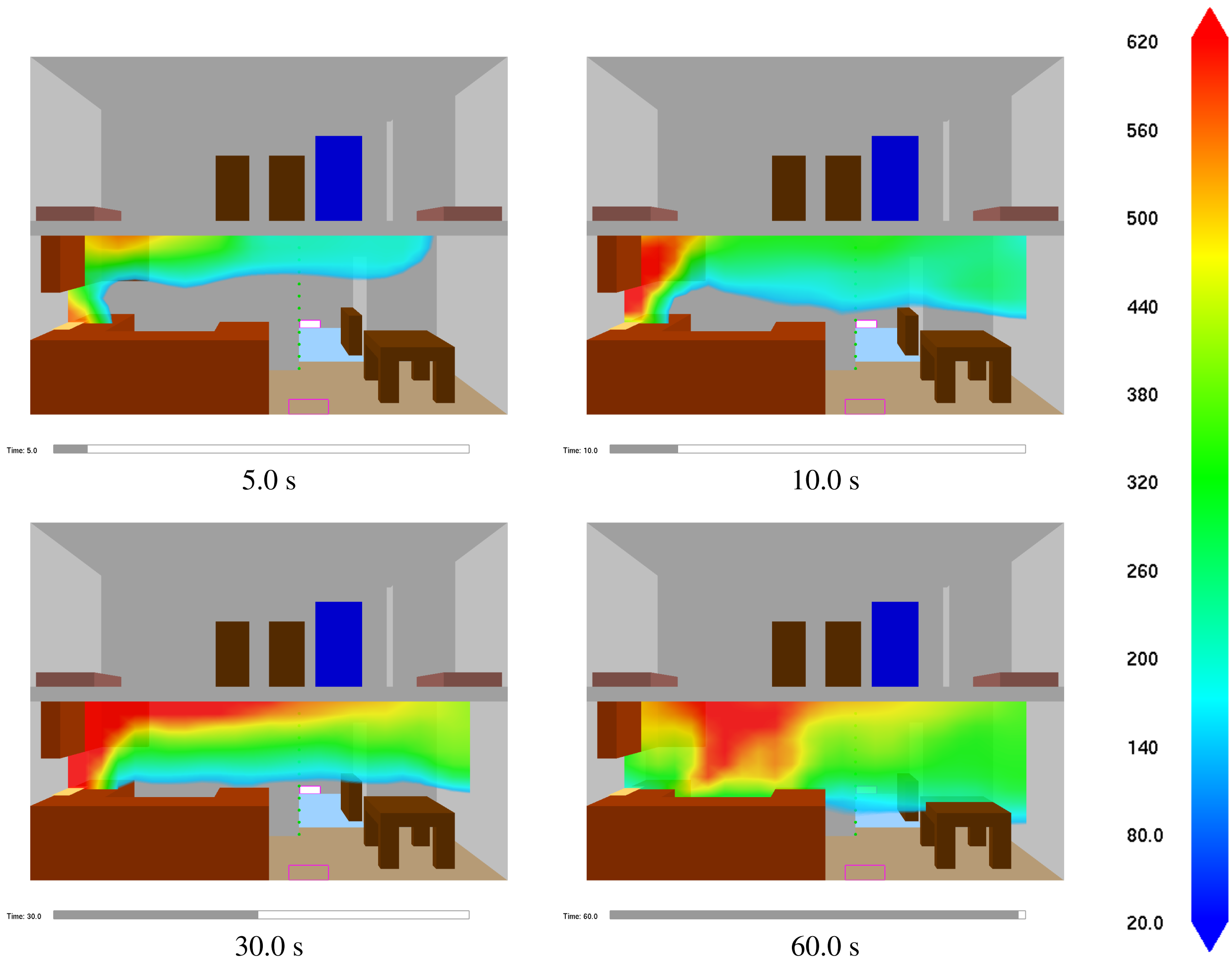


Figure 5.2: Ceiling jet visualization created by *chopping data* below 140 °C using the Bounds dialog box as illustrated in Fig. 5.1.

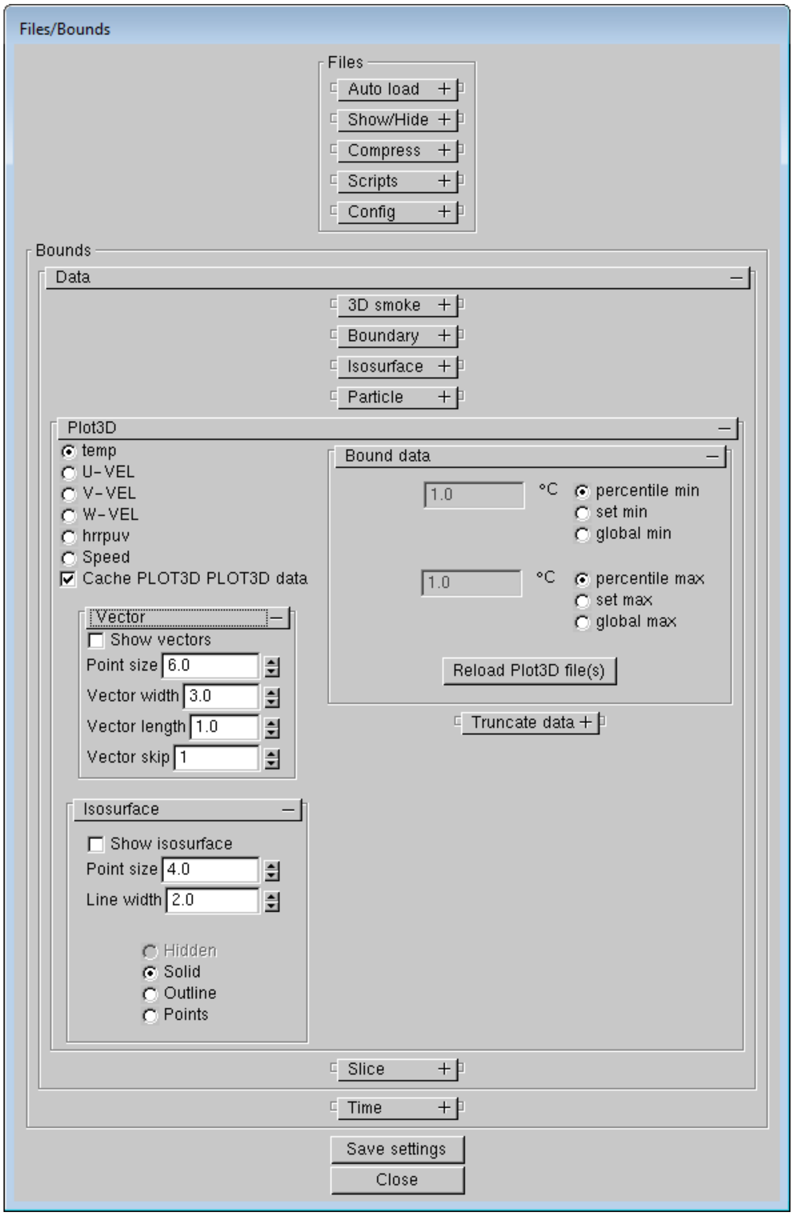


Figure 5.3: Dialog box for setting Plot3D file options. Select a variable and bound type. Enter a lower and/or upper bound if set bound type was selected. Data may be excluded from the plot by selecting a *Truncate data* bound. Select the type of contour plot to be displayed. Press the *Reload Plot3D files(s)* button for the new bounds to take effect.

The Boundary File portion of the File/Bounds dialog box has an *Ignition* checkbox which allows one to visualize when and where the blockage temperature exceeds its ignition temperature.

The Particle file portion of the File/bounds dialog box as illustrated in Figure 5.4 has controls for specifying whether particle files are loaded in parallel. The user may also specify the number files that are loaded simultaneously

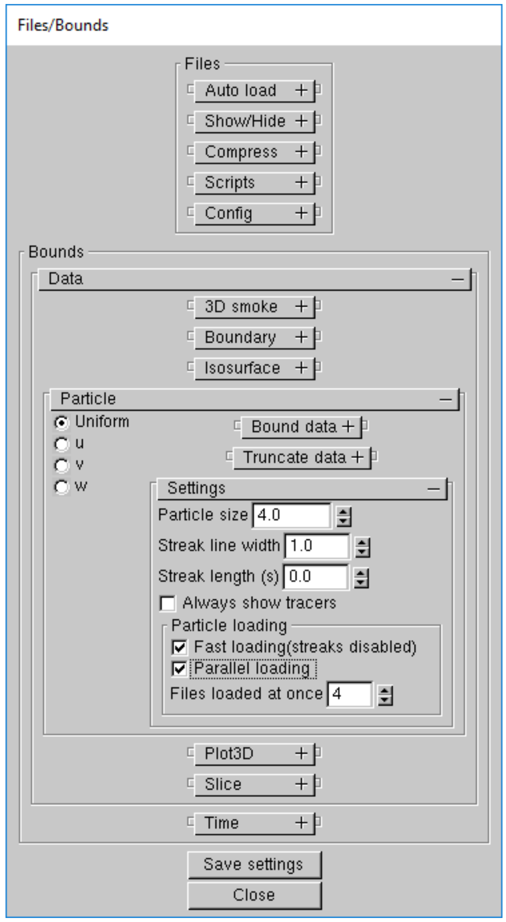


Figure 5.4: Dialog box for setting Particle file options. Select a variable and bound type. Enter a lower and/or upper bound if set bound type was selected. Particle files may be loaded in parallel by selecting the Parallel loading checkbox and setting the number of files to load in parallel.

5.2 3D Smoke Options

Figures 5.5 and Figure 5.6 show a dialog box for controlling the display of slice and volume rendered smoke. The user may specify parameters such as fire and smoke color, smoke albedo and an hrrpuv cutoff value used to determine what is colored as smoke and fire. The user may also specify cutoff values used to determine when to load smoke and fire data files. Red, green and blue color values range between 0 and 255. The

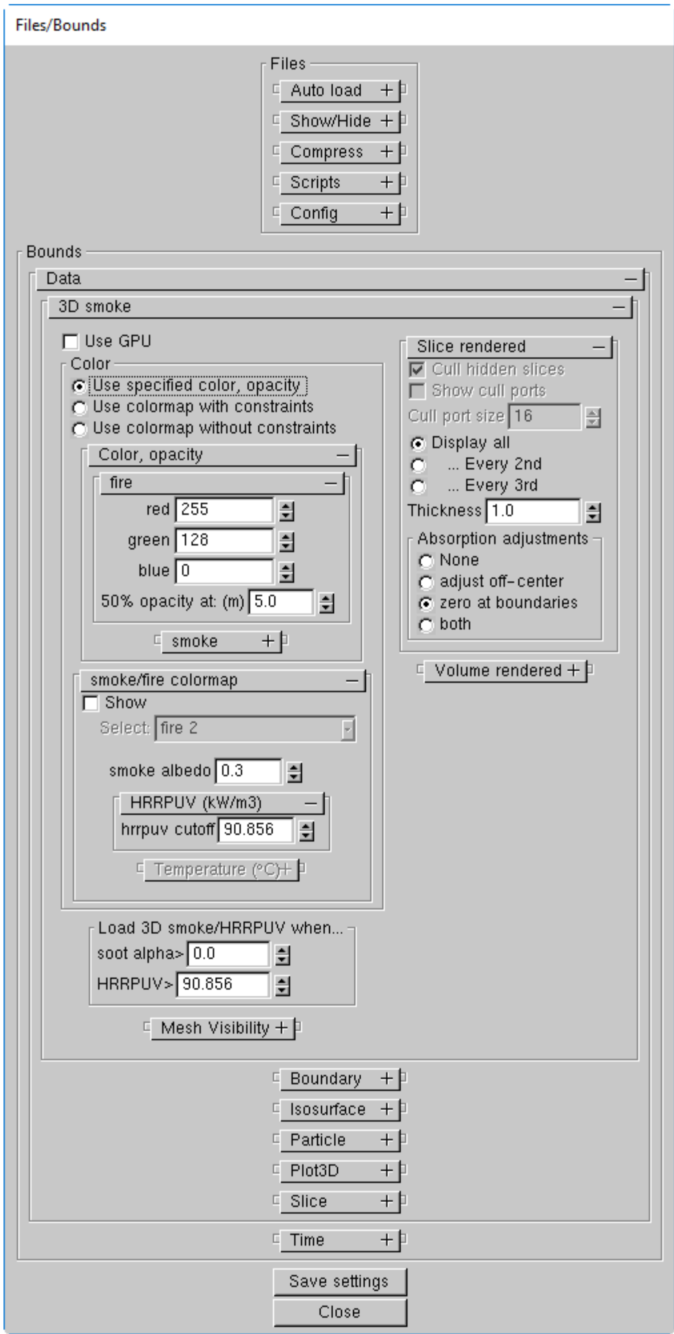


Figure 5.5: Dialog box for setting slice rendered 3D smoke options. Fire color may be specified for hrrpuv values above a specified cutoff. Smoke color may be specified for hrrpuv values below the same cutoff.

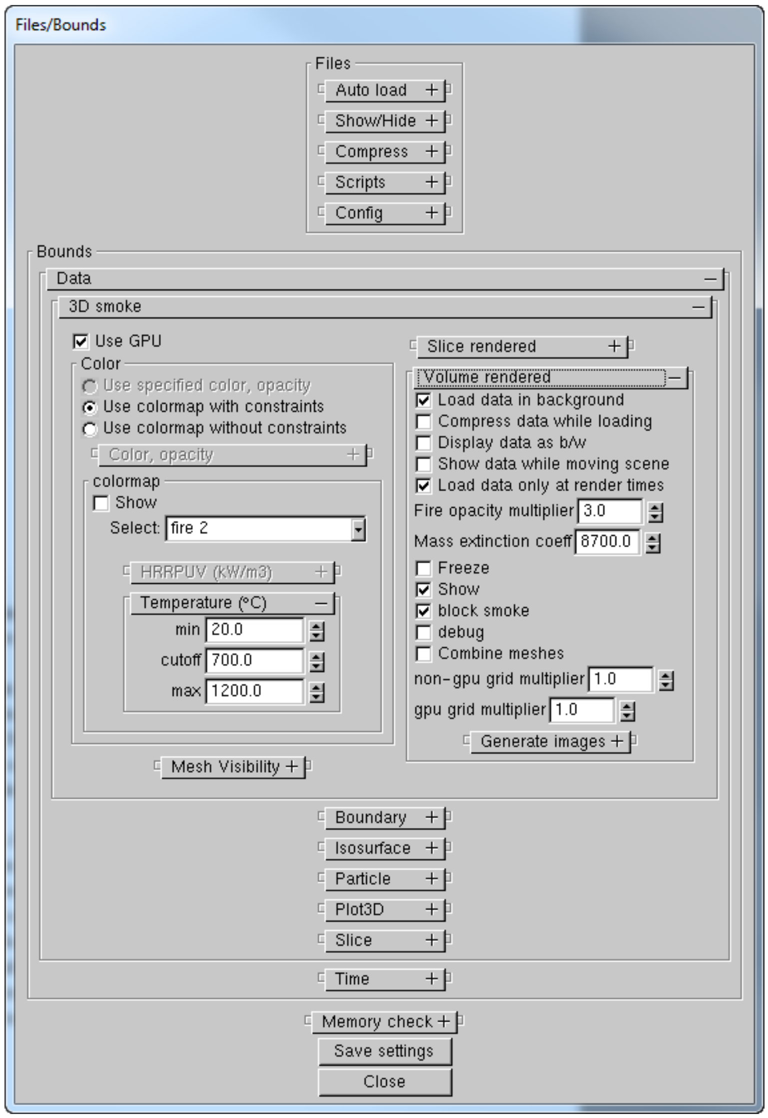


Figure 5.6: Dialog box for setting volume rendered 3D smoke options. Fire color may be specified for temperature values above a specified cutoff. Smoke color may be specified for hrrpuv values below the same cutoff.

hrrpuv cutoff parameter refers to the heat release rate required before Smokeview will color a node as fire rather than smoke. The *50% flame depth* allows one to specify the transparency or optical thickness of the fire (for visualization purposes only). A small value results in opaquely drawn fire while a large value results in a transparently drawn fire. The *Absorption Parameter* setting refers to how the smoke slices are drawn. The *adjust off-center* setting causes Smokeview to account for non-axis aligned paths. The *adjust off-center + zero at boundary* accounts for off center path lengths and zeros smoke density at boundaries in order to remove graphical artifacts.

5.3 Plot3D Viewing Options

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[1] different simulated quantities, view flow vectors and iso or 3D contours. Plot3D file visualization is initiated by selecting the desired entry from the *Load/Unload* `Plot3D` sub-menu and as with time files one may change color and transparency characteristics.

5.3.1 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 the *Solution Variable* sub-menu of the *Show/Hide* menu.

5.3.2 Iso-Contours

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

5.3.3 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 default. If the "`s`" is depressed then every other vector is skipped.

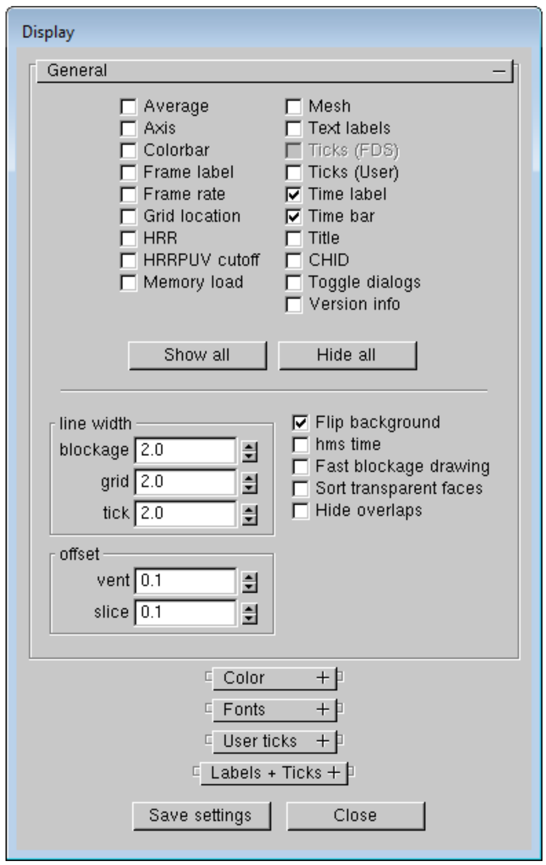


Figure 5.7: Dialog Box for setting miscellaneous Smokeview scene properties.

5.4 Display Options

5.4.1 General

The Display dialog box, illustrated in Fig. 5.7, allows one to set various options to control the scene display such as toggling the visibility of the colorbar, timebar, title *etc.*. The dialog box may be invoked by selecting the *Dialogs>Display* menu item.

5.4.2 Setting window parameters

Controls in the *Window Properties* region of the Motion/View/Render dialog box as illustrated in Fig. 5.8, allow one to change the scene magnification or zoom factor, the projection method used to draw objects (perspective or size preserving) and the window size. Perspective and size preserving projections differ in how objects are displayed at a distance. A perspective projection for-shortens or draws an object smaller when at a distance. An isometric or size preserving projection on the other hand draws objects the same size regardless of where it occurs in the scene.

The *zoom* and *aperture* edit boxes allow one to change the magnification of the scene or equivalently the angle of view across the scene. The relation between these two parameters is given by

$$\text{zoom} = \tan(45^\circ/2)/\tan(\text{aperture}/2) \quad (5.1)$$

A default aperture of 45° is chosen so that Smokeview scenes have a normal perspective.

The size selection list gives the user several pre-defined choices for changing window size or one may alter the width and height spinners to construct a window with a custom size.

5.4.3 Scaling Scenes

Controls in the *Scaling Depth* portion of the Motion/View dialog box, as illustrated in Fig. 5.9, allow one to scale the Smokeview scene. The x, y and z scene dimensions may be scaled independently. For example, a tunnel scenario could be scaled to make the tunnel's *long* dimension appear the same size on the screen as the *height* dimension. The near and far depth planes are used by OpenGL for setting up the depth buffer which in turn is used for determining when objects hidden by *closer* objects.

5.4.4 Stereo

Smokeview implements several methods for displaying scenes in stereo or 3D. These methods are temporal (sending odd frames to the left eye and even frames to right eye), spatial (drawing two frames side by side) and color (super imposing red/blue or magenta/cyan frames). Each method then creates two versions of the scene, one version for each eye. Figure 5.10 shows the dialog box used to configure this option. The *shuttered checkbox* is enabled if the *-stereo* command line option is used when invoking Smokeview and the video card supports shuttered stereo display.

The first method, denoted sequential stereo, works by displaying images for the left and right eye alternately in time. Shuttered glasses synchronized with the monitor are used to ensure that only the left eye sees the left image and only the right eye sees the right image. A monitor displaying this type of stereo should

[1]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.

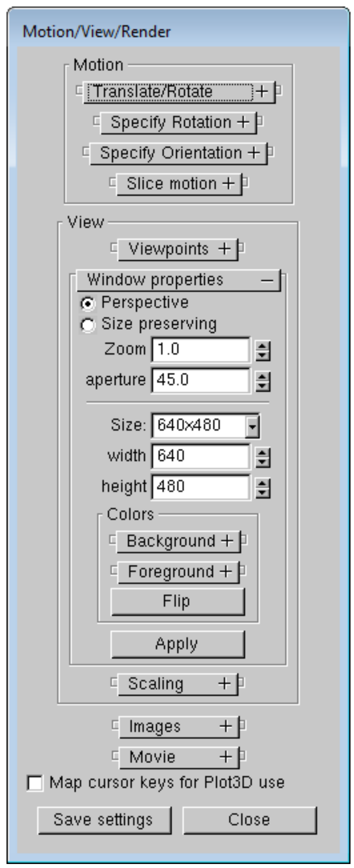


Figure 5.8: Dialog box for specifying window properties. The Windows portion of the Motion/View/Render dialog box allows one to set the window size, projection type (perspective or size preserving) and zoom level.

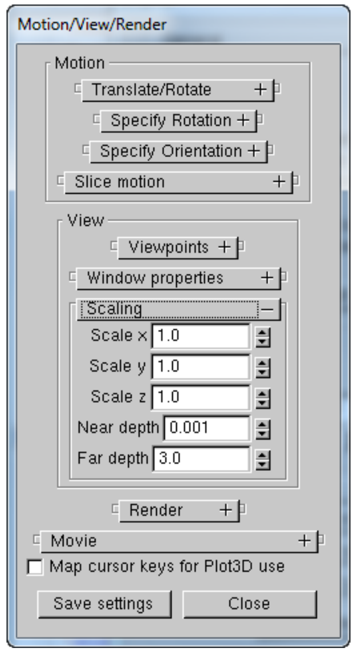


Figure 5.9: Dialog box for setting scaling and depth parameters. The Scaling/Depth portion of the Motion/View/Render dialog box allows one to specify scaling parameters for x, y and z scene dimensions and to specify the near and far depth planes.

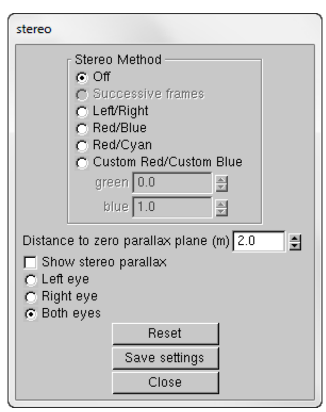


Figure 5.10: Dialog box for activating the stereo view option.

have a refresh rate of at least 120 frames per second (60 frames per second for each eye) otherwise flickering is noticeable. Unfortunately, most of today's LCD flat panel monitors typically do not have refresh rates faster than 60 to 80 frames per second. This method (for Smokeview) requires a video card that supports OpenGL *QUAD buffering*. This Smokeview stereo option may be enabled from the command line by using the `-stereo` option.

The second method, denoted left/right stereo, displays the two images side by side. With practice, one can merge both images without requiring specialized glasses (though they are available if desired) especially if the images are small and not separated by a large angle. A trick for seeing the stereo effect is to place a finger from each hand in the center of each picture. Then relax your eyes while trying to *merge* your two fingers together. Figure 5.11 show an example of the left/right method for generating a stereo image. This method can generate full colored images and requires no equipment (for most people) to view but results in smaller images.

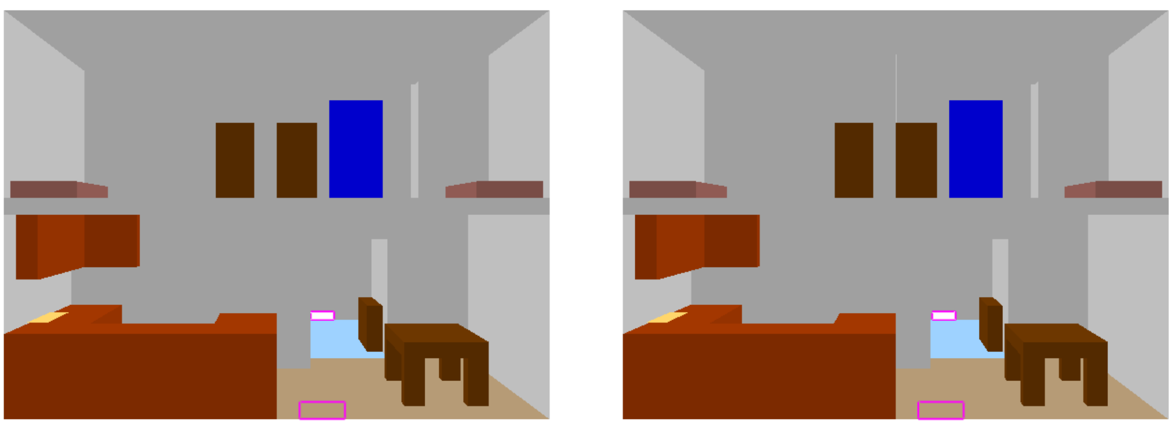


Figure 5.11: Stereo pair view of a townhouse kitchen fire. To aid in viewing the stereo effect, place a finger in front of each image. Relax your eyes allowing your two fingers and stereo pair images to merge into one.

The third method, uses color to separate left and right images. One method denoted red/blue stereo, displays red and blue versions of each image. Glasses with a red left lens and a blue right lens are required to view the image. As with the shuttered glasses for sequential stereo, the colored glasses *separate* the images enabling each eye to see only one image. Red/blue colored glasses may be obtained inexpensively. They also may be made using red and blue cellophane or by coloring clear plastic with read and blue marking pens. Figure 5.12 uses the red/blue method for generating a stereo image. This method generates full size images, requires only inexpensive glasses to view but can only display monochrome images. The red/cyan method for displaying stereo images works similarly to the red/blue method. The main difference is that since cyan is the made up of green and blue (the *opposite* in some sense of red), the combination of red and cyan lenses allow all colors to pass to your eyes.

Figures 5.13 uses the red/cyan method for generating a stereo image. As with red/blue, this method generates full size images. This method allows Smokeview scenes to be displayed in full color.

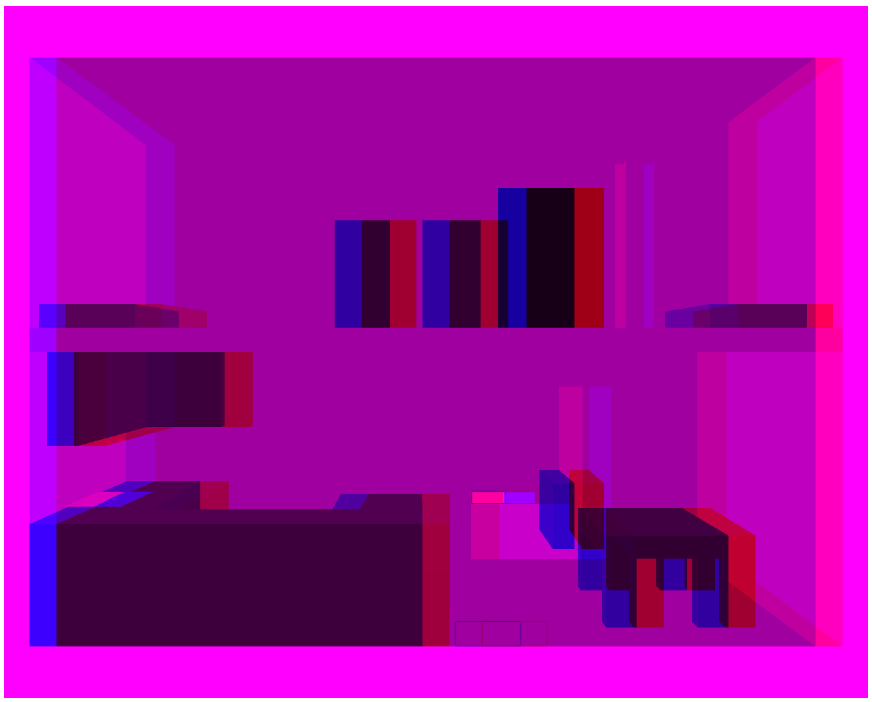


Figure 5.12: Red/blue stereo pair view of a townhouse kitchen fire. Red/blue glasses are required to see the 3D stereo effect.

5.5 Rendering Scenes - Creating Image Files

The *Render* portion of the Motion/View dialog box, as illustrated in Fig. 5.14, is used to convert a *Smokeview scene* to one or more image files. Image file formats may be either PNG or JPEG. One frame is rendered for each time step in a time dependent files unless a skipping parameter is specified. A skipping interval may be selected to generate fewer images using the *Where frames* control. One frame is rendered for static files. Higher resolution images may be generated by selecting a multiplier factor using the *Resolution multiplier* control. For example, a factor of 3 generates an image with 3 times the original scene resolution. Unwanted portions of a scene may be removed or clipped before it is rendered by specifying a clipping region. A clipping region is specified in terms of left, right, bottom and top pixel locations.

5.6 Setting Viewpoints

Controls in the *Viewpoint* portion of the Motion/View dialog box, as illustrated in Fig. 5.15, allow one to save the position and orientation of the scene. These saved position/orientations are called viewpoints. Viewpoints may then be selected resulting in the scene returning to a previously saved position and orientation.

To define a viewpoint, manipulate the scene to the desired position and orientation. Then press *Add* button. This adds the new viewpoint to a list of available viewpoints. The *Replace* button replaces the currently active viewpoint (as named by the *Select* list item) with the current position and orientation.

To change the view to a currently saved viewpoint, use the *Select* listbox to select the desired viewpoint. The *Delete* button, as one would expect, removes the viewpoint. The *Edit* text box is used to change the name of the currently selected viewpoint. The *view at startup* button is used to specify the viewpoint that should be active when Smokeview starts up.

5.7 Clipping Scenes

It is difficult to view the interior of a scene when modeling complicated geometries. To alleviate this problem, one may change the blockage view to *outline* with the Show/Hide>Blockages menu or one may clip the scene. Portions of the scene may be hidden or clipped by setting up to six clipping planes. The scene is then drawn on one side of a clipping plane but not the other. In general, a clipping plane may have any orientation. Smokeview defines six clipping planes, two clipping planes for each of the three coordinate axes. The two x axis clipping planes clip regions with x coordinates smaller than 'x_{min}' (in FDS coordinates) and larger than 'x_{max}' . Clipping planes for the y and z axis behave similarly. Clipping plane values are specified using the Clipping dialog box which is opened by selecting the *Dialogs>Clip Geometry* menu item. Figure 5.16 shows this dialog box with the y_{max} plane active. Figure 5.17 shows three versions of a scene. Figure 5.17a is drawn with no clipping. Figure 5.17b is drawn clipping just the geometry (blockages). Figure 5.17c is drawn clipping both the geometry and the data.

The clipping dialog box also allows one to hide blockages. Blockages for any given mesh may be hidden by selecting the appropriate checkbox in the *Hide blockages* rollout panel.

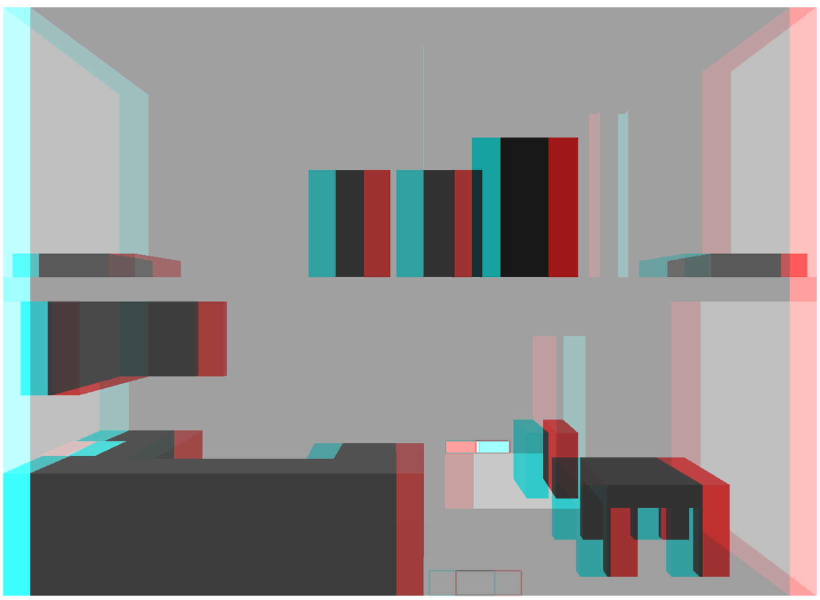


Figure 5.13: Red/cyan stereo pair view of a townhouse kitchen fire. Red/cyan glasses are required to see the 3D stereo effect.

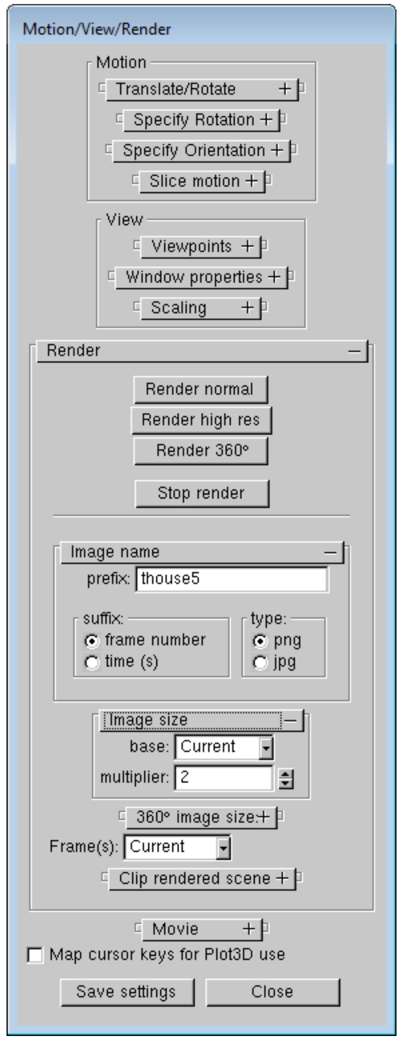


Figure 5.14: Dialog box for creating images of the smokeview scene. The Render portion of the Motion/View/Render dialog box allows one to create images of the smokeview scene. One may also clip or crop the rendered image.

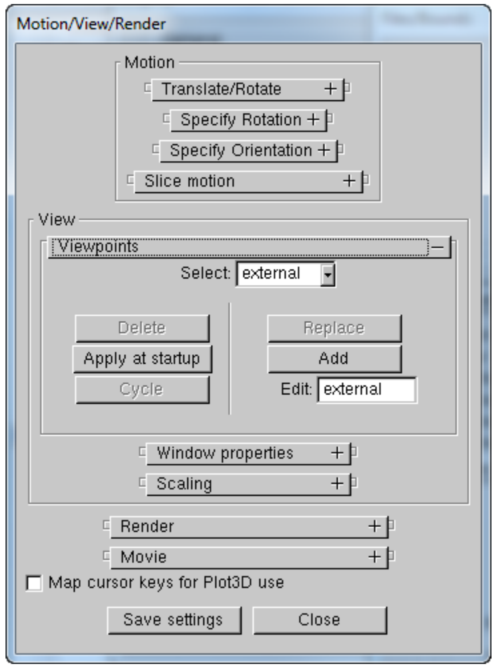


Figure 5.15: Dialog box for specifying scene viewpoints. The viewpoint portion of the Motion/View/Render dialog box allows one to define and control viewpoints.

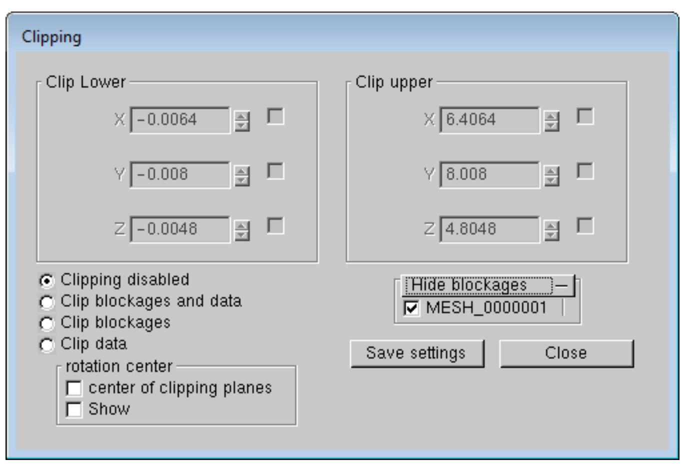
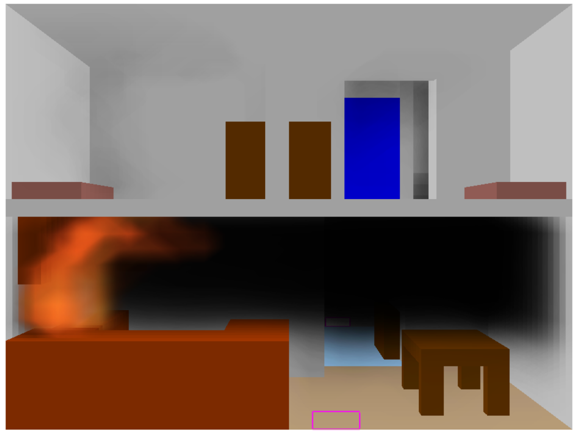
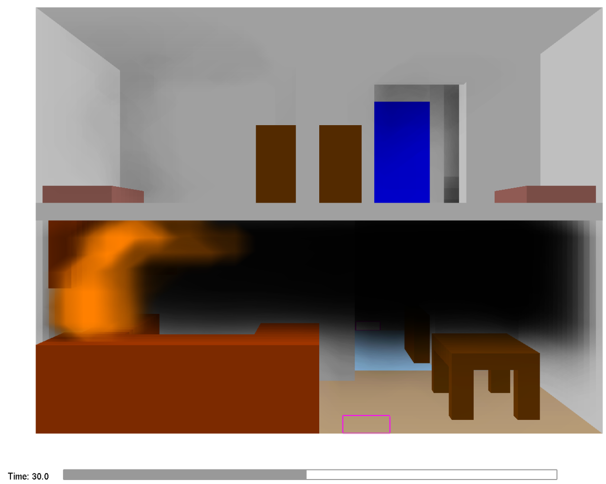


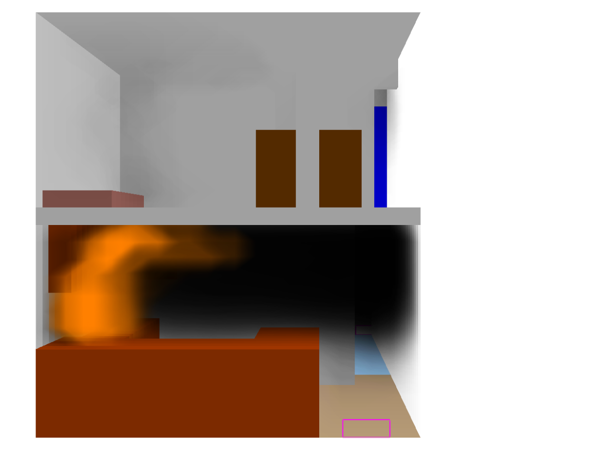
Figure 5.16: Clipping dialog box. Minimum and maximum clip plane values are set for X, Y and Z planes. When clipping, one may clip data, geometry or both.



a) no clipping



b) clip blockages



c) clip blockages and data

Figure 5.17: Three views of a scene. The first view is drawn without clipping, the second view shows the scene clipping only the geometry (blockages), the third view shows the scene clipping both the geometry and the data.

Chapter 6

Creating Custom Objects

Smokeview visualizes FDS devices such as heat and smoke detectors using instructions found in a file named `objects.svo`. Smokeview also uses these instructions to represent people (avatars) in FDS-EVAC simulations and to represent trees and shrubs in FDS WUI simulations. The Smokeview implementation of FDS devices is referred to as objects in this chapter.

The instruction file is located in the Smokeview installation directory[1]. The instructions correspond to OpenGL library calls, the same type of calls Smokeview uses to visualize FDS cases. Smokeview then acts as an interpreter executing OpenGL commands as specified in the object definition file. Efficiency is attained by compiling these instructions into display lists, terminology for an OpenGL construct for storing and efficiently drawing collections of OpenGL commands. New objects may be designed and drawn without requiring modifications to Smokeview and more importantly may be created by someone other than the Smokeview developer.

An object's appearance may be fixed or it may be altered based upon data specified in an FDS input file. The `sensor` object is drawn as a small green sphere with a fixed diameter. Its appearance is the same regardless of how an FDS input file is set up. The appearance of the `tsphere` object (t for texture) depends on data specified in the FDS input file. One may specify the diameter of the sphere and an image to cover it with (the image is known as a texture map).

As with preference or `.ini` files, Smokeview looks for object definition files in three locations: in a file named `objects.svo` located in the Smokeview installation directory, in a file named `objects.svo` located in the casename directory and in a file named `casename.svo` also located in the casename directory where `casename` is the name of the case being visualized.

This chapter describes how to create new objects. Though all of the examples are given for drawing FDS devices, the intent of this procedure is to be more general allowing Smokeview to draw other types of objects such as people walking.

6.1 Object File Format

The first statement in an object definition is the keyword `OBJECTDEF` (or `AVATARDEF` when defining a *person*). The next statement is the name or label for the object. Following this are the instructions used for creating the object. Each instruction consist of zero or more data values followed by a command. Comments may be placed anywhere in the object definition file by adding text after a double slash '//'.

Data from FDS may be optionally passed to the object definition by placing a series of labels, written as `:var1 ... :varn`, at the beginning of the definition. These data values may then be accessed later

[1] The current objects.svo file containing documentation and a listing of object definitions is listed in Appendix D.6

in the definition using `$var1 ... $varn` respectively. The data placed in `:vari` labels is specified in the FDS input file using the `SMOKEVIEW_PARAMETERS` keyword on the `&PROP` input line.

There are two types of instructions for drawing basic geometric objects. Instructions for drawing objects such as cubes, disks, spheres etc. and instructions for manipulating these objects through transformations such as scaling, rotation and translation. Collectively these instructions specify the type, location and orientation of objects used to represent objects. The important feature of this process is that new objects may be designed and drawn without the need to modify Smokeview.

Some examples of argument/instruction pairs are `d drawsphere` for drawing a sphere of diameter `d` or `x y z translate` for translating an object by (x,y,z). The symbols `d`, `x`, `y` and `z` are specified in the object file using a numerical constant such as 1.23 or using a reference such as $var to data located elsewhere.

Transformation commands are cumulative, each command builds on the effects of the previous one. The commands `push` and `pop` isolate these effects by saving and restoring the geometric state.

The format for an object definition file is given in more detail in Fig. 6.1. Each object definition consists of one or more frames. A frame is used to represent various states of the object. Objects such as thermocouples which do not activate use just one frame. Other objects such as sprinklers or smoke detectors which do activate use two frames, the first for normal conditions and the second for when the object has activated.

Figure 6.2 illustrates a simple example of an object definition used to draw a sensor along with the corresponding Smokeview view. The definition uses just one frame. A sphere is drawn with color yellow and diameter 0.038 m. Push and pop commands are not necessary because there is only one object and no transformations are used.

The example illustrated in Fig. 6.3 is more complicated. It shows a definition of a heat detector along with a corresponding Smokeview view. The definition uses two frames. The first frame represents the heat detector's inactive state, the second frame represents the active state (commands after the `NEWFRAME` keyword). This definition uses disks, a truncated cone and spheres. The scale and translate commands are used to draw these objects at the proper size. The translate command then positions them properly. Two frames are defined for both the inactive and active (after the heat detector has activated.) states.

Figure 6.4 shows an example of a definition used to draw a scaled sphere using scalings obtained from an FDS input file along with the corresponding Smokeview view. This definition is set up so that if the label value 'D' has a value greater than 0.0 then a sphere is drawn with diameter D otherwise an ellipsoid is drawn with dimensions 'DX', 'DY' and 'DZ'. This definition uses just one frame. The scaled sphere/ellipsoid is drawn using data specified on the `SMOKEVIEW_PARAMETERS` keyword in the FDS input file.

Figure 6.5 gives Smokeview views for several objects defined in the `objects.svo` file. A more complete list is found in the FDS User's Guide [4].

6.2 Elementary Geometric Objects

The objects described in this section are the building blocks used to construct more complex objects. Each command used to draw an elementary geometric object consists of one or more arguments followed by the command, for example, the command sequence `0.3 drawsphere` draws a sphere with diameter 0.3 (all units as with FDS are in meters).

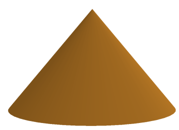
```
// ************ object file format ********************

//  1. comments and blank lines may be placed anywhere
//  2. any line not beginning with "//" is part of the definition.
//  3. the first non-comment line after OBJECTDEF is the object name
//  4. an object definition may contain, labels, numerical constants
//     (a number), string constants (enclosed in " ") and/or
//     commands (beginning with a-z)
//  5. a label begins with ':' as in :dx
//  6. the label :dx may be accessed afterward using $dx
//  7. An object may contain multiple frames or states.  A new frame within
//     an object is defined using NEWFRAME

// OBJECTDEF // OBJECTDEF begins the object definition

//   object_name // name or label for object
//   :var1 ... :varn  // a series of labels may be specified for use by
//                    // the object definition.  Data is copied to these
//                    // label locations using the SMOKEVIEW_PARAMETERS
//                    // &PROP keyword or from a particle file. The data
//                    // in :varn may be referenced  elsewhere in the
//                    // definition using $varn

//   // A series of argument/command pairs are specified on one or
//   // more lines.

//   arg1 ... argn command1 arg1 ... argn command2 ...

//   // An argument may be a numerical constant (e.g., 2.37), a string
//   // (e.g., "SKYBLUE"), a label (e.g., :var1),  or a reference to a
//   // label located elsewhere (e.g., $var1)

//  NEWFRAME    // beginning of next frame
//   more argument/command pairs for the next object frame
//   ....
```

Figure 6.1: Object file format.

drawarcdisk The command, `a d h drawarcdisk`, draws a portion of circular disk with angle a, diameter d and height h. The origin is located at the center of the disk's base.

`60.0 0.25 0.50 drawarcdisk`

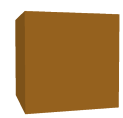
drawcircle The command, `d drawcircle` , draws a circle with diameter d. The origin is located at the center of the circle.



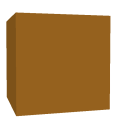
`0.50 0.30 drawcone`

drawcone The command, `d h drawcone`, draws a right circular cone where d is the diameter of the base and h is the height. The origin is located at the center of the cone's base.

drawcube The command, `s drawcube`, draws a cube where s is the length of the side. The origin is located at the center of the cube. An oblong box, a box with different length sides, may be drawn by using `scale` along with `drawcube`. For example, `1.0 2.0 4.0 scale 1.0 drawcube` creates a box with dimensions $1 \times 2 \times 4$.



`0.25 drawcube`



`0.25 drawcubec`

drawcubec The command, `s drawcubec`, is the same as `s drawcube` except that the origin is located at the front, left, bottom corner of the cube rather than at the cube center. An oblong box, a box with different length sides, may be drawn by using `scale` along with `drawcubec`. For example, `1.0 2.0 4.0 scale 1.0 drawcube` creates a box with dimensions $1 \times 2 \times 4$.

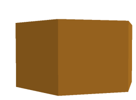
drawdisk The command, `d h drawdisk`, draws a circular disk with diameter d and height h. The origin is located at the center of the disk's base.

`0.25 0.50 drawdisk`

`0.25 0.50 drawcdisk`

drawcdisk The command, `d h drawcdisk`, draws a circular disk with diameter d and height h. The origin is located at the center of the disk. This command is a shortcut for `h 2.0 :hd2 div $hd2 offsetz d h drawdisk`.

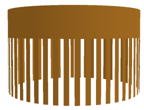
drawhexdisk The command, `d h drawhexdisk`, draws a hexagonal disk with diameter d and height h. The origin is located at the center of the hexagon's base.



`0.5 0.25 drawhexdisk`

drawline The command, `x1 y1 z1 x2 y2 z2 drawline`, draws a line between the points (x_1, y_1, z_1) and (x_2, y_2, z_2).

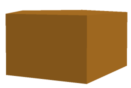
`0.5 0.1 0.2 1 drawnotchplate`



`0.5 0.1 0.2 -1 drawnotchplate`

drawnotchplate The command, `d h nh dir drawnotchplate`, draws a notched plate. This object is used to represent a portion of a sprinkler where d is the plate diameter, h is the plate height (not including notches), nh is the height of the notches and dir indicates the notch orientation (1 for vertical, -1 for horizontal). The origin is located at the center of the plate's base.

drawpoint The command, `drawpoint`, draws a point (small square). The command, `s setpointsize` may be used to change the size of the point. The default size is 1.0 .

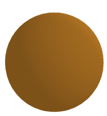


`5 0.35 0.15 drawpolydisk`

drawpolydisk The command, `n d h drawpolydisk`, draws an n-sided polygonal disk with diameter d and height h. The origin is located at the center of the polygonal disk's base. The example to the left is a pentagonal disk.

drawring The command, `di do h drawring`, draws a ring where `di` and `do` are the inner and outer ring diameters and h is the height of the ring. The origin is located at the center of the ring's base.

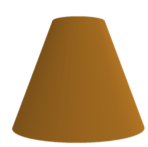
`0.3 0.5 0.1 drawring`



`0.25 drawsphere`

drawsphere The command, `d drawsphere`, draws a sphere with diameter d. The origin is located at the center of the sphere. As with an oblong box, an ellipsoid may be drawn by using `scale` along with `drawsphere`. For example, `1.0 2.0 4.0 scale 1.0 drawsphere` creates an ellipsoid with semi-major axes of length 1, 2 and 4. This is how the ball at the bottom of the heat detector in Fig. 6.3a is drawn.

drawtrunccone The command, `d1 d2 h drawtrunccone`, draws a right circular truncated cone where d1 is the diameter of the base, d2 is the diameter of the truncated portion of the cone and h is the height or distance between the lower and upper portions of the truncated cone. The origin is located at the center of the truncated cone's base.



`0.5 0.2 0.4 drawtrunccone`

6.3 Visual Transformations

As with geometric commands, transformation commands consist of zero or more arguments followed by the command. Transformation commands are used to directly or indirectly change how drawn objects appear. Visual transformations make changes directly, changing the location and orientation of drawn objects, setting drawing attributes such as point size, line width or object color or saving and restoring the geometric state. Arithmetic transformations, described in the next section, make changes indirectly by operating on data which in turn is used as inputs to various drawing commands.

Visual transformation commands map directly to counterparts in OpenGL. The rotate and translate commands change the origin (translate) or orientation of the x,y,z axes (rotate). The offsetx, offsety and offsetz commands translate objects along just one axis. The PUSH command is then used to save the origin or axis orientation while the POP command is used to restore the origin and axis orientation.

gettextureindex The command

```
"texture_file" :texture_index gettextureindex
```

finds the index in an internal Smokeview table containing the entry texture_file (a file containing a texture map image). This index is used by other object drawing routines that support texture mapping (presently drawtsphere).

gtranslate The command, `x y z gtranslate`, translates objects in a global reference frame, the same reference frame used to define FDS geometry. Objects drawn after the gtranslate command are moved by x, y and z along the x, y and z Cartesian axes respectively. Equivalently, one can think of think of `x y z gtranslate` as translating the origin by (-x,-y,-z).

offsetx The command, `x offsetx`, translates objects drawn afterwards by x along the x axis.

offsety The command, `y offsety`, translates objects drawn afterwards by y along the y axis.

offsetz The command, `z offsetz`, translates objects drawn afterwards by z along the z axis.

orienx, orieny, orienz The command, `x y z orienx`, rotates the scene so that the vector $u = (1,0,0)$ in the original scene maps to $w = (x,y,z)$. The commands `orieny` and `orienz` are similar to `orienx` mapping $(0,1,0)$ and $(0,0,1)$ to (x,y,z) instead.

pop The command, `pop`, restores the origin and axis orientation saved using a previous `push` command. The total number of `pop` and `push` commands must be equal, otherwise a fatal error will occur. Smokeview detects this problem and draws a red sphere instead of the incorrectly defined object.

push The command, `push`, saves the origin and axis orientation. (see above comment about number of `push` and `pop` commands).

randxy, randxz, randyz The command `flag randxy` performs a random rotation about the z axis (in the xy plane) if `flag` is 1. If `flag` is anything else this command has no effect. The commands `randxz` and `randyz` are similar, rotating within the xz and yz planes instead of the xy plane.

rotateaxis The command, `angle x y z rotateaxis`, rotates objects drawn afterwards by `angle` degrees about an axis defined by the vector (x,y,z).

rotatexyz The command, `x y z rotatexyz`, rotates objects from the vector $(0,0,1)$ to the vector (x,y,z) . The axis of rotation computed internally by Smokeview is $(0,0,1) \times (x,y,z) =$

$(-y,x,0)$ (a vector perpendicular to the plane formed by vectors $(0,0,1)$ and (x,y,z)) . The cosine of the angle of rotation is $z/\sqrt{x^2+y^2+z^2}$

rotatex The command, `r rotatex`, rotates objects drawn afterwards `r` degrees about the x axis.

rotatey The command, `r rotatey`, rotates objects drawn afterwards `r` degrees about the y axis.

rotatez The command, `r rotatez`, rotates objects drawn afterwards `r` degrees about the z axis. A cone or any object for that matter may be drawn upside down by adding a `rotatez` command as in `180 rotatez 1.0 0.5 drawcone`.

scalexyz The command, `x y z scalexyz`, stretches objects drawn afterwards by x, y and z respectively along the x, y and z axes. The `scalexyz` along with the `drawsphere` commands would be used to draw an ellipsoid by stretching a sphere along one of the axes.

scale The command, `xyz scale`, stretches objects drawn afterwards xyz along each of the x, y and z axes (equivalent to `xyz xyz xyz scalexyz`).

setbw The command, `gray setbw`, sets the red, green and blue components of color to gray (equivalent to gray gray gray setcolor). As with the `setcolor` command, `setbw` is only required when the gray level changes, not for each object drawn.

setcolor The command, `"color name" setcolor`, obtains sets the color to the red, green and blue components of the FDS standard color `color name`.

setlinewidth The command, `w setlinewidth` sets the width of lines drawn with the `drawline` and `drawcircle` commands.

setpointsize The command, `s setpointsize`, sets the size of points drawn with the `drawpoint` command.

setrgb The command, `r g b setrgb`, sets the red, green and blue components of the current color. Any objects drawn afterwards will be drawn with this color. This command is not required for each object part drawn. The color component values range from 0 to 255.

translate The command, `x y z translate`, translates objects drawn afterwards by x, y and z along x, y and z axes respectively relative to the current (local) reference frame.

6.4 Arithmetic Transformations

Arithmetic transformation commands allow one to indirectly change how objects are drawn using information passed from FDS. This information is passed using the `SMOKEVIEW_PARAMETERS` keyword on the `&PROP` namelist statement. These commands transform data to change the inputs of subsequent object commands.

add The command,

`a b :val add`,

is used to compute the value, $val = a + b$, where a and b are either numerical constants or references to previously defined data. The result, *val* is placed in the label `:val` accessible later in the definition file using $val.

clip The command,

`val_in` `val_min` `val_max` :`val_clipped` `clip`,

is used to clip a value `val_in` between `val_min` and `val_max` using

$$\text{val\_clipped} = \max(\text{val\_min}, \min(\text{val\_in}, \text{val\_max})) \quad (6.1)$$

The inputs, `val_in`, `val_min` and `val_max` are either numerical constants or references to previously defined data. The clipped result is placed in the label :`val_clipped` accessible later in the definition file using $`val_clipped`.

div The command,

`a b :val div`,

is used to compute the value, $val = a/b$, where a and b are either numerical constants or references to previously defined data. If the denominator, `b`, is zero then the result, *val*, returned is zero and is placed in the label `:val` accessible later in the definition file using $val.

eq The command,

`a b eq`,

is used to copy data from the label b to a, *ie* performs the operation a=b.

gett The command,

`:time gett`,

is used to obtain the current simulation time. The simulation time is placed in the label `:time` accessible later in the definition file using $time.

mirrorclip The command,

`val_in` `val_min` `val_max` :`val_clipped` mirrorclip,

is used to clip a value `val_in` between `val_min` and `val_max` using

$$\text{val\_1} = \text{mod}(\text{val\_in} - \text{val\_min}, 2(\text{val\_max} - \text{val\_min})) \quad (6.2)$$

$$\text{val\_clipped} = \begin{cases} \text{val\_min} + \text{val\_1} & \text{val\_1} \leq \text{val\_max} - \text{val\_min} \\ \text{val\_max} - \text{val\_1} & \text{val\_1} > \text{val\_max} - \text{val\_min} \end{cases} \quad (6.3)$$

mult The command,

`a b :val mult`,

is used to compute the value, $val = ab$, where a and b are either numerical constants or references to previously defined data. The result, *val* is placed in the label `:val` accessible later in the definition file using $val.
The inputs, `val_in`, `val_min` and `val_max` are either numerical constants or references to previously defined data. The clipped result is placed in the label :`val_clipped` accessible later in the definition file using $`val_clipped`.

multiaddt The command,

`a b :val multiaddt`,

is used to compute the value, $val = at + b$, where t is the simulation time and a and b are either numerical constants or references to previously defined data. The result, *val* is placed in the label `:val` accessible later in the definition file using $val. This allows one to change how an object appears as a function of time (changing its size, rotating it, changing its color, etc.).

The command, `a b :val multiaddt`, is a shortcut for

```
:time gett a $time :at mult $at b :val add
```

periodicclip The command,

`val_in val_min val_max :val_clipped periodicclip`,

is used to clip a value `val_in` between `val_min`and `val_max` using

$$\text{val\_clipped} = \text{val\_min} + \text{mod}(\text{val\_in} - \text{val\_min}, \text{val\_max} - \text{val\_min}) \quad (6.4)$$

(6.5)

The inputs, `val_in`, `val_min` and `val_max` are either numerical constants or references to previously defined data. The clipped result is placed in the label `:val_clipped` accessible later in the definition file using `$val_clipped`.

sub The command,

`a b :val sub`,

is used to compute the value, $val = a - b$, where a and b are either numerical constants or references to previously defined data. The result, *val* is placed in the label `:val` accessible later in the definition file using $val.

6.5 Logical and Conditional Operators

Logical and conditional operators are used in conjunction to test values and execute portion of an object definition depending on the results of the test. Logical operators return 1 if the test is true and 0 if the test is false.

and The command

`a b :val AND`

returns 1 in :val if both a and b are true (any value other than 0), otherwise it returns 0.

gt The command

```
a b :val GT
```

returns 1 in :val if a is greater than b, otherwise it returns 0.

ge The command

```
a b :val GE
```

returns 1 in :val if a is greater than or equal to b, otherwise it returns 0.

if,else,endif Consider the object command sequence

```
$val IF
 arg1 arg2 command1 arg1 arg2 command2 ....
ELSE
 arg1 arg2 command3 arg1 arg2 command4 ....
ENDIF
```

The value `$val` is typically generated from a previous logical operation (*ie* with `GE, LT`, etc.). The commands between the `IF` and `ELSE` operators are executed if `$val` is not 0 otherwise the commands between `ELSE` and `ENDIF` are executed. The `ELSE` operator is optional.

lt The command

```
a b :val LT
```

returns 1 in :val if a is less than b, otherwise it returns 0.

le The command

```
a b :val LE
```

returns 1 in :val if a is less than or equal to b, otherwise it returns 0.

or The command

```
a b :val OR
```

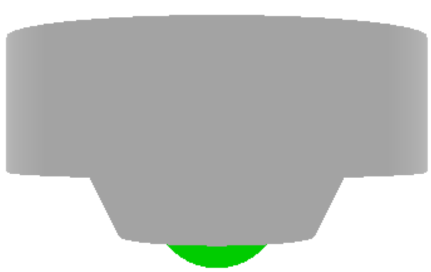
returns 1 in :val if either a or b are true (any value other than 0), otherwise it returns 0.

```
OBJECTDEF
 sensor
 1.0 1.0 0.0  setcolor
 0.038 drawsphere
```

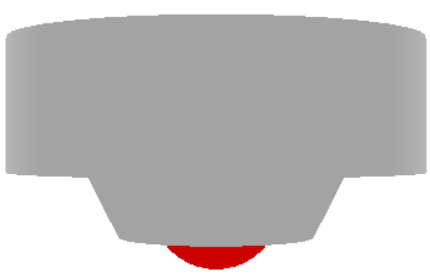
Figure 6.2: Instructions for drawing a sensor along with the corresponding Smokeview view.

Heat detector Instructions

```
OBJECTDEF
 heat_detector         // label, name of object

 // The heat detector has three parts
 //   a disk, a truncated disk and a sphere.
 //   The sphere changes color when activated.

 0.8 0.8 0.8 setcolor  // set color to off white
 push 0.0 0.0 -0.02 translate 0.127 0.04 drawdisk pop
 push 0.0 0.0 -0.04 translate 0.06 0.08 0.02 drawtrunccone pop
 0.0 1.0 0.0 setcolor
 push 0.0 0.0 -0.03 translate  0.04 drawsphere pop
 // push and pop are not necessary in the last line
 //   of a frame.  Its a good idea though, to prevent
 //   problems if parts are added later.
NEWFRAME  // beginning of activated definition
 0.8 0.8 0.8 setcolor
 push 0.0 0.0 -0.02 translate 0.127 0.04 drawdisk pop
 push 0.0 0.0 -0.04 translate 0.06 0.08 0.02 drawtrunccone pop
 1.0 0.0 0.0 setcolor
 push 0.0 0.0 -0.03 translate 0.04 drawsphere pop
```



inactive



active

Figure 6.3: Instructions for drawing an inactive and active heat detector along with the corresponding Smokeview view.

```
OBJECTDEF // object for a general ball
 ball
 :R=0 :G=0 :B=0 :DX :DY :DZ :D=-.1
 $D 0.0 :DGT0 GT
 $R $G $B setrgb
 $DGT0 IF
  $D drawsphere
  ELSE
  $DX $DY $DZ scalexyz 1.0 drawsphere
 ENDIF
 NO_OP
```

```
FDS input lines to create ball

The data labels (:R=0 :G=0 :B=0 :DX :DY :DZ :D=-.1) in the object file
correspond to the SMOKEVIEW_PARAMETERS inputs in the FDS input file
though the order may be different.

&PROP ID='ball' SMOKEVIEW_PARAMETERS(1:5)='R=0','G=0','B=255',
                       'DX=0.25','DY=.5','DZ='1.0' SMOKEVIEW_ID='ball' /
&DEVC XYZ=0.5,0.8,2.5, QUANTITY='TEMPERATURE' PROP_ID='ball' /
```

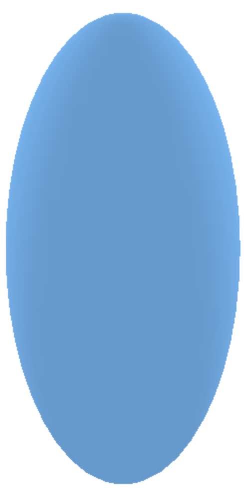


Figure 6.4: Instructions for drawing the dynamic object, ball, along with the corresponding FDS input lines and the Smokeview view.

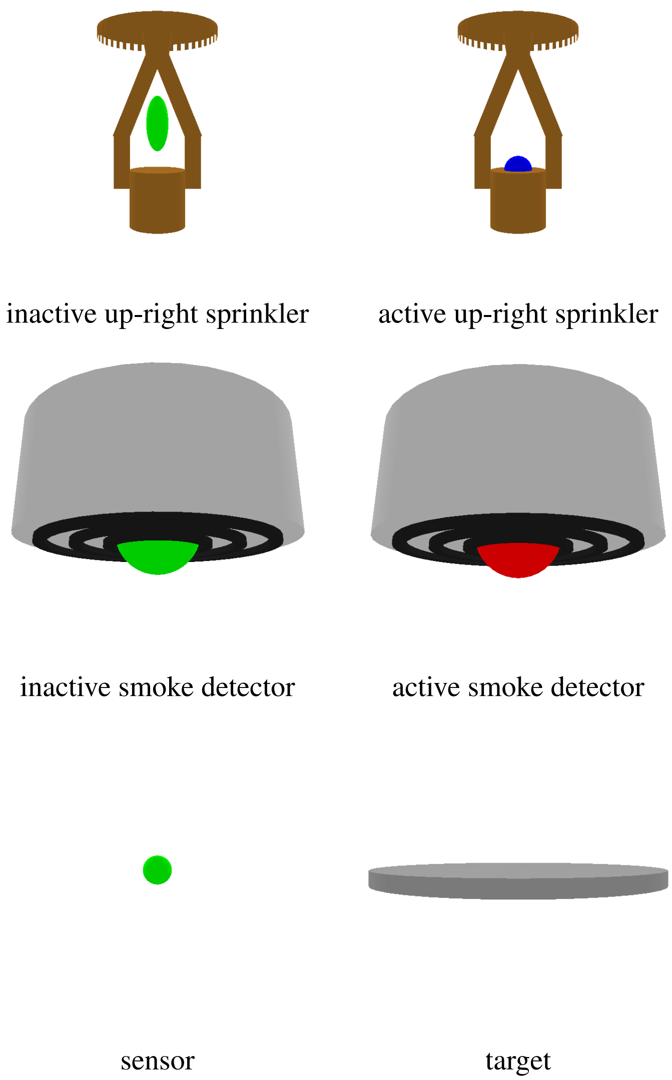


Figure 6.5: Smokeview view of several objects defined in the objects.svo file.

Chapter 7

Manipulating the Scene Automatically - The Touring Option

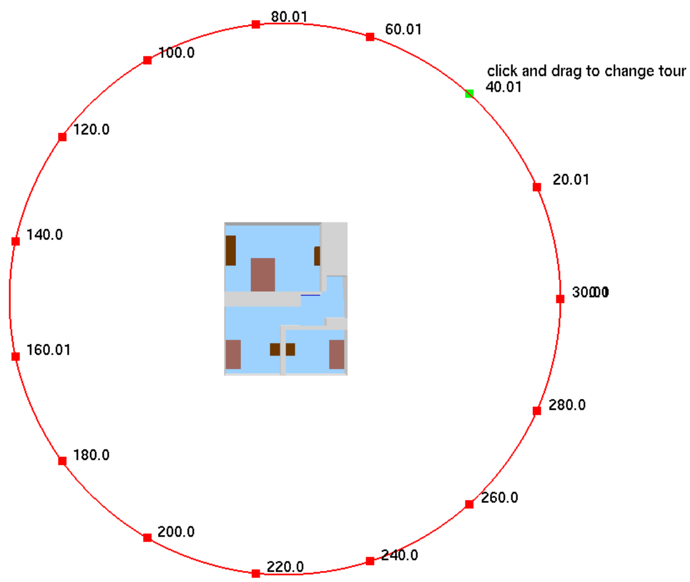


Figure 7.1: Overhead view of townhouse example showing the default *Circle* tour. The square dots indicate the key frame locations. Key frames may be edited using the Touring dialog box or by clicking and dragging with the mouse.

The touring option allows one to specify paths or tours through a Smokeview scene. One may then view a scene from the vantage point of an observer moving along one of these paths. Smokeview creates a tour surrounding the scene by default. This tour is given the name *Circular*. An example for the townhouse case is illustrated in Fig. 7.1. The user may create a new tour (or modify an existing one) using the Tour dialog box illustrated in Fig. 7.2. The user creates a tour by defining two or more key frames each with an associated view direction. The default view direction is towards the position (0.0,0.0,0.0) or lower front

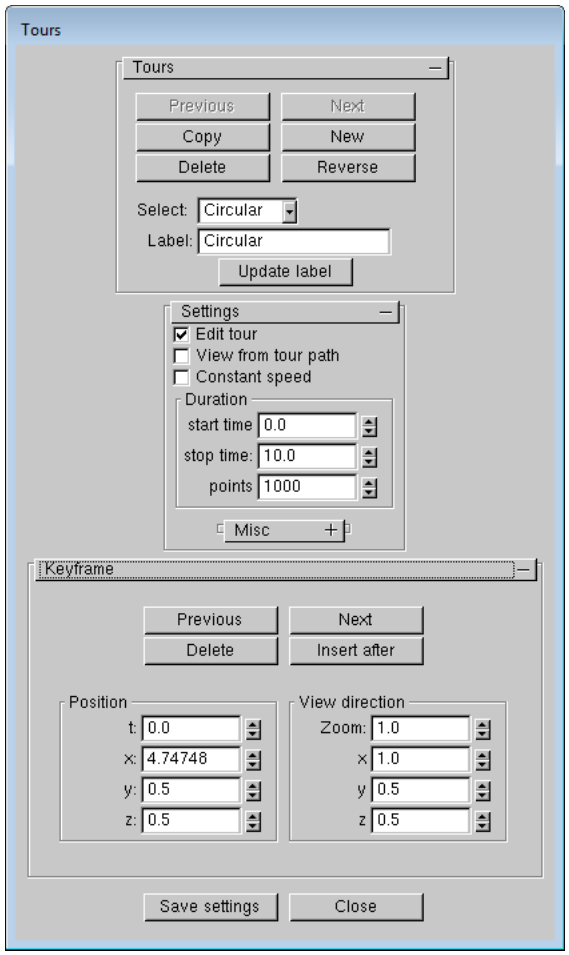


Figure 7.2: The Touring dialog box may be used to select tours or keyframes, change the position or view direction at each key frame and change the tension of the tour path.

left of the scene. Smokeview creates a smooth path through these positions using piecewise cubic Hermite polynomials.

7.1 Tour Settings

A new tour is created by clicking the `New Tour` button in the Edit Tour dialog box. The newly created tour has two key frames. The tour goes through the middle of the Smokeview scene starting at the front left and finishing at the back right. A tour may be modified by selecting the *Edit Tour* checkbox. Tour characteristics such as key frame positions and view directions at those positions are saved in the configuration file, `casename.ini`.

The *View From Tour Path* and *View From Selected Keyframe* checkboxes are used to control how one observes the scene using a Tour. If *View From Tour Path* is checked then one moves through the scene along the currently specified tour. If *View From Selected Keyframe* is checked then one sees the scene from the point of view of the currently selected key frame. This allows one to observe how the position and view direction settings will affect the view. Unchecking these options returns control of scene movement to the user allowing one to have a global view of the tour.

The speed traversed along a tour is determined from the number of points along the path and by the time value assigned at each key frame. If the *Constant Speed* checkbox is checked then the time at each key frame is determined so that the number of path points along the tour occur at a constant rate. The key frame times are determined so that the points between any two key frames are proportional to the path distance between these two key frames. The start and stop times of the tour may also be specified using the *start time* and *stop time* entries in the dialog box.

7.2 Key frame Settings

A tour is created from a series of key frames. One may specify the time, position and view direction at each key frame. Smokeview then obtains positions and view directions between key frames by interpolating using piecewise cubic Hermite cubic splines. A tour is created by pressing the `Add Tour` button. This tour has two key frames located at opposite ends of the Smokeview scene. Additional key frames may be created by selected the `Add` button. A tour may also be created by pressing the `Copy` button which makes a copy of the currently selected tour. A orientation of a tour may be reversed by selecting the `Reverse` button.

The position and viewpoint of a key frame may be adjusted. First it must be selected. A key frame may be selected by either clicking on it with the left mouse button or by *moving* through the key frames using the `Next` or `Previous` buttons. The active key frame as drawn within the Smokeview scene changes color from red to green. Key frame positions may then be modified by changing data in the t, X, Y or Z edit boxes or dragging the key frame rectangle with the mouse. A different view direction may also be set.

A key frame may also be moved with the mouse. Clicking on a key frame node then dragging the mouse left/right and up/down moves the key frame horizontally and vertically. Pressing the <CTRL> key while dragging the mouse restricts key frame movement to a horizontal direction (with respect to the mouse). Pressing the <ALT> key while dragging the mouse restricts key frame movement to a vertical direction (again with respect to the mouse).

A new key frame is created by clicking the `Add` button. It is formed by averaging the positions and view directions of the current and next key frames. If the selected key frame is the last one in the tour then a new key frame is added beyond the last key frame. A key frame may also be created by pressing the 'a' key.

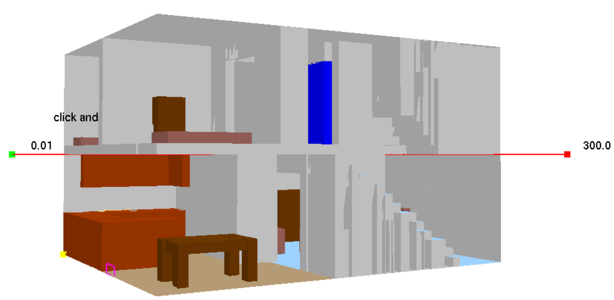
A key frame may be deleted by clicking the `Delete` button. The currently selected key frame may also be deleted by pressing the 'd' key. There is no `Delete Tour` button. A tour may be deleted by either deleting all of its key frames or by deleting its entry in the casename.ini file.

A view direction may be defined at each key frame by either setting direction angles relative to the path (an azimuthal and an elevation angle) or by setting a direction relative to the scene geometry (a Cartesian (X,Y,Z) view direction).

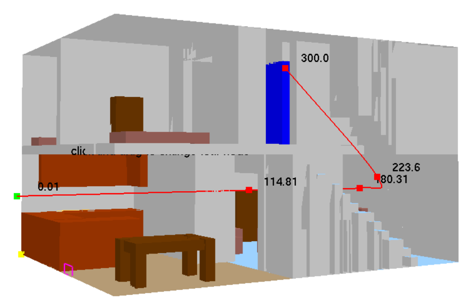
Path view directions are absolute. One selects the (x,y,z) view position by editing the x,y and/or z edit boxes in the View direction panel.

Cubic Hermite polynomials are uniquely specified for each interval using function and derivative values at each endpoint of the interval (*i.e., 4 data values*).

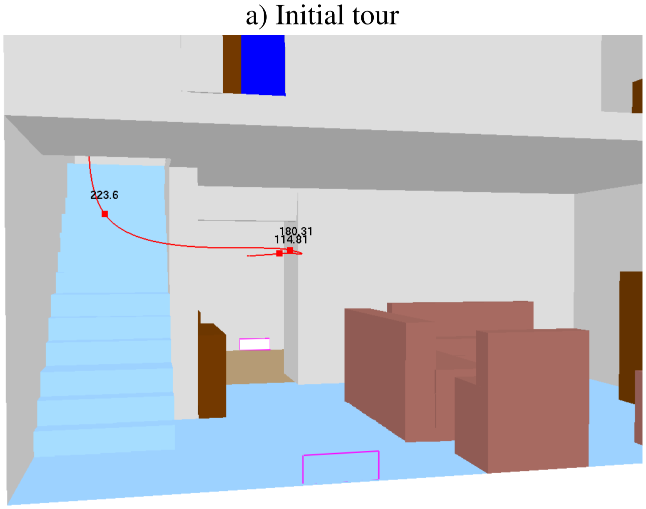
7.3 Setting up a tour



a) Initial tour



b) First and last step set with 5 key frames



c) All key frame positions set

Figure 7.3: Tutorial examples for Tour option.

The following steps give a simple example of setting up a tour in the townhouse scenario. The tour will begin at the back of the house, go towards the front door and then end at the top of the stairs. These steps are illustrated in Fig. 7.3.

1. Start by clicking the *Dialog>View>Create/edit tours...* menu item which opens up the Edit Tours dialog box.

2. Click on the `New Tour` button in the Edit Tour dialog box. This creates a tour, illustrated in Fig. 7.3a, starting at the front left of the scene and ending at the back right. This tour has two key frames. The elevation of each key frame is halfway between the bottom and top of the scene.

3. Click on the *Edit Tour Path* checkbox. This activates buttons that allows the user to edit the properties of each individual key frame. Click on the square dot at the back of the townhouse. This is the first key frame. Change the "Z" value to 1.0. Click on the second dot and change its "Z" value to 1.0.

4. Click on the `Add` button, found inside the *Edit Keyframe's Position* panel, three times. This will add three more key frames to the tour which will be needed so that the path bends up the stairs. You should now have five key frames.

5. Move the first key frame at the back of the townhouse near the double door by setting X, Y, Z positions to (3.8,-1.0,1.6). Move the last key frame to the top of the steps by setting X, Y, Z positions to (6.1,3.6,4.1). The path should now look like Figure 7.3b.

6. Move the second, third and fourth key frames to positions (4.0,4.0,1.6), (4.4,6.8,1.6) and (5.9,6.1,2.0). The path should now look like Fig. 7.3c.

7. Click on the `Save Settings` button to save the results of your editing changes.

8. To see the results of the tour, click on the *View From Tour Path* checkbox.

Chapter 8

Running Smokeview Automatically - The Scripting Option

8.1 Overview

Smokeview may be run in an automatic or batch mode using instructions found in a text file. The intent of the scripting option is to allow one to reproducibly document a case. A script may be re-run resulting in newly generated images guaranteed to correspond properly with the previously generated ones whenever changes occur in the FDS input file or in the FDS or Smokeview applications.

Script instructions direct Smokeview to perform actions such as loading data files, moving the scene to a specified view point, setting the time and rendering the scene. Smokeview settings such as font sizes, file bounds, label visibility, etc. are set by using the script command LOADINI to load a custom named .ini file. A simplified scripting language results by allowing most customizations to be performed through the use of .ini files.

8.2 Creating a Script

Scripts may be created by Smokeview using the script recorder feature or may be created by editing a text file using commands described in the glossary that follows. A script may be run using three methods. It may be run from within Smokeview using the *Load/Unload>Script Option* menu or from the Scripts panel of the File/Bounds dialog box illustrated in Fig. 8.1. It may also be run from a Windows command shell using the command

```
smokeview -runscript casename
```

where casename is the name specified by the `CHID` keyword defined in the FDS input data file.

The recorder is turned on using the *Load/Unload>Script Option* menu and selecting *Start Recording*. After performing a sequence of steps, it is turned off and the script is saved. Typically steps involve loading data files, setting view points, setting times and rendering images.

8.2.1 Example 1

This example describes the steps used to create a simple script. This script will load a slice file and then display and render it at 10 s, 20 s, 30 s and 40 s. The script corresponding to the steps listed below is given

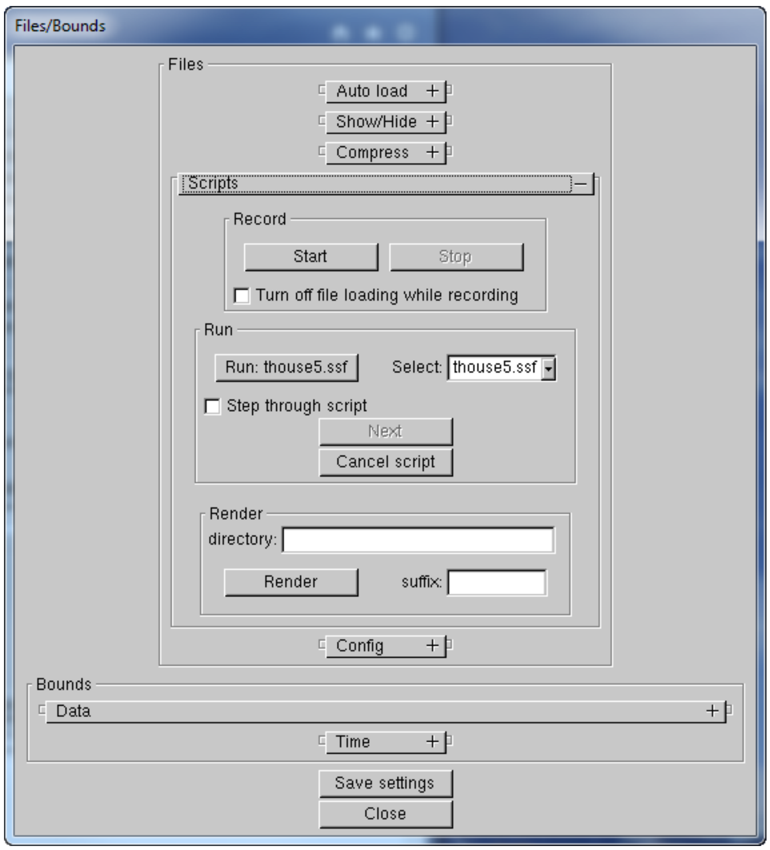


Figure 8.1: Script dialog box. The Script dialog box allows one to setup and run Smokeview scripts.

```
RENDERDIR
  ..\..\Manuals\SMV_User_Guide\SCRIPT_FIGURES
XSCENECLIP
 0 -0.001600 0 1.601600
YSCENECLIP
 0 -0.001600 0 1.601600
ZSCENECLIP
 0 -0.003200 0 3.203200
SCENECLIP
 0
// LOADFILE
//  script_slice_test_05.sf
LOADSLICEM
 TEMPERATURE
 2 0.800000
 1
SETTIMEVAL
 10.001744
RENDERCLIP
 1 212 212 23 48
RENDERONCE
 script_slice_test_10
SETTIMEVAL
 20.009329
RENDERCLIP
 1 212 212 23 48
RENDERONCE
 script_slice_test_20
SETTIMEVAL
 30.001192
RENDERCLIP
 1 212 212 23 48
RENDERONCE
 script_slice_test_30
SETTIMEVAL
 40.009205
RENDERCLIP
 1 212 212 23 48
RENDERONCE
 script_slice_test_40
```

Figure 8.2: Script commands generated using the Smokeview script recorder option.

in Fig. 8.2 and the resulting generated images are given in Fig. 8.3. .

Note that the keyword, `RENDERDIR`, may used to *direct* that rendered images be placed in any directory not just the current one. Also, the `RENDERONCE` keywords in this script have a blank line afterwards (put there by default by the Smokeview script recorder). In this case, Smokeview uses the default name for the resulting rendered image file. If this line is not blank, it is then used for the file name.

1. Obtain the test case `script_slice_test.fds` from the `Verification/Visualization` directory in the FDS−SMV repository.

2. Run the case with FDS

3. After opening the case in Smokeview, select the *Load/Unload>Script Options>Start Recording* menu item.

4. Load a slice file (doesn't matter which one).

5. Move the timebar to 10 s and then press the 'r' key. Repeat for 20 s, 30 s, and 40 s

6. Unload the slice file. (Not necessary, this step just makes the script action more obvious.)

7. Select the *Load/Unload>Script Options>Stop Recording* menu item. This is very important. The script will not be saved if you exit Smokeview without selecting this option.

8. Run the script using the *Load/Unload>Script Options* menu .

8.2.2 Example 2

This example describes the steps used to create a script that is more involved. It is listed in Fig. 8.4 which in turn was used to create the images illustrated in Fig. 8.5. The script built here will create three images, a slice file viewed and clipped from the left at 5 s, the same slice file viewed from the center at 10 s, and again the same slice file viewed and clipped from the right at 15 s. The center slice is not clipped.

Several preliminary steps need to be performed before script actions may be recorded. In particular a left and right view point will be defined and an .ini file will be setup that contains clipping values for the left and right slice file images.

Obtaining and setting up the example

1. Obtain the test case `script_test.fds` from the `Verification/Visualization` directory in the FDS−SMV repository. Copy this file to a separate directory if a local copy of the repository already exists Of course, these steps may be repeated for any test case that have data files defined.

2. Run the case with FDS

3. Open the case in Smokeview

4. Open the Scripts/Config panel of the File/Bounds dialog box, the Clip Geometry dialog box and the Viewpoints panel of the Motion/View/Render dialog box.

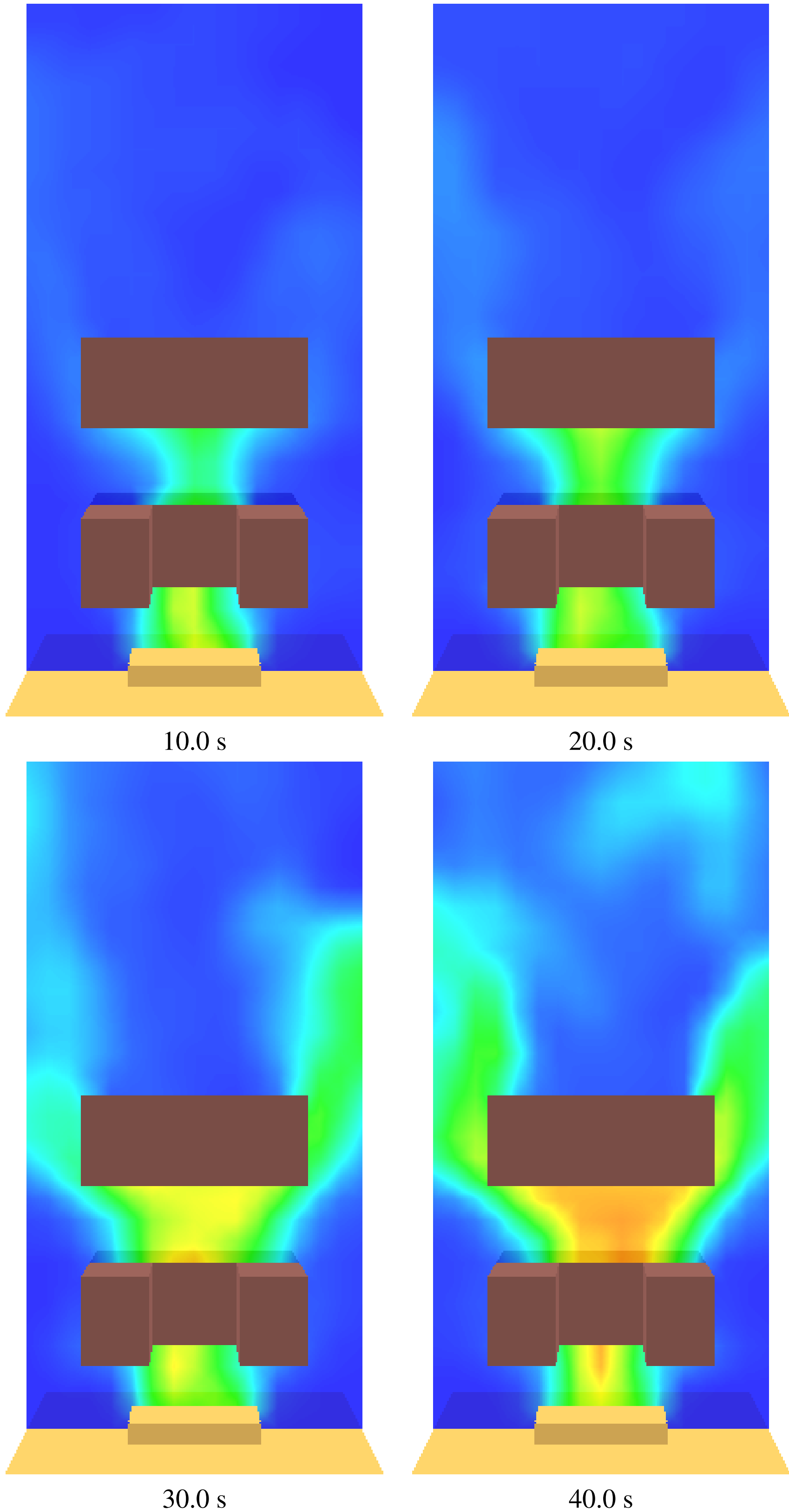


Figure 8.3: Smokeview images generated using script detailed in Fig. 8.2

Preliminary Steps - Setting up the viewpoints

1. Rotate the scene slightly to the right of center so that you can see the left side of the geometry. In the Viewpoints panel of the Motion/View/Render dialog change `new view` to `left` then click on the `Add` button.

2. Rotate the scene slightly to the left of center so that you can see the right side of the geometry. In the Viewpoints panel of the Motion/View/Render dialog box change `new view` to `right` then click on the `Add` button.

3. Click the `Save Settings` button.

   An .ini file has now been saved with two custom view points defined named left and right.

Preliminary Steps - Defining clip planes and creating additional .ini files

Defining the left clipping plane.

1. Click on the Clip Blockages + Data radio button,

2. change the *Clip Lower x* value 0.5 after checking the checkbox next to edit field.

3. Save an .ini file named script_test_left.ini by entering the text `left` in the suffix field of the Config files section of the Scripts/Config dialog box.

4. Click on the `Set` button then the `Save script_test_left.ini` button.

   Defining the right clipping plane.

1. Click on the Clip Blockages + Data radio button,

2. change *Clip Upper x* value 1.0 after checking checkbox next to edit field.

3. Save an .ini file named script_test_right.ini by entering the text `right` in the suffix field of the Config files section of the Scripts/Config dialog box.

4. Click on the `Set` button then the `Save script_test_right.ini` button.

Two .ini files named `scripts_test_left.ini` and `scripts_test_right.ini` have now been created.

Recording the Script

The script may be recorded now that the .ini files and viewpoints have been created. The following steps reference the Scripts/Config dialog box.

1. Click on the `Start Recording` button

2. Load the $y = 0.8$ temperature slice from the *Load/Unload* menu.

3. Generate the left image

   (a) Select the script_test_left.ini file and click on Load

   (b) Select the left view from the View menu.

(c) Set the time to 5.0
(d) Set the render suffix to left_05 and press the `Render` button

4. Generate the center image

   (a) Select the script_test.ini file and click on Load
   (b) Select external from the View menu.
   (c) Set the time to 10.0
   (d) Set the render suffix to right_10 and press the `Render` button

5. Render the right image

   (a) Select the script_test_right.ini file and click on the `Load` button
   (b) Select the right view from the View menu.
   (c) Set the time to 15.0
   (d) Set the render suffix to right_15 and press the `Render` button

6. Click on the `Stop Recording` button

```
// note: The RENDERDIR pathname has been changed to point
//       to where the Smokeview User guide script figures are kept
RENDERDIR
 ..\..\Manuals\SMV_User_Guide\SCRIPT_FIGURES
LOADINIFILE
 script_test.ini
SETVIEWPOINT
 left
// LOADFILE
//  script_test_05.sf
LOADSLICEM
 TEMPERATURE
 2 0.800000
 1
SETTIMEVAL
 5.012974
RENDERONCE
 script_test_left_05
SETVIEWPOINT
 center
SETTIMEVAL
 10.006555
RENDERONCE
 script_test_center_10
SETVIEWPOINT
 right
SETTIMEVAL
 15.006024
RENDERONCE
 script_test_right_15
```

Figure 8.4: Script commands generated using the Smokeview script recorder option.

8.3 Script Glossary

This section contains documentations for the script commands. Commands fall into three logical categories. Commands to load data files, commands to position the scene in both time and space and commands to output the scene to image files.

8.3.1 Loading and Unloading Files

LOADFILE Use LOADFILE to load a particular file. Smokeview will determine what kind of file it is (3d smoke, slice, etc.) and call the appropriate routine to load the data.

Use other LOAD commands to load files of the specified type for all meshes. Usage:

```
LOADFILE
  file (char)
```

LOADINIFILE Use LOADINIFILE to load a configuration of `.ini` file. Usage:

```
LOADINIFILE
  file (char)
```

LOADVFILE Use LOADVFILE to load a particular vector slice file. Smokeview will load the file specified along with the corresponding U, V and W velocity slice files if they are available. Usage:

```
LOADVFILE
  file (char)
```

LOADBOUNDARY Load a boundary file of a particular type. The type is the same as what Smokeview displays in the Load menus for boundary files. Usage:

```
LOADBOUNDARY
   type (char)
```

LOAD3DSMOKE Load a 3D smoke file. The types supported are SOOT DENSITY, HRRPUV, TEMPERATURE and CARBON DIOXIDE DENSITY. Usage:

```
LOAD3DSMOKE
  type (char)
```

LOADPARTICLES Load particle files. Only particle files created with FDS version 5 or later are supported. Usage:

```
LOADPARTICLES
```

PARTCLASSCOLOR Show a particular particle class. Class names supported for a given run are displayed in the Particle Class Smokeview menu. Usage:

```
PARTCLASSCOLOR
   color (char)
```

PARTCLASSTYPE Show a particular particle type. Type names supported for a given run are displayed in the Particle Type Smokeview menu. Usage:

```
PARTCLASSTYPE
   type (char)
```

LOADPLOT3D Load a Plot3D file for a given mesh at a specified time. Usage:

```
LOADPLOT3D
  mesh number (an integer from 1 to the number of meshes) time (float)
```

PLOT3DPROPS Specifies Plot3D plot properties that apply to all Plot3D plots currently being displayed. Usage:

```
PLOT3DPROPS
  variable_index (int) showvector (0/1) (int) vector_length_index
  (int) display_type (int) vector_length (float)
```

where

- variable_index - is an integer from 1 to the number of Plot3D file components (usually 5),
- showvector - is 1 to draw vectors, 0 otherwise
- vector_length_index, is an integer index from 0 to 6 pointing to an internal Smokeview array used to determine vector length.
- display_type - is 0 for stepped contours, 1 for line contours and 2 for continuous contours
- vector_length - if vector_length_index is negative then Smokeview uses vector_length set the PLOT3D vector length

SHOWPLOT3DDATA Specifies a particular Plot3D plot to be displayed (mesh number, whether visible or not, orientation and position) Usage:

```
SHOWPLOT3DDATA
  mesh number (int) plane orientation (int)  display show/hide (0/1) (int)
      position (float)
```

where

- mesh number - the mesh number (ranges from 1 to the number of meshes),
- orientation - direction or orientation of the plane being plotted, 1 for YZ planes, 2 for XZ planes and 3 for XY planes
- display - 0 if Plot3D plane is hidden, 1 if it is displayed
- position - position of Plot3D plane

LOADISO Load an iso-surface file for all meshes for a specified type. The type is the same as what Smokeview displays in the Load menus for iso-surface files. Usage:

```
LOADISO
  type (char)
```

LOADISOM Load an iso-surface file for a specified mesh and type. The type is the same as what Smokeview displays in the Load menus for iso-surface files. Usage:

```
LOADISO
  type (char)
  mesh number (int)
```

LOADSLICE Load a slice file of a given type. The type is the same as what Smokeview displays in the Load menus for slice files. The plane orientation is specified by using 1 for x, 2 for y and 3 for z. Usage:

```
LOADSLICE
  type (char)
  1/2/3 (int)  val (float)
```

LOADVSLICE Load a vector slice file of a given type. The type is the same as what Smokeview displays in the Load menus for slice files. The plane orientation is specified by using 1 for x, 2 for y and 3 for z. Usage:

```
LOADVSLICE
 type (char)
 1/2/3 (int)  val (float)
```

LOADVOLSMOKE Load files needed to view volume rendered smoke. One may either load files for all meshes or for one particular mesh. Usage:

```
LOADVOLSMOKE
 mesh number (-1 for all meshes) (int)
```

LOADVOLSMOKEFRAME Load a volume rendered smoke frame. Usage:

```
LOADVOLSMOKEFRAME
 mesh_index (int) frame_index (int)
```

If the mesh_index is positive then volume rendered smoke for that mesh index is loaded. If the mesh_index is negative then volume rendered smoke for all meshes is loaded.

UNLOADALL Unload all data files currently loaded. Usage:

```
UNLOADALL
```

8.3.2 Controlling the Scene

EXIT Cause Smokeview to quit. Usage:

```
 EXIT
```

CBARFLIP Display the colorbar in the opposite orientation from which it was defined. Usage:

```
 CBARFLIP
```

CBARNORMAL Usage: Display the colorbar in the same orientation as it was defined.

```
CBARNORMAL
```

KEYBOARD Passes a keyboard character to Smokeview Usage:

```
KEYBOARD
 c
```

or

```
KEYBOARD
 ALT c
```

where `c` is any keyboard character (recognized by Smokeview) and `ALT` is the ALT key.

LOADTOUR Load a tour of a given name. Usage:

```
LOADTOUR
 type (char)
```

SETTIMEVAL Set the time for displaying data to a specified value. Usage:

```
SETTIMEVAL
  time (float)
```

SETVIEWPOINT Set a view point . The view point must have been previously defined and saved in an .ini file. Usage:

```
SETVIEWPOINT
  viewpoint (char)
```

UNLOADTOUR Unload a tour.

```
UNLOADTOUR
```

XYZVIEW Sets the position (x, y and z location) and view direction (azimuth and elevation angle in degrees). Usge:

```
XYZVIEW
 xpos ypos zpos az elev
```

VIEWXMIN,VIEWYMIN,VIEWZMIN The `VIEWXMIN` script keyword sets the view so that the scene is viewed towards the XMIN direction. It sets the scene view location to (x,ymid,zmid) and the scene view direction to $(-1.0, 0.0, 0.0)$ where x is chosen so that the entire scene is in view and ymid, zmid are the middle of the scene along y and z directions. The `VIEWYMIN` and `VIEWZMIN` script keywords are defined similarly. Usage:

```
VIEWXMIN
VIEWYMIN
VIEWZMIN
```

VIEWXMAX,VIEWYMAX,VIEWZMAX The `VIEWXMAX` script keyword sets the view so that the scene is viewed towards the XMAX direction. It sets the scene view location to (x,ymid,zmid) and the scene view direction to $(+1.0, 0.0, 0.0)$ where x is chosen so that the entire scene is in view and ymid, zmid are the middle of the scene along y and z directions. The `VIEWYMAX` and `VIEWZMAX` script keywords are defined similarly. Usage:

```
VIEWXMAX
VIEWYMAX
VIEWZMAX
```

XSCENECLIP Sets clipping planes along the X axis. Portions of the scene before `xmin` or after `xmax` are hidden if setxmin or setxmax are set to 1 respectively. Usage:

```
XSCENECLIP
  setxmin xmin setxmax xmax
```

YSCENECLIP Sets clipping planes along the Y axis. Portions of the scene before `ymin` or after `ymax` are hidden if setymin or setymax are set to 1 respectively. Usage:

```
YSCENECLIP
  setymin ymin setymax ymax
```

ZSCENECLIP Sets clipping planes along the Z axis. Portions of the scene before `zmin` or after `zmax` are hidden if setzmin or setzmax are set to 1 respectively. Usage:

```
ZSCENECLIP
  setzmin zmin setzmax zmax
```

8.3.3 Rendering Images

RENDERCLIP Specify clip offsets in pixels when rendering a scene. Usage:

```
RENDERCLIP
  flag left right bottom top
```

where clipping is turn on if `flag` is set to 1 and turned off if `flag` is set to 0.

RENDERDIR Specify a directory where rendered files should go. Usage:

```
RENDERDIR
  directory name
```

Smokeview automatically converts directory separators ('/' for Linux/Mac systems and '//' for Windows systems) to the separator appropriate for the host system.

RENDERONCE Render the current scene. Usage:

```
RENDERONCE
 file name (optional)
```

Smokeview will assign the filename automatically if the entry after the `RENDERONCE` keyword is blank.

RENDERDOUBLEONCE Render the current scene at double resolution. Usage:

```
RENDERDOUBLEONCE
 file name (optional)
```

As with `RENDERONCE`, Smokeview will assign the filename automatically if the entry after the `RENDERDOUBLEONCE` keyword is blank.

RENDERALL Render a sequence of frames. By default this command renders every frame starting with the first. One may also specify a starting frame index (default: 0) and a skip value (default: 1) indicating the difference in indices between rendered frames. Usage:

```
RENDERALL
  skip first
  file name base (char) (or blank to use the Smokeview default)
```

The command

```
RENDERALL
  1 0
  casename
```

would render all frames while the command

```
RENDERALL
  3 1
  casename
```

renders every third frame starting with the second (index 1).

VOLSMOKERENDERALL Render a sequence of volume rendered frames. As with `RENDERALL`, this command by default command renders every frame starting with the first. One may also specify a starting frame index (default: 0) and a skip value (default: 1) indicating the difference in indices between rendered frames.

This command differs from the RENDERALL command by automatically loading volume rendered smoke files but only one frame (time step) at a time as they are rendered. This allows one to create volume rendered movies for data sets too large to be viewed within Smokeview.

```
VOLSMOKERENDERALL
  skip first
  file name base (char) (or blank to use the Smokeview default)
```

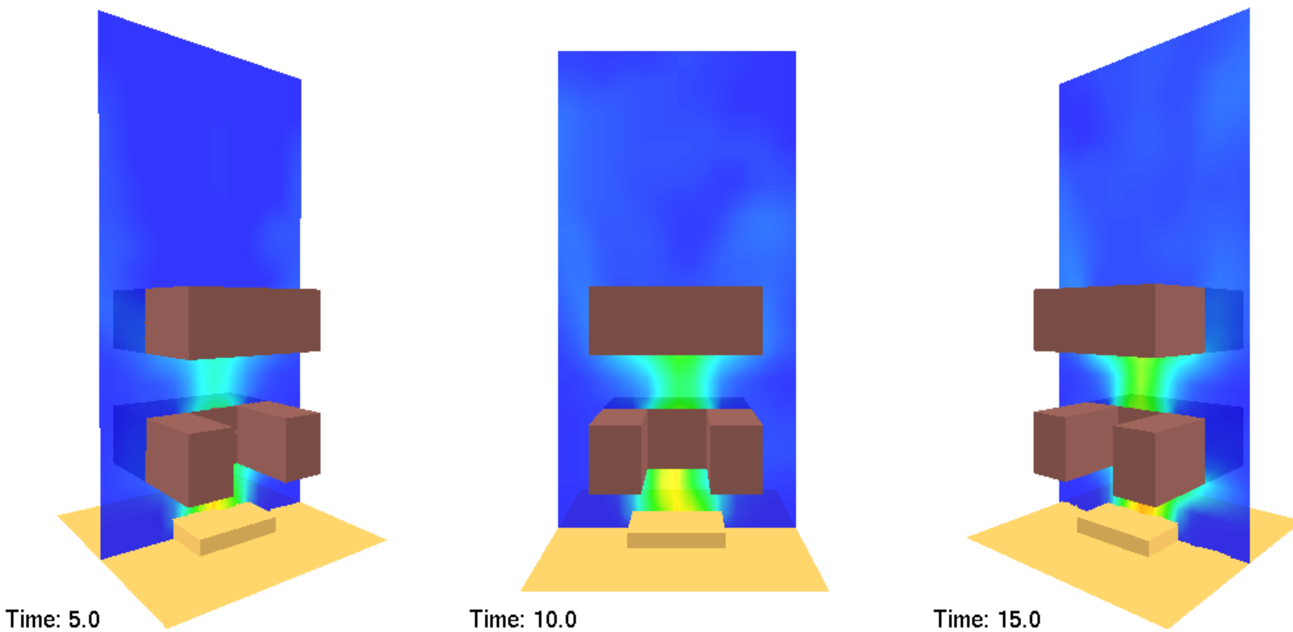


Figure 8.5: Smokeview images generated using script detailed in Fig. 8.4

Part III

Miscellaneous Topics

Chapter 9

Coloring Data

9.1 Overview

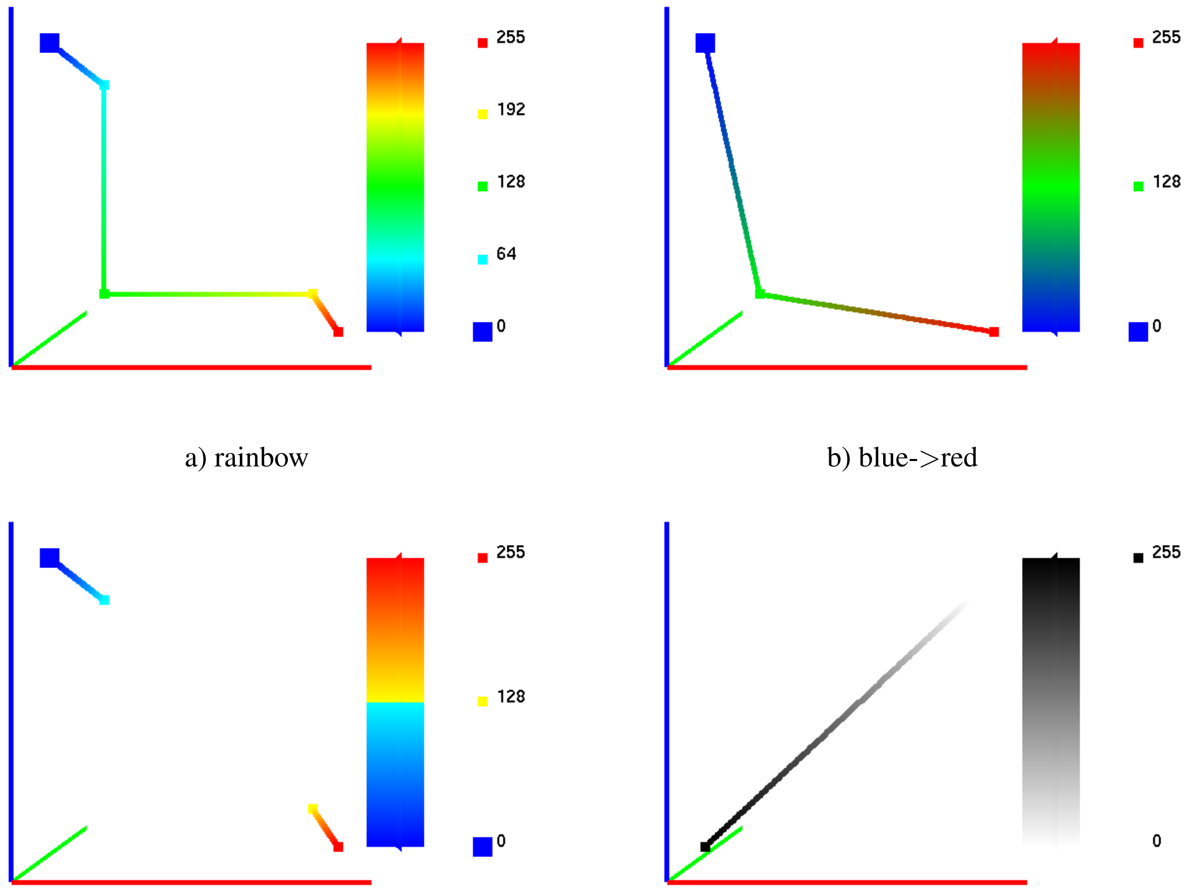
A colorbar is used to map data with color. A colorbar is normally visualized by displaying its color in sequence forming a bar or rectangle. When designing a colorbar, it is convenient to also visualize it by thinking of color spatially associating red, green and blue color components with x, y and z spatial coordinates. A colorbar may then be thought of as a path within a cube where the lower left bottom cube corner is colored black and the upper right top corner is colored white. Other corners are colored red, green, blue, cyan, magenta and yellow depending on their color components present. Figure 9.1 gives several examples of colorbars pre-defined by Smokeview. Each image was generated using the colorbar editor illustrated in Fig. 9.2.

A colorbar in Smokeview consists of a set of color nodes forming a path within a cube. This way of visualizing it is helpful in defining new colorbars by allowing one to more easily judge changes in color within the colorbar. Though most colorbars paths are continuous, a colorbar path need not be. Discontinuous colorbars are useful for highlighting regions in a simulation with a particular property, for example where the temperature exceeds the boiling point of water or in a topographic map where a shoreline (zero elevation) occurs. Figure 9.1c gives an example of a colorbar with a break. This colorbar jumps in the middle from a shade of cyan to a shade of yellow.

9.2 Using the Colorbar Editor

The Colorbar Editor dialog box is opened by selecting the *Dialogs>View>Edit colorbar* menu entry. When this menu item is selected, a spatial representation of the currently selected colorbar is displayed within the Smokeview scene along with the Colorbar Editor dialog box. The FDS simulation scene is hidden by default but may be shown along with the colorbar display by un-checking the *Hide scene* checkbox.

The colorbar is represented visually in two ways. First, as a series of colored nodes and lines within a unit cube. The cube has axese red (x direction), green (y direction) and blue (z direction). The nodes and lines are a spatial representation of the colorbar where as stated earlier the r, g, b color components are mapped to x, y, z spatial coordinates. Second, as a rectangle with a series of colored squares and numbered indices displayed along side. (This rectangle is equivalent to the colorbar displayed beside a regular Smokeview scene). The numbered indices indicate the position in the colorbar where the node color occurs. Once the node indices and colors are defined, Smokeview interpolates to form a table of colors (256 rows, 3 columns).



a) rainbow b) blue->red

c) blue->red split d) white->black

Figure 9.1: Colorbar Examples. Several colorbars are presented both as a 1D strip of changing color, each color corresponding to a different data value and as a 3D path where the x, y, z spatial locations of a color node correspond to the red, green and blue components of the color at that node.

The Colorbar Editor dialog box contains a list of colorbars pre-defined by Smokeview and others if defined by the user. A new colorbar is created by selecting the `New` button. The new colorbar is created initially by making a copy of the currently selected colorbar. Once created, it may be altered by adding/deleting nodes with the `Add/Delete` buttons and altering color with the red, green, blue spinners. The colorbar nodes may also be selected with the mouse and dragged to change their position and hence their value. If the `CTRL` key is pressed when dragging then the node only moves along a horizontal direction. If the `ALT` key is pressed when dragging then the node is constrained along a vertical direction. Note that only colorbars created by a user may be changed. The *Add/Delete* and other buttons for modifying colorbar characteristics are only enabled for user defined colorbars. They are disabled for Smokeview predefined colorbars. Colorbar definitions (only colorbars created by the user) are saved in the Smokeview configuration (.ini) file. The `Distribute nodes uniformly` button is used as the name infers to distribute the individual colors of the colorbar uniformly along the colorbar path.

The bottom portion of the Colorbar Editor dialog box is used to define colors for extreme data. That is, data occurring below the specified minimum or above the specified maximum. This data may be highlighted by selecting the *Highlight Extreme Data* checkbox. The color used to highlight this data may also be specified. The colors defined using this dialog box are shown in the triangular regions at the top and bottom of the colorbar.

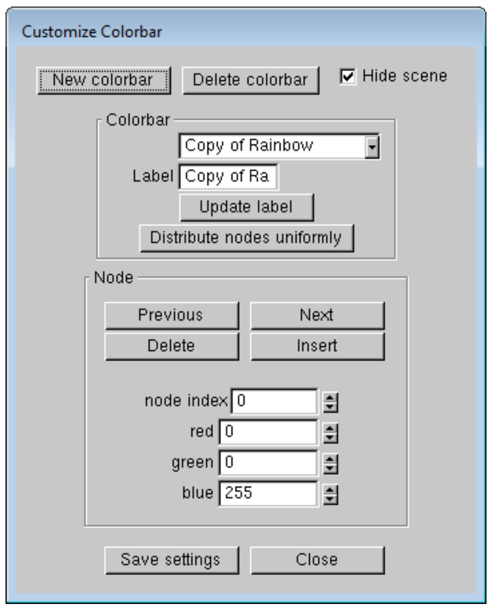


Figure 9.2: Colorbar Editor dialog box. One may create or edit existing colorbars by specifying a sequence of red/green/blue coordinates.

Chapter 10

Smokeview - Demonstrator Mode

A simplified version of Smokeview may be invoked in order to present a fire scenario for training or demonstration purposes. All actions are performed using one unified dialog box, illustrated in Fig. 10.1. This dialog box is opened for the user at startup and allows the user to select data to be viewed, tours to travel along, viewpoints to observe and scene manipulation to perform. Smokeview loads data when it starts up. The intent is to allow one not using Smokeview daily to more easily make use of Smokeview's capabilities.

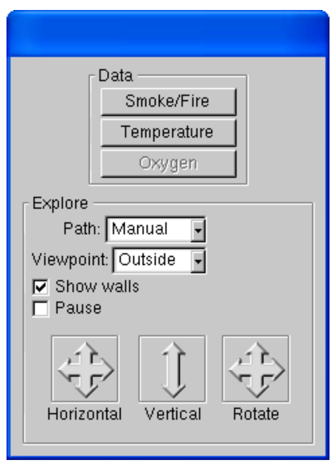


Figure 10.1: Demonstrator dialog box. This dialog box allows the user to 1) switch between temperature, oxygen and realistic views of the data, select tours and viewpoints and to manipulate the scene using translations and rotations.

In order to setup this demonstration mode, several tasks need to be performed. The results of these tasks are recorded in the `casename.ini` file. These tasks are detailed below.

1. Define one or more tours that give the user an overview of the data or that highlight important aspects of the scenario. Tours are setup using the Touring dialog box.

2. Define one or more viewpoints that highlight some important aspect of the simulation scenario. The viewpoint is defined by manipulating the scene as desired and then selecting the *View>Save* menu item. The viewpoint label may be changed by using the Motion/View/Render dialog box.

3. Pick the data to be viewed from a set of temperature and oxygen slice files and a set of 3D smoke and HRRPUV files.
   (a) Load the desired files into Smokeview.
   (b) Select these files for *auto-loading* by selecting the *Auto Load Now* panel in the File/Bounds dialog box and pressing the `Save Auto Load File List` button.
   (c) Compress these files with Smokezip using the `-auto` option . This option will only compress files selected with Smokeview for *autoloading*. Note that compression can either be performed at a command line by typing `Smokezip -auto casename` or by using the *Load/Unload>Compression* menu item.
4. Save the settings and choices selected by saving a *casename.ini* configuration file for the case.
5. Create a `.svd` file by copying the `casename.smv` to `casename.svd`.
6. Copy all the compressed files and the files: `casename.ini`, `casename.end` and `casename.svd` file to a separate directory. This directory is then is what one would distribute to be demonstrated.

The demonstrator mode of Smokeview is activated by double-clicking on `casename.svd`. Smokeview treats this file just like `casename.smv` except that it opens up the Demonstrator Mode dialog box and hides the standard Smokeview menus. Smokeview then loads the selected slice, 3D smoke and HRR files and opens the dialog box illustrated in Fig. 10.1.

This dialog box is used to toggle the data viewed by pressing the `Smoke/Fire`, `Temperature` or `Oxygen` buttons. The scene may be manipulated by clicking the mouse in one of the *arrow* buttons and dragging. The scene may also be manipulated as before by pressing the mouse within the scene and dragging. Views and/or tours may be selected using the corresponding *pull down* box.

Chapter 11

Texture Maps

Texture mapping is a technique used by Smokeview to make a scene appear more realistic by pasting images onto obstructions or vents. For example, to apply a wood paneling image to a wall, add the keywords `TEXTURE_MAP='paneling.jpg', TEXTURE_WIDTH=1., TEXTURE_HEIGHT=2.` to the `&SURF` line where **paneling.jpg** is the JPEG file containing the texture map (SGI users should use RGB files instead of JPEG) and **TEXTURE_WIDTH** and **TEXTURE_HEIGHT** are the characteristic dimensions of the texture map in meters. Note that the image will not appear when Smokeview first starts up. The user must select the texture maps using the `Show/Hide` menu.

One can create texture maps using a digital camera or obtain them commercially. The maps should be *seamless* so that no breaks or seams appear when the maps are tiled on a blockage or vent. This is important, because Smokeview replicates the image as often as necessary to cover the blockage or vent.

When the texture does have a pattern, for example windows or bricks, the keyword `TEXTURE_ORIGIN` may be used to specify where the pattern should begin. For example,

```
&OBST XB=1.0,2.0,3.0,4.0,5.0,7.0, SURF_ID='wood paneling',
      TEXTURE_ORIGIN=1.0,3.0,5.0 /
```

will apply paneling to an obstruction whose dimensions are 1 m by 1 m by 2 m, such that the image of the paneling will be positioned at the point (1.0,3.0,5.0). The default value of `TEXTURE_ORIGIN` is (0,0,0), and the global default can be changed by added a `TEXTURE_ORIGIN` statement to the `MISC` line.

Figure 11.1 shows a simple application of a texture applied to two different blockages and a vent. The same jpeg file was used in two different `&SURF` lines so that the texture could be stretched by differing amounts (using the `TEXTURE_WIDTH` parameter.) The FDS data file used to create this Figure follows.

```
&HEAD CHID='sillytexture', TITLE='Silly Test Case' /
&MISC TEXTURE_ORIGIN=0.1,0.1,0.1 /

&MESH IJK=20,20,20, XB=0.0,1.0,0.0,1.0,0.0,1.0 /
&TIME T_END=0. /
&SURF ID     = 'TEXTURE 1'
      TEXTURE_MAP= 'nistleft.jpg'
      TEXTURE_WIDTH=0.6
      TEXTURE_HEIGHT=0.2 /

&SURF ID     = 'TEXTURE 2'
      TEXTURE_MAP= 'nistleft.jpg'
      TEXTURE_WIDTH=0.4
      TEXTURE_HEIGHT=0.2 /

&OBST XB=0.1,0.3,0.1,0.7,0.1,0.3, SURF_ID='TEXTURE 1'  /
&OBST XB=0.5,0.9,0.3,0.7,0.1,0.5, SURF_ID='TEXTURE 2',
```

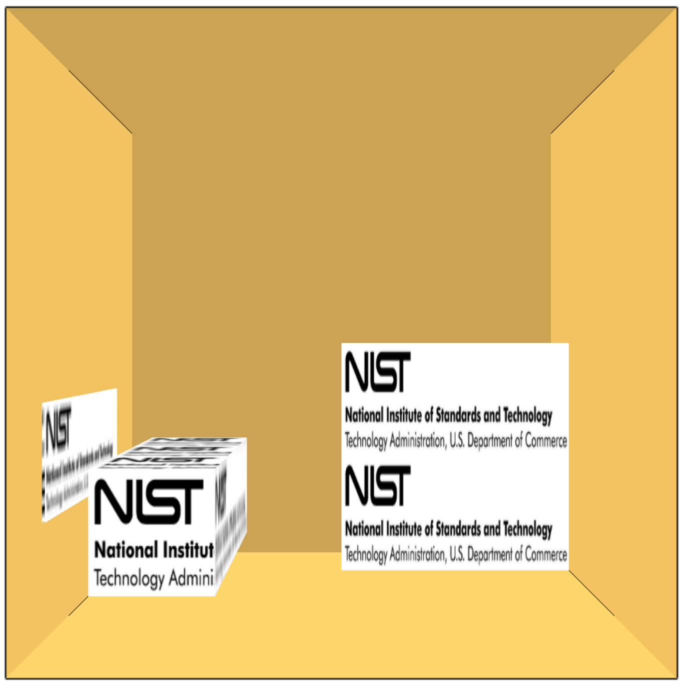


Figure 11.1: Texture map example. The same texture was applied to two different blockages and a vent (with different widths) by assigning different TEXTURE_WIDTH parameters in the input file.

```
      TEXTURE_ORIGIN=0.5,0.3,0.1  /

&VENT XB=0.0,0.0,0.2,0.8,0.2,0.4, SURF_ID='TEXTURE 1',
      TEXTURE_ORIGIN=0.0,0.2,0.2 /
&VENT XB=0.3,0.9,0.0,0.0,0.3,0.5, SURF_ID='TEXTURE 1',
      TEXTURE_ORIGIN=0.9,0.0,0.3 /
&TAIL /
```

Chapter 12

Using Smokeview to Debug FDS Input Files

One of the most difficult tasks in setting up an FDS input file is defining the geometry (blockages, vent locations, etc.) properly. Smokeview may be used to debug FDS input files by making short model runs and observing whether blockages, vents and other geometric features of a model run are located correctly. Blockages may then be created or changed using a text editor and location information provided by the Examine geometry dialog box called from the `Dialogs>View>Examine geometry` menu.

The following is a general procedure for identifying problems in FDS input files. Assume that the FDS input data file is named `testcase1.fds`.

1. In the FDS input file, set the stop time to 0.0 using `TWFIN=0.0` on the `&TIME` line. This causes FDS to read the input file and create a `.smv` file without performing lengthy startup calculations.

2. Run the FDS model (for details see the FDS User's Guide [4])

   FDS creates a file named `testcase1.smv` containing information that Smokeview uses to visualize model.

3. To visualize the model, open `testcase1.smv` with Smokeview by either typing `smokeview testcase1` at a command shell prompt or if on the PC by double-clicking the file `testcase1.smv`.

4. Make corrections to the FDS data file, if necessary. Using the `COLOR` or `RGB` option of the `OBST` keyword to more easily identify blockages to be edited. For example, to change a blockage's color to red use:

   ```
   &OBST XB=0.0,1.0,0.0,1.0,0.0,1.0, COLOR='RED' /
   ```

   or

   ```
   &OBST XB=0.0,1.0,0.0,1.0,0.0,1.0 RGB=255,0,0 /
   ```

   Save testcase1.fds file and go back to step 2.

5. If corrections are unnecessary, then change the `TWFIN` keyword back to the desired final simulation time, remove any unnecessary FDS `COLOR` keywords and run the case.

12.1 Examining Blockages

Blockages locations and SURF properties may be examined by selecting the menu item *Examine Blockages* which opens up the dialog box illustrated in Fig. 12.1. Note, clipping planes need to be turned off when using this dialog box. Associating unique colors with each surface allows the user to quickly determine whether blockages are defined with the proper surfaces. One can then verify that these modeling elements have been defined and positioned as intended. Position coordinates are displayed *snapped* to the nearest grid line or as specified in the input file.

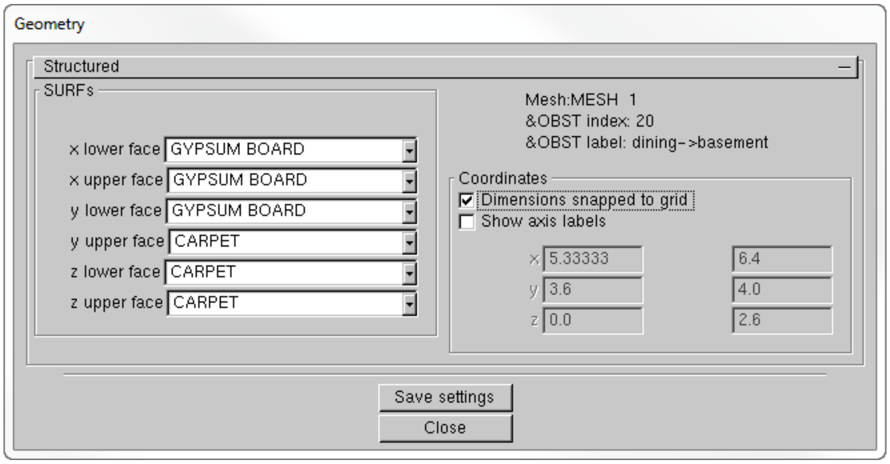


Figure 12.1: Examine blockages dialog box.

Chapter 13

Making Movies

A movie may be made of a Smokeview animation by converting the scene into a series of still images, one image for each time step and then combining the images into a movie file. The images may be combined using a commercial program such as Antechinus Media Editor (`http://www.c-point.com`), Apple Quicktime Pro (`http://www.quicktime.com`), or Adobe Premiere Pro (`http://www.adobe.com`) or the public domain program ffmpeg (`https://www.ffmpeg.org`). Smokeview will add a dialog box for making a movie if ffmpeg is installed. The steps to making a movie are:

1. Set up Smokeview by orienting the scene and loading the desired data files.

2. Select the *Options/Render* menu and pick the desired frame skip value. The more frames you include in the animation, the smoother it will appear. Of course, more frames result in larger file sizes. Choose fewer frames if the movie is to appear on a web site.

   The dialog box illustrated in Figure 13.1 may also be used for generating an image sequence. Widgets exist for selecting the image type, the number of frames to skip between images and for creating the image sequence. One may also select a clipping region making the final image size smaller.

3. Use a program such as the Antechinus Media Editor, Apple Quicktime Pro or Adobe Premiere Pro, to assemble the JPEGS or PNGS rendered in the previous step into a movie file. If ffmpeg is installed use the Movie dialog illustrated in Figure 13.1.

The default Smokeview image size is 640×480 . This size is fine if the movie is to appear in a presentation located on a local hard disk. If the movie is to be placed on a web site then care needs to be taken to insure that the generated movie file is a reasonable size. Two suggestions are to reduce the image size to 320×240 or smaller by modifying the `WINDOWWIDTH` and `WINDOWHEIGHT` `smokeview.ini` keywords and to reduce the number of frames to 300 or less by skipping intermediate frames *via* the *Options/Render* menu.

Sometimes when copying or *capturing* a Smokeview scene it is desirable, or even necessary, to have a margin around the scene. This is because the capturing system does not include the entire scene but itself captures an indented portion of the scene. To indent the scene, either press the “h” key or select the *Option>Viewpoint>Offset Window* menu item. The default indentation is 45 pixels. This may be changed by adding/editing the WINDOW_OFFSET keyword in the `smokeview.ini` file.

Note, the Smokeview animation must be running when the render command is selected or only one frame will be saved instead of the entire image sequence.

Volume rendered smoke files which are really 3D slice files can be quite large. Normally Smokeview loads an entire data set before visualizing it. When creating image sequences for volume rendered smoke

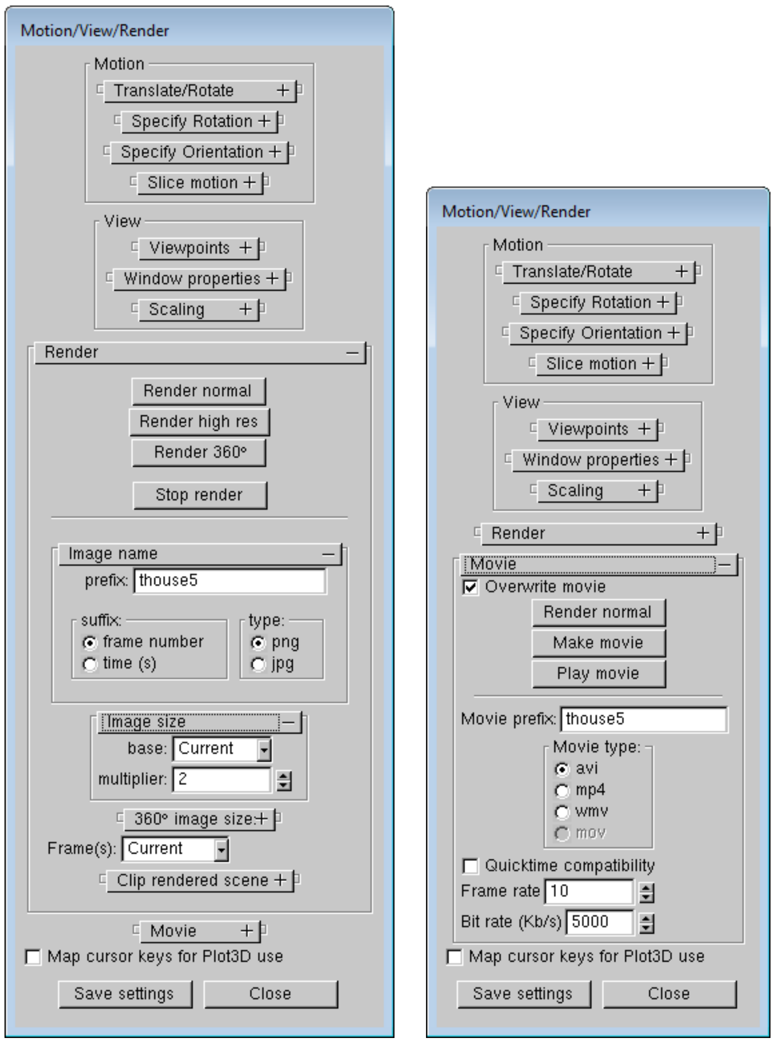


Figure 13.1: Render and Movie dialog box. These dialog boxes allow one to specify options for rendering an image sequence of the displayed scene and to combine these images into a movie. The Movie dialog box is available if the program ffmpeg is installed.

files, Smokeview allows one to load data and create an image one frame or time step at a time. Figure 13.2 shows the dialog for doing this. This dialog is a part of the 3D smoke dialog box. One can specify the starting frame index and the number of frames to skip. This allows one to run multiple Smokeview's in parallel.

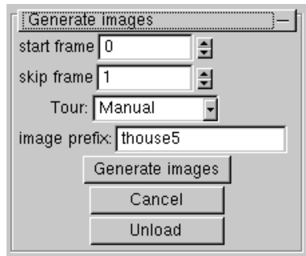


Figure 13.2: Volume Render dialog box. This dialog box allows one to specify options for rendering an image sequence of a volume rendered smoke file. Data is loaded and an image is constructed one frame at a time allowing movies to be made of cases with very large data sets.

Chapter 14

Annotating the Scene

14.1 Overview

Smokeview scenes may be annotated by adding *ticks* with associated text documenting their location or by adding text strings at arbitrary locations and time durations. Both of these annotations are added with the *User Ticks* and the User Label dialog box respectively. These dialog boxes are both invoked by selecting the *Dialogs>Display* menu item.

14.2 User Ticks Settings Dialog Box

The User Ticks Settings dialog box allows one to place ticks and labels along one or more coordinate axes. The user may specify tick spacing, number of sub-tick intervals and how far axes extend. There is an automatic placement option that allows the tick axes to be placed based upon the orientation of the scene. The user may specify which tick axes are visible if the automatic placement option is not invoked. Figure 14.1 illustrates the User Ticks Settings dialog box. It is a panel of the Display dialog box. Figure 14.2 shows the ticks and labels resulting from the dialog box.

14.3 *User Label* Dialog Box

The User Label dialog box allows one to place text strings at arbitrary locations within the scene. The user may also control the time interval when they are visible. Figure and at arbitrary time intervals. Figure 14.3 illustrates the User Label dialog box. It is a panel within Display dialog box. It has controls for specifying the text string, (x,y,z) location, start and stop time and color.

14.4 TICKS and LABEL keywords

Tick marks and label annotation can be also placed within the 3D scene using the TICKS and LABEL keywords. FDS places tick marks and labels documenting the scene dimensions. To replace or customize these annotations add the `TICK` keyword to a .smv file using the following format:

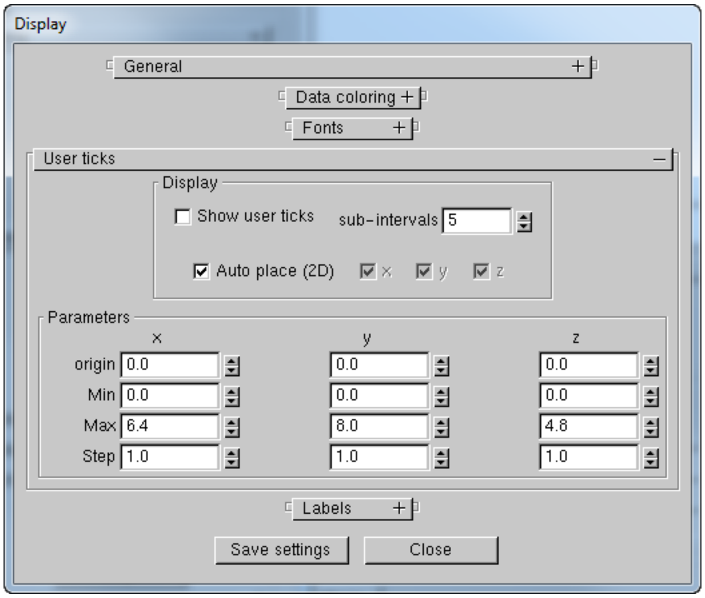


Figure 14.1: Ticks dialog box. The Ticks dialog box is invoked by selecting *Dialogs>Display*.

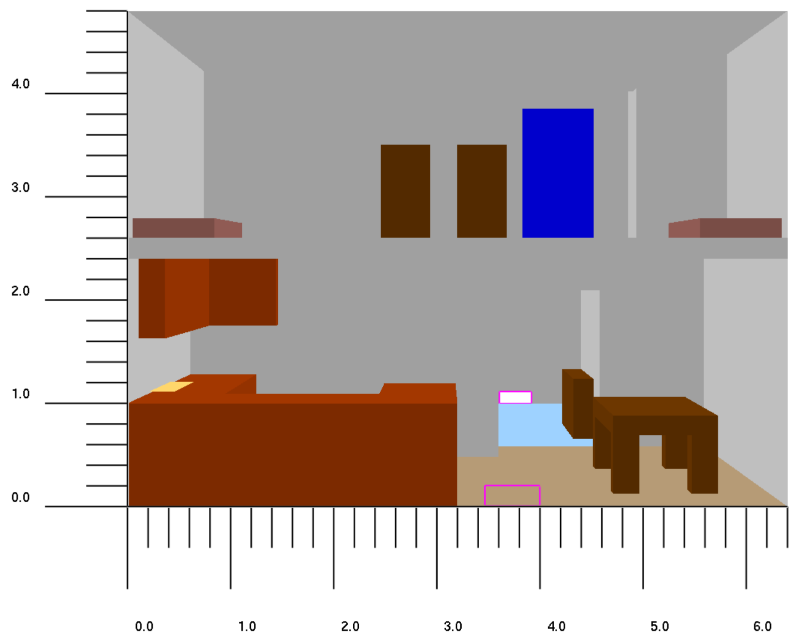


Figure 14.2: Annotation example using the Ticks dialog box

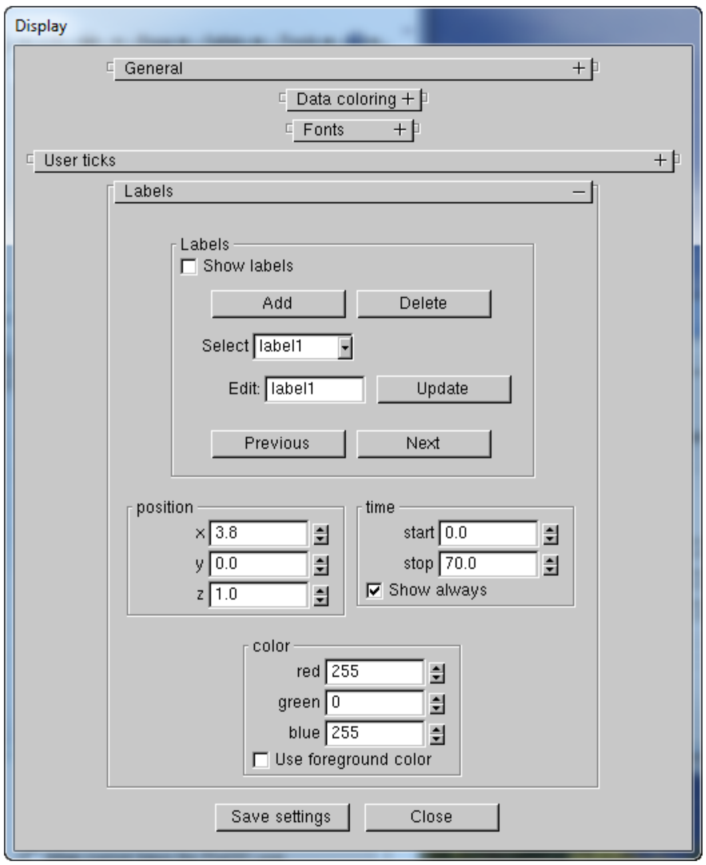


Figure 14.3: User Label dialog box. The User Label dialog box is invoked by selecting *Dialogs>Display*.

```
TICKS
xb yb zb xe ye ze nticks
ticklength tickdir r g b tickwidth
```

where `xb`, `yb`, and `zb` are the x, y and z coordinates of the first tick; `xe`, `ye` and `ze` are the x, y and z coordinates of the last tick and `nticks` is the number of ticks. The coordinate dimensions are in physical units, the same units used to set up the FDS geometry. The parameter `ticklength` specifies the length of the tick in physical units. The parameter `tickdir` specifies the tick direction. For example 1(-1) places ticks in the positive(negative) x direction. Similarly, 2(-2) and 3(-3) place ticks in the positive(negative) y and positive(negative) z directions.

The color parameters `r`, `g` and `b` are the red, green and blue components of the tick color each ranging from 0.0 to 1.0. The foreground color (white by default) may be set by setting any or all of the `r`, `g` and `b` components to a negative number. The `tickwidth` parameter specifies tick width in pixels. Fractional widths may be specified.

The `LABEL` keyword allows a text string to be added within a Smokeview scene. The label color and start and stop appearance time may also be specified. The format is given by

```
LABEL
x y z r g b tstart tstop
label
```

where (`x, y, z`) is the label location in Cartesian coordinates and `r, g, b` are the red, green and blue color components ranging from 0.0 to 1.0. Again, if a negative value is specified then the foreground color will be used instead (white is the default). The parameters, `tstart` and `tstop` indicate the time interval when the label is visible. The text string is specified on the next line (`label`).

Figure 14.4 shows how the `TICKS` and `LABEL` keywords can be used together to create a *ruler* with major and minor tick marks illustrated in Fig. 14.5.

```
TICKS
0.0 0.0 0.0 8.0 0.0 0.0 5
0.5 -2.0 -1. -1.0 -1.0 4.0
TICKS
1.0 0.0 0.0 9.0 0.0 0.0 5
0.25 -2.0 -1. -1.0 -1.0 4.0
TICKS
0.0 0.0 0.0 0.0 0.0 2.0 3
0.5 -1.0 -1. -1.0 -1.0 4.0
TICKS
0.0 0.0 0.0 0.0 4.0 0.0 5
0.5 -1.0 -1. -1.0 -1.0 4.0
LABEL
0.0 -0.6 0.0 -1.0 0.0 0.0 0.0 20.0
0
LABEL
2.0 -0.6 0.0 -1.0 0.0 0.0 0.0 20.0
2
LABEL
4.0 -0.6 0.0 -1.0 0.0 0.0 0.0 20.0
4
LABEL
6.0 -0.6 0.0 -1.0 0.0 0.0 0.0 20.0
6
LABEL
8.0 -0.6 0.0 -1.0 0.0 0.0 0.0 20.0
8
LABEL
9.5 -0.6 0.0 -1.0 0.0 0.0 0.0 20.0
m
```

Figure 14.4: TICKS and LABEL commands used to create image in Fig. 14.5

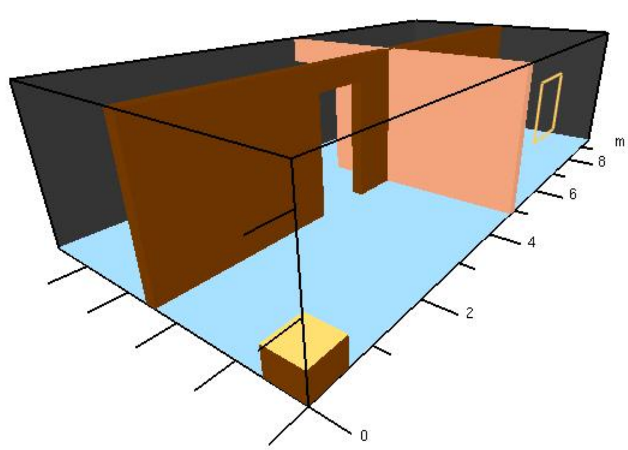


Figure 14.5: Annotation example using the TICKS and LABEL keyword.

Chapter 15

Utilities

Several utilities are included with the FDS/Smokeview distribution allowing one to more easily analyze and generate data. Smokezip may be used to compress FDS data files resulting in quicker load times in Smokeview. Smokediff may be used to compare two FDS cases. Smokediff generates another .smv file and a set of data files which can be viewed with Smokeview. Background may be used to take advantage of multiple core computers by running more than one FDS case at a time. This is most useful when running a long list of FDS cases. Background runs a case whenever the CPU load is below a specified level.

15.1 smokezip - A utility for reducing FDS file sizes

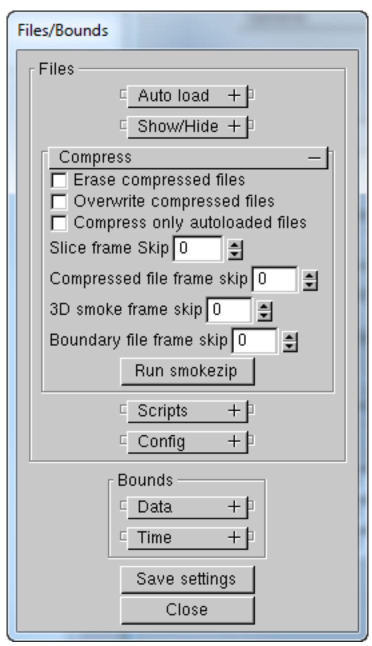


Figure 15.1: File/Bounds dialog box showing compression and autoload options. 3D smoke, boundary and slice files may be compressed using Smokezip. All currently loaded files may be loaded automatically when Smokeview first starts by selecting the autoload checkbox.

3D smoke, boundary, isosurface and slice files may be compressed using the utility Smokezip. FDS data files may also be compressed from within Smokeview using the compression menu item found in the *Load/Unload* menu. File compression may also be activated from the Compressions, Autoload section of the File/Bounds dialog box illustrated in Fig. 15.1. Compression is performed using the ZLIB compression library (see `http://www.zlib.org`). Smokeview compresses files in the background allowing one to continue visualizing cases. Smokeview adds the label *ZLIB* to Load menu entries for any file that has been compressed. Smokezip adds the extension .svz to any FDS data file that has been compressed.

The usage for Smokezip (which may be obtained by typing Smokezip -help at a command line) is

```
  smokezip 1.5.0(SMV6.7.15-0-g4fa68dd) - Aug 21 2020

  Compress FDS data files

  smokezip [options] casename

  casename - Smokeview .smv file for case to be compressed

options:
  -c  - cleans or removes all compressed files
  -t nthread - Compress nthread files at a time (up to 16)
  -help      - display help summary
  -help_all  - display all help info
  -version   - display version information
```

Smokezip either determines data bounds itself (if the -bounds option was specified) or uses min and max values found in the casename.ini file. These bounds are used to map four byte floating point data found in FDS data files to one byte color indices used by Smokeview. The algorithms for determining the data mappings used by Smokeview and Smokezip are identical so it should result in the same views.

Particle files may be converted to isosurface files using the `-part2iso` option as in `Smokezip -part2iso casename`. The resulting isosurface file highlights particle boundaries (where particle density is 0.5 particles per grid cell). These isosurface files are accessible in the .smv file named `casename_smvzip.smv`.

15.2 smokediff - A utility for comparing two FDS cases

The utility Smokediff compares two FDS cases with the same geometry. Smokediff examines two `.smv` files looking for boundary, slice and Plot3D files containing the same type of data and located in the same region in space. Data in one file is subtracted from corresponding data in the other. Smokediff then generates a new `.smv` file referencing the differenced boundary, slice and Plot3D data files. To compare two .smv files named `casename1.smv` and `casename2.smv` one would use the command

```
smokediff -smv casename1 casename2
```

The `-smv` option caused Smokeview to open after the differencing is complete to examine the differenced results. Smokediff allows the grid for slice files in casename2 to be refined by an integer multiple. Other usage options for Smokediff are detailed below

```
  smokediff [options] smv_case1 smv_case2
    version: 1.0.11 (githash SMV6.7.15-0-g4fa68dd) - Aug 21 2020

  smokediff compares two FDS cases by subtracting data referenced in smv_case2 from
```

```
corresponding data referenced in smv_case1 (smv_case1 - smv_case2).  Slice, PLOT3d
and boundary files are supported.  Differenced results may be viewed by opening
smv_case1_diff.smv in Smokeview or by using the -smv option when running smokediff.

Mesh bounds must be identical for corresponding meshes.  Mesh resolutions must be
identical when differencing boundary and PLOT3D files.  The x, y, and z mesh
resolutions in smv_case2 must be integer multiples of the corresponding x, y, z mesh
resolutions in smv_case1 when differencing slice files.

-help      - display help summary
-help_all  - display all help info
-version   - display version information

smv_case1,smv_case2 - Two smokeview cases to compare.
```

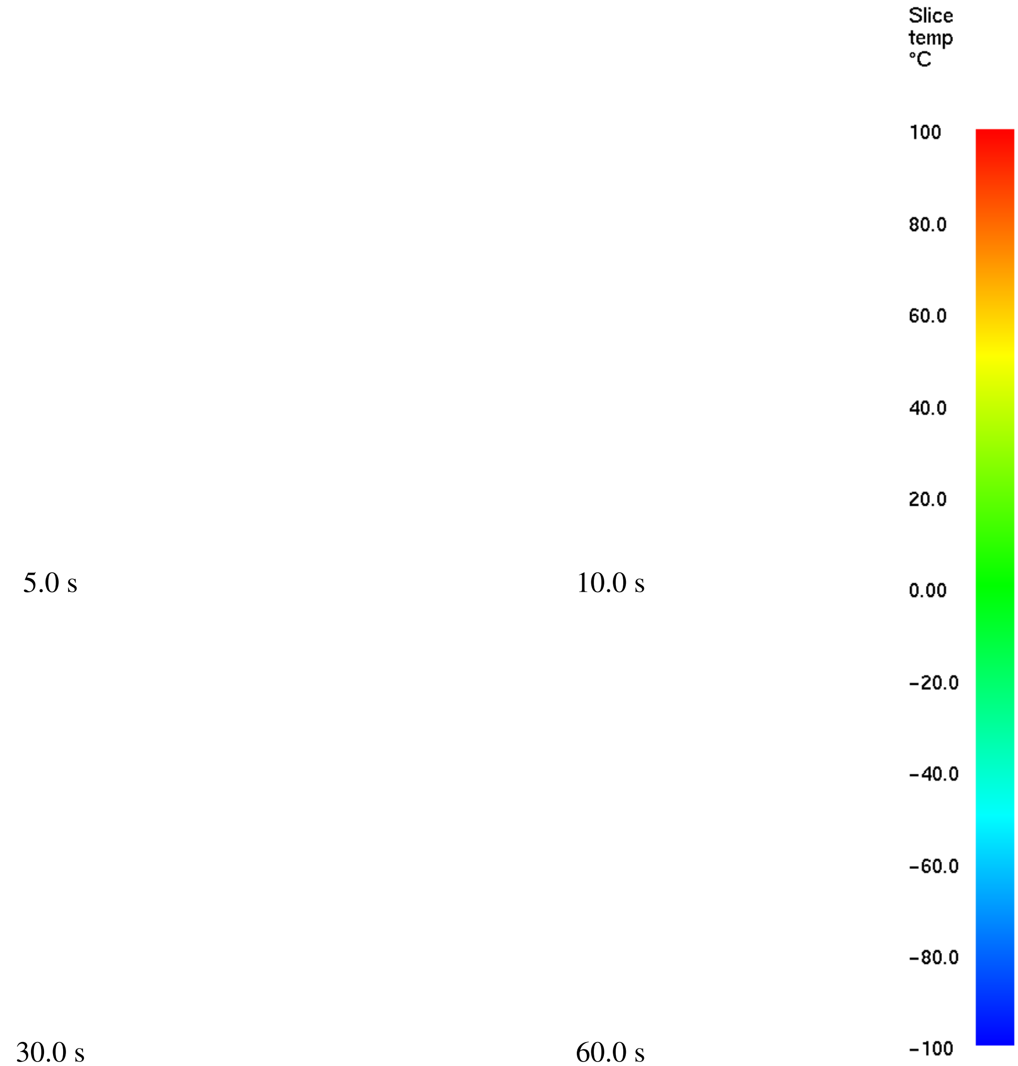


Figure 15.2: Slice file snapshots of differenced temperature data.

15.3 background - A utility for running multiple programs simultaneously

This section explains how to use the utility *background* (*background.exe* on the PC) what it is and how it might be useful to FDS users. It is included with the FDS/Smokeview bundle. A Windows user can use the *Start* command to run multiple programs at the same time. This is fine for a few programs but a computer can easily become overloaded if many jobs are run using this method. Similarly, a Linux/Mac user can run a program appending an `&` character to the end of a command line to put a job in the background. As on the PC, if many jobs are run, the system can become overloaded. The program *background* throttles job submissions so that a job won't start until the CPU load is below a specified level. A job submission can also be delayed until the memory usage is below a specified level. This enables one to submit a long list of FDS cases without saturating the CPU, since only a small number of jobs will be running at any one time.

The utility *background.exe* allows parallel processing to occur at the program level. It is often the case that one is doing a parameter study or running a long list of cases to verify the use of FDS. Typically you would create a windows batch file (.bat) containing a list of commands like

```
fds casename_1.fds
....
fds casename_n.fds
```

On a Windows system, each entry in the above list will not start until the previous entry has completed, even if the computer has multiple cores or CPUs. Unix/Linux based systems have the capability of putting computer jobs in the background, meaning that when a job is run, control returns immediately allowing the next job in the list to start running. With computers that have multiple cores or CPUS, one can then run more than one job simultaneously.

Here is how one might use *background* with FDS

```
background -d 1.0 -u 90 fds casename.fds
```

This command runs "fds casename.fds" after waiting 1 s and ensuring that the CPU usage is less than 90 %. If the CPU usage happens to be more than 90 %, the program *background* waits to submit the fds command until the usage drops below 90 % . Once this occurs, it runs the command, `fds casename.fds`.

The purpose of the delay before submitting a job is to give Windows a chance to update the usage level from previous invocations. This ensures that a large number of jobs are not submitted at once.

The background utility is designed to be used in a Windows batch file. For example, suppose you have a list of five FDS jobs you want to run in a Windows batch file. On a Windows computer you would have a batch file containing something like

```
fds case1.fds
fds case2.fds
fds case3.fds
fds case4.fds
fds case5.fds
```

Using background with a 2 second delay and 75 per cent maximum load level, you would change your script to something like

```
background -d 2 -u 75 fds case1.fds
background -d 2 -u 75 fds case2.fds
background -d 2 -u 75 fds case3.fds
background -d 2 -u 75 fds case4.fds
background -d 2 -u 75 fds case5.fds
```

Usage information for `background` may be obtained by typing `background -h` which gives output listed in Figure 15.3.

```
background 1.1.2(SMV6.7.15-0-g4fa68dd) - Aug 21 2020
  Runs a program in the background when resources are available

Usage:

  background [-d delay time (s) -h -u max_usage -v] prog [arguments]

where

  -d dtime  - wait dtime seconds before running prog in the background
  -m max    - wait to run prog until memory usage is less than max (25-100%)
  -u max    - wait to run prog until cpu usage is less than max (25-100%)
  -help      - display help summary
  -help_all  - display all help info
  -version   - display version information
  prog      - program to run in the background
  arguments - command line arguments of prog
```

Figure 15.3: Usage information for the program `background`

15.4 wind2fds - A utility for converting wind data for use by FDS and Smokeview

The utility `wind2fds` is designed to enable Smokeview to visualize wind velocity measurements made by SODARs (SOnic Detection And Ranging). A SODAR uses sound waves to measure wind velocity at various heights above ground level. Vertical wind profiles can then be visualized using Smokeview and compared with velocity profiles generated by FDS.

A typical (abbreviated) output generated by a SODAR is listed in Figure 15.4 where the `ws30`, `ws35` headings indicate wind speed and the `wd30`, `wd35` headings indicate wind direction. The utility, `wind2fds`, then converts this data into a form given in Figure 15.5. The converted wind data file begins with a `HEADER` section containing information about where various wind measurements were taken. Smokeview uses the information in the `HEADER` section to associate wind data measurements with locations. The `HEADER` section is followed by a `DATA` section. This section is identical to the file format used by Smokeview to visualize device data.

`wind2fds` allows one to link measurement locations with wind data, to exclude data before and/or after specified date/times and to label measurements as presented in Smokeview. Help information may be obtained at the command line by typing `wind2fds -h` as given in Figure 15.6.

Figure 15.7 gives an example of a visualization of wind data converted for use by Smokeview using wind2fds. The line segments represent wind speed and direction. The spherical shells represent uncertainty in wind direction (shell diameter) and wind speed (shell thickness).

```
date,          time, ws30, ws35, wd30, wd35
10/26/2011, 0:00:00,  6.5, 6.56,  333,  334
10/26/2011, 0:05:00, 7.83, 8.09,  336,  336
10/26/2011, 0:10:00, 8.36, 8.41,  334,  336
10/26/2011, 0:15:00, 7.6,  8.57,  335,  336
10/26/2011, 0:20:00, 7.44,  8.3,  340,  342
10/26/2011, 0:25:00, 7.92,  9.2,  343,  339
10/26/2011, 0:30:00, 9.36, 9.25,  336,  340
```

Figure 15.4: Experimental wind data.

```
//HEADER
DEVICE
 ws30 % VELOCITY % sensor
 0.0 0.0 30.0
DEVICE
 ws35 % VELOCITY % sensor
 0.0 0.0 35.0
DEVICE
 wd30 % ANGLE % sensor
 0.0 0.0 30.0
DEVICE
 wd35 % ANGLE % sensor
 0.0 0.0 35.0
//DATA
s,s,m/s,m/s,deg,deg
time,time_orig,ws30,ws35,wd30,wd35
0,0:00:00,6.5,6.56,333,334
300,0:05:00,7.83,8.09,336,336
600,0:10:00,8.36,8.41,334,336
900,0:15:00,7.6,8.57,335,336
1200,0:20:00,7.44,8.3,340,342
1500,0:25:00,7.92,9.2,343,339
1800,0:30:00,9.36,9.25,336,340
```

Figure 15.5: Experimental wind data converted by wind2fds for use by Smokeview.

```
wind2fds 1.0.1(SMV6.7.15-0-g4fa68dd) - Aug 21 2020
  Convert spreadsheets containing wind data to files compatible with Smokeview:

  wind2fds prog [-prefix label] [-offset x y z] datafile

where

  -prefix label  - prefix column headers with label
  -offset x y z  - offset sensor locations by (x,y,z)
  -help      - display help summary
  -help_all  - display all help info
  -version   - display version information

  datafile.csv   - spreadsheet file to be converted. Use '-' to input data
                   from standard input
```

Figure 15.6: Usage information for the program `wind2fds`.

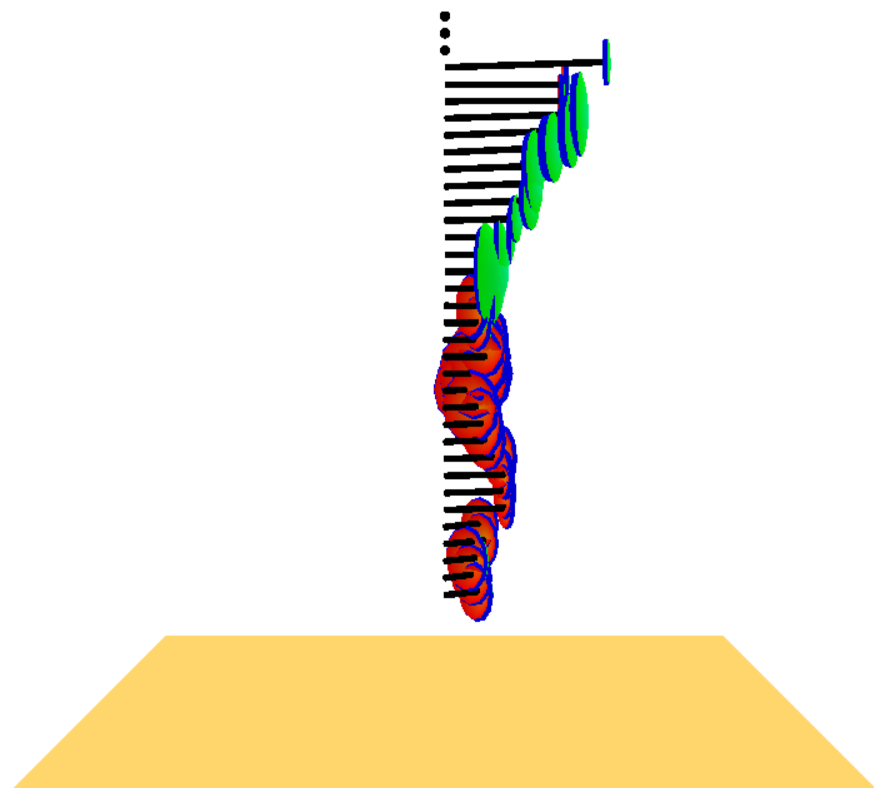


Figure 15.7: Visualization of wind data converted for use by Smokeview using wind2fds. The line segments represent wind speed and direction. The spherical shells represent uncertainty in wind direction (shell diameter) and wind speed (shell thickness).

Chapter 16

Summary

Often fire modeling is looked upon with skepticism because of the perception that eye-catching images shroud the underlying physics. However, if the visualization is done well, it can be used to assess the quality of the simulation technique. The user of FDS chooses a numerical grid on which to discretize the governing equations. The more grid cells, the better but more time-consuming the simulation. The payoff for investing in faster computers and running bigger calculations is the proportional gain in calculation accuracy and realism manifested by the images. There is no better way to demonstrate the quality of the calculation than by showing the realistic behavior of the fire.

Up to now, most visualization techniques have provided useful ways of analyzing the output of a calculation, like contour and streamline plots, without much concern for realism. A rainbow-colored contour map slicing down through the middle of a room is fine for researchers, but for those who are only accustomed to looking at real smoke-filled rooms, it may not have as much meaning. Good visualization needs to provide as much information as the rainbow contour map but in a way that speaks to modelers and non-modelers alike. A good example is smoke visibility. Unlike temperature or species concentration, smoke visibility is not a local quantity but rather depends on the viewpoint of the eye and the depth of field. Advanced simulators and games create the illusion of smoke or fog in ways that are not unlike the techniques employed by fire models to handle thermal radiation. The visualization of smoke and fire by Smokeview is an example of the graphics hardware and software actually computing results rather than just drawing pretty pictures. A common concern in the design of smoke control systems is whether or not building occupants will be able to see exit signs at various stages of a fire. FDS can predict the amount of soot is located at any given point, but that doesn't answer the question. The harder task is to compute on the fly within the visualization program what the occupant would see and not see. In this sense, Smokeview is not merely a *post-processor*, but rather an integral part of the analysis.

The purpose of Smokeview is to help one gain insight into the results of fire modeling simulations. Some areas of future work pertaining to the technical aspects of Smokeview include improving the visual modeling of smoke and fire and improving Smokeview's ability to handle larger cases. General strategies for improving Smokeview's ability to visualize cases and therefore to improve the understanding of computed fire flow are discussed in more detail in the Smokeview Technical Guide [2].

Bibliography

[1] G.P. Forney. *Smokeview, A Tool for Visualizing Fire Dynamics Simulation Data, Volume I: User's Guide*. National Institute of Standards and Technology, Gaithersburg, Maryland, USA, sixth edition, May 2013. i

[2] G.P. Forney. *Smokeview, A Tool for Visualizing Fire Dynamics Simulation Data, Volume II: Technical Reference Guide*. National Institute of Standards and Technology, Gaithersburg, Maryland, USA, sixth edition, May 2013. i, 137

[3] G.P. Forney. *Smokeview, A Tool for Visualizing Fire Dynamics Simulation Data, Volume III: Verification Guide*. National Institute of Standards and Technology, Gaithersburg, Maryland, USA, sixth edition, May 2013. i

[4] K. McGrattan, S. Hostikka, R. McDermott, J. Floyd, C. Weinschenk, and K. Overholt. *Fire Dynamics Simulator, User's Guide*. National Institute of Standards and Technology, Gaithersburg, Maryland, USA, and VTT Technical Research Centre of Finland, Espoo, Finland, sixth edition, September 2013. i, 3, 9, 13, 21, 35, 72, 117, 145

[5] K. McGrattan, S. Hostikka, R. McDermott, J. Floyd, C. Weinschenk, and K. Overholt. *Fire Dynamics Simulator, Technical Reference Guide, Volume 1: Mathematical Model*. National Institute of Standards and Technology, Gaithersburg, Maryland, USA, and VTT Technical Research Centre of Finland, Espoo, Finland, sixth edition, September 2013. i

[6] K. McGrattan, S. Hostikka, R. McDermott, J. Floyd, C. Weinschenk, and K. Overholt. *Fire Dynamics Simulator, Technical Reference Guide*. National Institute of Standards and Technology, Gaithersburg, Maryland, USA, and VTT Technical Research Centre of Finland, Espoo, Finland, sixth edition, September 2013. Vol. 1: Mathematical Model; Vol. 2: Verification Guide; Vol. 3: Validation Guide; Vol. 4: Software Quality Assurance. 3

[7] R. D. Peacock, G. P. Forney, and P. A. Reneke. CFAST – Consolidated Model of Fire Growth and Smoke Transport (Version 6): Technical Reference Guide. Special Publication 1026, National Institute of Standards and Technology, Gaithersburg, Maryland, July 2011. 3, 43

[8] D. Madrzykowski and R.L. Vettori. Simulation of the Dynamics of the Fire at 3146 Cherry Road NE, Washington, DC May 30, 1999. Technical Report NISTIR 6510, Gaithersburg, Maryland, April 2000. URL: http://fire.nist.gov/6510. 3

[9] D. Madrzykowski, G.P. Forney, and W.D. Walton. Simulation of the Dynamics of a Fire in a Two-Story Duplex, Iowa, December 22, 1999. NISTIR 6854, National Institute of Standards and Technology, Gaithersburg, Maryland, January 2002. 3

[10] R.L. Vettori, D. Madrzykowski, and W.D. Walton. Simulation of the Dynamics of a Fire in a One-Story Restaurant – Texas, February 14, 2000. Technical Report NISTIR 6923, Gaithersburg, Maryland, October 2002. 3

[11] R.G. Rehm, W.M. Pitts, Baum H.R., Evans D.D., K. Prasad, K.B. McGrattan, and G.P. Forney. Initial Model for Fires in the World Trade Center Towers. Technical Report NISTIR 6879, Gaithersburg, Maryland, May 2002. 3

[12] Brian W. Kernighan, Dennis Ritchie, and Dennis M. Ritchie. *C Programming Language (2nd Edition)*. Prentice Hall PTR, March 1988. 3

[13] T. M. Ellis, Ivor R. PHilips, and Thomas M. Lahey. *Fortran 90 Programming*. Addison-Wesley, 1994. 3

[14] Dave Shreiner, Graham Sellers, John M. Kessenich, and Bill M. Licea-Kane. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.3*. Addison-Wesley Professional, 8th edition, 2013. 3

[15] Mark J. Kilgard. *OpenGL Programming for the X Window System*. Addison-Wesley Developers Press, Reading, Massachussets, 1996. 3

[16] Thomas Boutell. *CGI Programming in C & Perl*. Addison-Wesley Publishing Co., Reading, Massachussets, 1996. 3

[17] Thomas Boutell. GD version 2.0.7, http://www.boutell.com/gd/, November 2002. 3

[18] Guy Eric Schalnat, Andreas Dilger, and Glenn Randers-Pehrson. libpng version 1.2.5, http://www.libpng.org/pub/png/, November 2002. 3

[19] JPEG version 6b, http://www.ijg.org/. 3

[20] Jean loup Gailly and Mark Adler. zlib version 1.1.4, http://zlib.net/, November 2002. 3

[21] Paul Rademacher. GLUI version 2.1, http://www.cs.unc.edu/ rademach/glui/. 3

[22] Tomas Akenine-Moller and Eric Haines. *Real-Time Rendering*. A K Peters, Ltd., Natick, Massachusetts, 2nd edition, 2002. 7

[23] D.A. Purser. *SFPE Handbook of Fire Protection Engineering*, chapter Combustion Toxicity. Springer, New York, 5th edition, 2016. 22

[24] Pamela P. Walatka and Pieter G. Buning. PLOT3D User's Manual, version 3.5. NASA Technical Memorandum 101067, NASA, 1989. 38

Part IV

Appendices

Appendix A

Command Line Options

Smokeview may be run from a command shell. Various command line options are available altering Smokeview's startup behavior such as creating a configuration file, using stereo, using the demo mode or running a script. To obtain a list of command line options, type:

```
smokeview -help
```

without any arguments which results in output similar to:

```
Smokeview  SMV6.7.15-0-g4fa68dd-release - Aug 21 2020
Visualize fire/smoke flow simulations.

Usage: smokeview_linux_64 [options] casenamewhere

 casename       - project id (file names without the extension)
 -bindir dir    - specify location of smokeview bin directory
 -ini           - output smokeview parameter values to smokeview.ini
 -runscript     - run the script file casename.ssf
  -help      - display help summary
  -help_all  - display all help info
  -version   - display version information

Other options:
 -build         - show directives used in this build of Smokeview
 -convert_ini case1.ini case2.ini - update case1.ini to the current format
                  and save results into case2.ini
 -demo          - use demonstrator mode of Smokeview
 -fast          - assume slice files exist in order to reduce startup time
 -fed           - pre-calculate all FED slice files
 -geominfo      - output information about geometry triangles
 -html          - output html version of smokeview scene
 -info            generate casename.slcf and casename.viewpoint files containing
    slice file and viewpiont info
 -lang xx       - where xx is de, es, fr, it for German, Spanish, French or Italian
 -ng_ini        - non-graphics version of -ini.
 -scriptrenderdir dir - directory containing script rendered images
                  (override directory specified by RENDERDIR script keyword)
 -setup         - only show geometry
 -script scriptfile - run the script file scriptfile
 -htmlscript scriptfile - run the script file scriptfile without using the video card
 -runhtmlscript - run the script file casename.ssf without using the video card
```

```
    the -htmlscript and -runhtmlscript keywords are used to generate JSON files
-sizes         - output files sizes then exit
-skipframe n   - render every n frames
-smoke3d       - only show 3d smoke
-startframe n  - start rendering at frame n
-stereo        - activate stereo mode
-update_bounds - calculate boundary file bounds and save to casename.binfo
-update_slice  - calculate slice file parameters
-update        - equivalent to -update_bounds and -update_slice
-update_ini case.ini - update case.ini to the current format
-volrender     - generate images of volume rendered smoke and fire
 -md5       - display an md5 hash when -version is invoked
 -sha1      - display a sha1 hash when -version is invoked
 -sha256    - display a sha256 hash when -version is invoked
 -hash_all  - display all hashes when -version option is invoked
 -hash_none - do not display any hashes  when -version is invoked
```

Appendix B

Menus

The user interacts with Smokeview using menus, dialog boxes and the keyboard. This appendix gives a brief overview of menus and dialogs. Appendix C documents keyboard shortcuts.

Menus are accessed by clicking the mouse anywhere in the scene with the right mouse button. The top level menu contains the *Load/Unload* menu for loading and unloading data files , the *Show/Hide* menu for showing and hiding data files previously loaded and other scene elements, the *Options* menu for setting options and performing actions such as rendering images of the scene or setting data units, the *Dialogs* menu for opening dialog boxes, the *Help* menu and *Quit* menu items.

B.1 Load/Unload

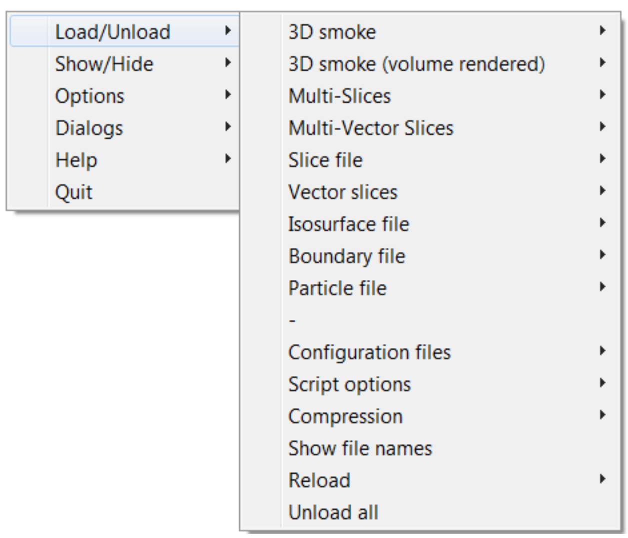


Figure B.1: Load/Unload Menu.

The Load/Unload menu, illustrated in Fig. B.1, is used to load or unload data files generated by FDS, CFAST or any fire model outputting data files using file formats documented in this or the FDS user's guide[4].

A menu item for each file generated by the fire model is present under *Load/Unload*. Selecting one of these menu entries causes the corresponding data file to load and be displayed. The data may be unloaded by selecting an *Unload* menu item appearing under the file list. Selecting *Unload All* unloads all files. To hide a data file, select the *Show/Hide* menu option corresponding to the file type to be hidden.

The character "*" occurring before a file name in the menu entry indicates that the file is loaded. If a file is loaded but not visible, use the appropriate *Show/Hide* option to make it visible.

The following is a list of file types visualized by Smokeview.

3D Smoke File (`.s3d`) This menu item allows one to load soot opacity and hrrpuv (heat release per unit volume) files. Smokeview uses the information contained in these files to visualize smoke realistically .

Multi-Slice File (`.sf`) This menu item allows one to load all slices occurring in one plane (within a grid cell) simultaneously. It also gives the option to unload the currently loaded multi-slices.

Multi-Vector Slice File (`.sf`) This menu item allows one to load all vector slices occurring in one plane (within a grid cell) simultaneously. It also gives the option to unload the currently loaded multi-slices.

Slice File (`.sf`) This menu item gives the name and location of all available slice files and also the option to unload the currently loaded slice files.

Vector Slice File (`.sf`) This menu item gives the name of all slice files that have one or more associated U, V and/or W velocity slice files. These slice files must be defined for the same region (or slice) in the simulation.

Isosurface File (`.iso`) This menu item gives the name of all isosurface files and also the option to unload the currently loaded isosurface file.

Boundary File (`.bf`) This menu item gives the name of all boundary files and also the option to unload the currently loaded boundary file.

Particle File (`.part`) This menu item gives the name of all particle file and also the option to unload the currently loaded particle file.

Plot3D File (`.q`) This menu item gives the name of all Plot3D files and also the option to unload the currently loaded Plot3D file.

Configuration Files (`.ini`) The INI or preference file contains configuration parameters that may be used to customize Smokeview's appearance and behavior. This menu item allows one to create (or overwrite) a preference file named either `smokeview.ini` or `casename.ini`. A preference file contains parameter settings for defining how Smokeview visualizes data. This file may be edited and re-read while Smokeview is running.

Compression 3D smoke and boundary files may be compressed using this menu item.

Script options Smokeview scripts may be recorded or run using this menu.

Show File Names Load and Unload menus by default are specified using the location and type of visual to be displayed. This menu item adds file names to the Load and Unload menus.

Reload This menu item allows one to reload files at immediately or at intervals of 1, 5 or 10 minutes. The `u` key may used to reload files from the keyboard. This is useful when using Smokeview to display a case that is currently running in FDS.

Unload All This option causes all data files to be unloaded.

B.2 Show/Hide

The *Show/Hide* menu allows one to show or hide various parts of the simulation. These menu items only appear if they pertain to the simulation. For example the *Particles* menu only appears if a particle file has been loaded. The "`*`" character is used to indicate that the visualization feature corresponding to that menu item is set or active.

B.3 Options

The option menu allows one to specify display units, rotation method, maximum frame rate, to render images, create tours and set font size.

Units Select alternate units for quantities such as temperature, velocity, distance *etc.*

Rotation Several rotation methods may be used to rotate the scene. The "`e`" keyboard shortcut may be used to toggle between rotation methods.

- The *Eye Centered* method allows one to rotate the scene relative to the observer's point of view or *eye*. Eye centered views make it easier to move around within the scene as in modern computer games.
- The *World Centered* rotation method allows one to rotate the scene relative to the scene's center.
- The *World Centered/level* rotation method is the same as *World Centered* but with level rotations.

Max Frame Rate This max frame rate option controls the rate at which image frames are displayed. The sub-menus allow one to specify a maximum frame rate. The actual frame rate may be slower if the scene is complex and the graphics card is unable to draw the scene sufficiently fast. The *unlimited* menu item allows one to display frames as rapidly as the graphics hardware permits. The *Real Time* menu item allows one to draw frames so that the simulation time matches real time. The *step* menu item allows one to step through the simulation one time step at a time. This menu item may be used in concert with the *Render* menu item described below to create images at the desired time and view orientation for inclusion into reports. This is how figures were generated in this report.

Render The *Render* menu allows one to create PNG or JPEG image files of the currently displayed scene.

The *Render* menus allow one to specify an integer indicating the number of frames between rendered images. This allows one to generate images encompassing the entire time duration of the simulation which in turn can be converted into movie files (`mpeg, mov, avi`, etc) using software available on the internet. Rendering may be stopped by selecting *Cancel*.

Tours The keyboard shortcut for the render option is `r`. The *Tour* menu allows one to show and hide available tours.

Font Size This option allows one to display text in either a normal or a large font.

B.4 Dialogs

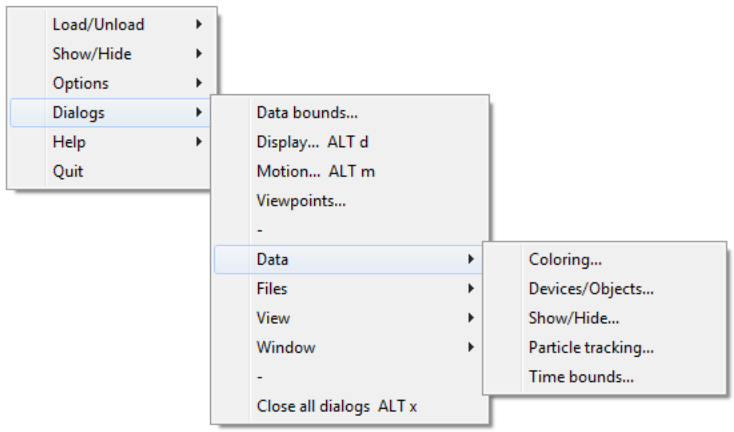


Figure B.2: Dialogs Menu.

The *Dialogs* menu, illustrated in Fig. B.2, allows one to select dialog boxes used for setting various Smokeview features and configuration parameters. Commonly used dialog box entries appear first. These are *Data bounds* for setting data file bounds (and other data file characteristics), *Display* for setting various parameters that control how the scene appears, *Motion* for controlling scene movement through rotation and translation and *Viewpoints* for defining and setting viewpoints. Less commonly used dialog box entries appear at the bottom of the menu under the sub-menus *Data*, *Files*, *View* and *Window*. Dialog menu entries and a short description for each are listed below.

Data bounds Dialog box for setting min/max data bounds, min/max clipping data bounds and other parameter settings related to data files.

Display Dialog box for setting various display parameters.

Motion Dialog box for rotating and translating the scene.

Viewpoints Dialog box for saving and setting viewpoints.

Dialog menu entries along with the sub-menu of *Dialogs* where they appear (enclosed in parenthesis) are listed below.

Auto load (*Files*) Dialog for automatically loading data files.

Clip scene (*View*) Dialog for clipping data and/or geometry.

Compress (*Files*) Dialog for compressing FDS generated data files using the program Smokezip.

Coloring (*Data*) Dialog for setting data coloring characteristics. One may select the colorbar used to color data, how the colorbar is displayed (continuous, stepped or discrete lines, color opacity level *etc*.

Configuration (*Files*) Dialog box for saving or loading configuration files.

Device/Objects (*Data*) Dialog box for scaling Smokeview objects and showing data values associated with FDS devices.

Edit colorbar (*View*) Dialog box for creating new and editing existing colorbars.

Examine geometry (*View*) Dialog box for examining FDS blockages.

Fonts (*Window*) Dialog box for selecting the font size used to display text. The choices are limited to small, large or scaled.

Labels (*Window*) Dialog box for defining text labels and controlling when and where they are placed.

Particle tracking (*Data*) Dialog box for releasing and viewing particles within the scene.

Render images (*Files*) Dialog box for creating images of a smokeview scene. In addition, if the program `ffmpeg` is present in the user path, one may create movies.

Scaling (*Window*) Dialog box for scaling the X, Y and or Z dimensions of a Smokeview scene.

Scripts (*Files*) Dialog box for recording or running a smokeview script.

Show/Hide (*Data*) Dialog box for showing or hiding data.

Slice motion (*Data*) Dialog box for controlling the position and/or orientation of a 3D slice file.

Stereo parameters (*View*) Dialog box for specifying the method used to display a Smokeview scene in stereo.

Time bounds (*Data*) Dialog box for specifying the min/max time bounds for loading data.

Tours (*View*) Dialog box for creating new tours and editing existing ones.

User ticks (*Display*) Dialog box for specifying the placement of tick marks to appear along the edges of a Smokeview scene.

Window properties (*Window*) Dialog box for specifying the characteristics (screen size, projection method) of the window containing the Smokeview scene.

Appendix C

Keyboard Shortcuts

Many menu commands have equivalent keyboard shortcuts. These shortcuts are described here and are also briefly described under the *Help* menu item from within Smokeview.

C.1 alphanumeric shortcuts

`a`	Lengthen slice and PLOT3D vector lengths (use ALT a to shorten vector lengths).
`a`	When in *eye centered* movement mode, slide to the left.
`A`	Toggle axis label smoothing on and off.
`b`	Toggle visibility of boundary files
`B`	Switch between visibility of vents/blockages and boundary files.
`c`	Toggle 2D contour display between banded, continuously shaded and line contours.
`c`	Advance highlighted zone fire modeling compartment.
`C`	Toggle 3D smoke culling. If the scene is a zone fire modeling simulation, toggle zone fire modeling compartment highlighting.
`d`	Activates `CTRL` key when moving the scene with the mouse. Pressing and releasing the `d` key then moving the mouse causes the scene to go up and down until the mouse button is released.
`D`	Slide right when in *eye centered* movement mode.
`e,E`	Toggle how the scene is manipulated. In *eye view* scene motion is relative to the observer. In *world view* scene motion is relative to the scene center.
`f`	Activates the `ALT` key when moving the scene with the mouse. Pressing and releasing the `f` key then moving the mouse causes the scene to go in and out until the mouse button is released.
`F`	Toggle algorithm for hiding blockage overlaps.
`g`	Toggle the grid visibility. When the grid display option is active, the `x`, `y` and `z` keys may be used to show or hide the grid perpendicular to the x, y and z axes respectively.
`G`	Toggle the use of the GPU (if present).
`h`	Toggle window indentation, for use with window capturing.
`H`	Toggle the visibility of slice and vector slice files.
`i,I`	Toggle Plot3D iso-contour visibility.

`j,J` Increase size of smokeview objects (use ALT j to decrease size).

`k,K` Toggle timebar visibility.

`l` Reload all currently loaded files.

`L` Unload most recently loaded slice file.

`m,M` Switch between meshes in multiple mesh cases.

`o` Switch between outline viewing modes. The modes are

1. outline of the current mesh
2. outline of the entire case
3. no outline

This option may be used with the *m* key to highlight all meshes in sequence of a case.

`O` Toggle the blockage view state between 1) defined in input file and 2) outline only

`p` Cycle through particle file types.

`P` Cycle through Plot3D file types.

`q` Switch between blockage views. These views are blocks that are aligned on grid lines, blocks as specified by the user (in the FDS input file) and blocks as generated by a CAD (computer aided drawing) package.

Also switch between vent views. When a circular vent is specified, the *q* key will switch between how the vent is specified by the user, a circle and how it is represented by FDS, a series of grid cell faces.

`Q` Toggle texture visibility.

`r,R` Render the current scene as a JPEG or a PNG file which can be viewed in a web browser or inserted into a word processing document. If `R` is selected then the scene is rendered in with double the screen resolution (or greater if the image multiplier menu is selected).

`s` Increment the number of vectors skipped. This is useful for making vector displays more readable when grids are finely meshed.

`s` Move backwards when in *eye centered* movement mode.

`S` Change stereo modes (left/right, red/blue, none, etc.)

`t` Toggle the time stepping mode. Time stepping mode allows one to step through the simulation one time step at a time.

`T` Toggle the time time bar time label between showing time as seconds and time as hour, minutes and seconds.

`u,U` Reload files from the keyboard. This is useful when using Smokeview to display a case that is currently running in FDS.

`v` Toggle vector visibility. This option is only active when there are U, V and/or W velocity components present in the Plot3D data set.

`V` Toggle visibility of volume rendered smoke.

`w` Toggle visibility of 3D node centered general slice. When in *eye centered* movement mode, move forward.

`W` Toggle between four clipping modes: 1) disabled, 2) blockages and data, 3) blockages and 4) data.

x,X y,Y, z,Z Toggle the visibility of the Plot3D data planes perpendicular to the x, y and z axes respectively (parallel to the yz, xz and xy planes).

0 Reset a time dependent animation to the initial time.

1-9 Number of frames to skip when viewing an animation.

C.2 ALT shortcuts

ALT a Shorten slice and PLOT3D vector lengths (use 'a' to lengthen vector lengths).

ALT b Open the Bounds dialog box.

ALT c Open the Clipping dialog box.

ALT d, D Open the Display dialog box. On the OSX version of smokeview only ALT D can be used to open the Display dialog box as d and f are used for CTRL and ALT modifier keys respectively.

ALT e Open the Blockage info dialog box.

ALT g Open the Viewpoint dialog box.

ALT j Decrease size of smokeview objects (use 'j' to increase the size).

ALT m Open the Motion/View/Render dialog box.

ALT o Switch between various blockage view states. States are: 1) defined in input file, 2) defined in input file + outline, 3) solid, 4) outline only and 5) hidden

ALT r Toggle research mode. Research mode uses global min and max bounds for coloring data and turns off smoothing when displaying colorbar labels.

ALT R Display scene in 360° view mode - all view directions of the scene are displayed in a 1024×512 image.

ALT s Open the 3D Smoke dialog box.

ALT t Open the Edit Tours dialog box.

ALT u Toggle the option to draw a coarse portion of a 2D slice file within an embedded mesh.

ALT v Toggle the projection method used to visualize a scene. The two projection methods are size preserving and perspective.

ALT w Open the WUI dialog box.

ALT x Close all dialog boxes.

ALT z Open the Compress Files portion of the File/Bounds dialog box.

C.3 Special character short cuts

~ Level scene.

! Snap scene to nearest 45 degree rotation angle.

@ Toggle display of FDS values (as floating point numbers) when viewing vector centered slice files.

Save configuration settings to the casename.ini file.

$ Toggle trainer or demonstrator mode. When active, displays a dialog box that provides a simple set of controls for controlling the scene.

% Toggle single stepping mode. If activated, Smokeview will execute a script one command at a time.

^ When single stepping mode is activated, this key causes the next script command to be executed.

& Toggle line anti-aliasing. When active, this option draws line smoothly without *jaggies*.

* Hide all 3D slice planes (aligned with 3 coordinate axes and general slice planes).

= Toggle vertex selected in examine geometry dialog

, Toggle display of colorbar display (show vertical, horizontal or now colorbar).

. Toggle locking of left/right scene motion when using the SHIFT mouse to change the scene aperture.

(Toggle render clipping mode

[Turn on tour editing

] Turn off tour editing

< Increase vector point size

> Decrease vector point size

; Flip colorbar

/ Toggle parallel loading of particle files. The number of threads used to load particles files in parallel may be specified using the Settings portion of the particle bounds dialog box.

Left/Right Cursor When the *eyeview* mode is *eye centered* then these keys rotate the scene to the left or right otherwise they increment/decrement the Plot3D plane location displayed in the xz plane.

Up/Down Cursor Increment/decrement the Plot3D plane location displayed in the yz plane.

Page Up, Page Down Increment/decrement the Plot3D plane location displayed in the xy plane.

- Decrement Plot3D data planes, Plot3D iso-contour levels or time step displayed.

space bar Increment Plot3D data planes, Plot3D iso-contour levels or time step displayed.

Appendix D

File Formats and Extensions

D.1 FDS and Smokeview File Extensions

D.1.1 FDS file extensions

.bf File containing boundary file data.

.end File containing Endian information.

.fds File containing the FDS input file.

.iso File containing iso-surface data

.out File containing FDS output.

.prt File containing particle file data using FDS 4 and earlier.

.prt5 File containing particle file data using FDS 5 and later.

.q File containing Plot3D data.

.sf File containing slice file data.

.s3d File containing 3D smoke, HRRPUV data.

D.1.2 Smokeview file extensions

.bini File containing percentile and global data bounds for boundary files in referenced casename.smv.

.ini File containing Smokeview configuration settings.

.smv File containing Smokeview keyword data.

.ssf File containing a Smokeview script.

.svz File containing compressed boundary, slice or 3D smoke/fire data. The .svz extension is appended to the .bf, .sf or .s3d extension respectively.

.sz File containing sizing information for uncompressed data files. The .sz files contain information about each data frame used by Smokeview to allocate memory.

.szz File containing sizing information for compressed .svz files (files compressed with Smokezip with a .svz extension).

D.2 Smokeview Bound File Format (.bini files)

The first time a user views a boundary file, Smokeview computes data bounds by inputting all boundary file data of the same type. Smokeview records the bound computations result in a casename.bini file so that it does not need to be performed a second time. The .bini file is then used in subsequent Smokeview sessions for displaying boundary file data. The .bini file contains one or more B_BOUNDARY keywords .

B_BOUNDARY defines the global minimum and maximum and percentile minimum and maximum boundary data bounds used to convert boundary data values to color indices. The `B_BOUNDARY` keyword also has a parameter allowing one to specify the data type. The format is given by

```
B_BOUNDARY
 global_min percentile_min percentile_max global_max data_type
```

D.3 Smokeview Preference File Format (.ini files)

Smokeview uses preference files to set input parameters not settable by using menus or the keyboard and to save the *state* of a visualization. Smokeview looks for preference files in three locations in the following order:

1. a file named `smokeview.ini` in a global directory defined by the `SMOKEVIEWINI` environment variable. On the PC, the directory `C:\Program Files\ FDS\FDS5\bin\smokeview.ini` is the default location for this preference file. The `SMOKEVIEWINI` environment variable may be defined on the PC to specify the location of the smokeview.ini file. This step is performed automatically by the Smokeview installation program.

   This environment variable may be defined on a UNIX workstation by adding the line:

   ```
   setenv SMOKEVIEWINI dir
   ```

   to a `.login` or `.cshrc` start up file again where `dir` is the directory containing the global preference file. Changes to this `smokeview.ini` file apply to all cases visualized on the computer unless overridden by preference files named or located in directories named in steps 2. and 3.

2. a file named `smokeview.ini` in the directory containing the case being visualized. Changes to this `smokeview.ini` file apply to all cases in the current directory unless overridden by the `casename.ini` file contained in this directory.

3. a file named `casename.ini` in the directory containing the case being visualized where casename is the name of the case.

   The `smokeview.ini` file may be created by typing:

```
smokeview -ini
```

from the command line or by selecting the `smokeview.ini` menu item. The `casename.ini` preference file can be created *via* the menus or by copying a previously created `smokeview.ini` file.

Smokeview reads the global `smokeview.ini` file first (step 1. above), followed by the local `smokeview.ini` file (step 2. above), followed by the `casename.ini` file. The global `smokeview.ini` file is used to customize parameters for all Smokeview runs. The local `smokeview.ini` file is used to customize parameters for just those Smokeview runs contained in the local directory. The `casename.ini` file is used to customize parameters for only those Smokeview runs with the prefix casename.

All preference file parameters unless otherwise noted consist of a `KEYWORD` followed by a value, as in:

```
KEYWORD
value
```

D.3.1 Color and lighting

All colors are specified using a 3-tuple: r g b where r, g and b are the red, green and blue components of the color respectively. Each color component is a floating point number ranging from 0.0 to 1.0 where 0.0 is the darkest shade and 1.0 is the lightest shade. For example the 3-tuple 1.0 0.0 0.0 is bright red, 0.0 0.0 0.0 is black and 1.0 1.0 1.0 is white.

AMBIENTLIGHT Sets the color used for specifying ambient light. (default: `0.6 0.6 0.6`)

BACKGROUNDCOLOR Sets the color used to visualize the scene background. (default: `0.0 0.0 0.0`)

BLOCKCOLOR Sets the color used to visualize internal blockages. (default: `1.0 0.8 4.0`)

BOUNDCOLOR Sets the color used to visualize floors, walls and ceilings. (default: `0.5 0.5 0.2`)

COLORBAR Entries for the color palette in RGB (red, green, blue) format where each color component ranges from 0.0 to 1.0 . The default values (rounded to 2 digits) are specified with:

```
COLORBAR
12
0.00 0.00 1.00
0.00 0.28 0.96
0.00 0.54 0.84
0.00 0.76 0.65
0.00 0.91 0.41
0.00 0.99 0.14
0.14 0.99 0.00
0.41 0.91 0.00
0.65 0.76 0.00
0.84 0.54 0.00
0.96 0.28 0.00
1.00 0.00 0.00
```

COLOR2BAR Miscellaneous colors used by Smokeview. The default values are specified using:

```
COLOR2BAR
8
1.0 1.0 1.0 :white
1.0 1.0 0.0 :yellow
0.0 0.0 1.0 :blue
1.0 0.0 0.0 :red
0.0 1.0 0.0 :green
1.0 0.0 1.0 :magenta
0.0 1.0 1.0 :cyan
0.0 0.0 0.0 :black
```

where the 8 indicates the number of colors defined and the character string after the ":" are ignored.

COLORBAR_FLIP Specifies whether the colorbar is flipped (1) or not flipped (0) (default: `0`).

DIFFUSELIGHT Sets the color for specifying diffuse light (default: `0.5 0.5 0.5`).

FLIP Specifies whether to flip (1) or not to flip (0) the foreground and background colors. By default the background color is black and the foreground color is white. Setting FLIP to 1 has the effect of having a white background and black foreground. (default: `0`).

FOREGROUNDCOLOR Sets the color used to visualize the scene foreground (such as text labels). (default: `1.0 1.0 1.0`)

HEATOFFCOLOR Sets the color used to visualize heat detectors before they activate. (default: `1.0 0.0 0.0`)

HEATONCOLOR Sets the color used to visualize heat detectors after they activate. (default: `0.0 1.0 0.0`)

ISOCOLORS Colors and parameters used to display animated isocontours. Default:

```
ISOCOLORS
 10.000000 0.800000 : shininess, transparency
 0.700000 0.700000 0.700000 : specular
 3 : number of levels
 0.960000 0.000000 0.960000 0.800000 : red, green, blue, alpha (opaqueness)
 0.750000 0.800000 0.800000 0.800000
 0.000000 0.960000 0.280000 0.800000
```

SENSORCOLOR Sets the color used to visualize sensors. (default: `1.0 1.0 0.0`)

SETBW The parameter used to set whether color shades (0) or shades of gray (1) are to used for coloring contours and blockages. (default: `0`)

SPRINKOFFCOLOR Sets the color used to visualize sprinklers before they activate. (default: `1.0 0.0 0.0`)

SPRINKONCOLOR Sets the color used to visualize sprinklers after they activate. (default: `0.0 1.0 0.0`)

STATICPARTCOLOR Sets the color used to visualize static particles (particles displayed in frame 0). (default: `0.0 1.0 0.0`).

TIMEBARCOLOR Sets the color used to visualize the timebar. (default: `0.6 0.6 0.6`)

VENTCOLOR Sets the color used to visualize vents. (default: `1.0 0.0 1.0`)

D.3.2 Size

The parameters described in this section allow one to customize the size of various Smokeview scene elements.

ISOLINEWIDTH Defines the width in pixels of lines used to draw animated iso-surfaces in outline mode. (default: 2.0)

ISOPOINTSIZE Defines the size in pixels of iso-surface particles. (default: 4.0)

LINEWIDTH Defines the width of lines[1] in pixels. (default: 2.0)

PARTPOINTSIZE Defines the size in pixels of smoke or tracer particles. (default: 1.0)

PLOT3DLINEWIDTH Defines the width in pixels of lines used to draw Plot3D iso-surfaces in outline mode. (default: 2.0)

PLOT3DPOINTSIZE Defines the size in pixels of Plot3D iso-surface particles. (default: 4.0)

SENSORABSSIZE Defines the sensor size drawn by Smokeview using the same units as used to specify the grid coordinates. (default: 0.038)

SLICEOFFSET Defines an offset distance[2] animated slices are drawn from adjacent solid surfaces. (default: 0.10)

SMOOTHLINES Specifies whether lines should be drawn (1) or not drawn (0) using anti-aliasing (default: `1`).

SPRINKLERABSSIZE Defines the sprinkler size drawn by Smokeview using the same units as used to specify the grid coordinates. (default: 0.076)

STREAKLINEWIDTH Defines the width of a streak line. (default: 1.0)

VECLENGTH Defines the length of Plot3D vectors. A vector length of 1.0 fills one grid cell. Vector lengths may also be changed from within Smokeview by depressing the "`a`" key. (default: 4.0)

VECTORPOINTSIZE Defines the size in pixels of the point that is drawn at the end of a Plot3D vector. (default: 3.0)

VENTLINEWIDTH Defines the width of lines used to draw vents in pixels. (default: 2.0)

VENTOFFSET Defines a distance used to offset vents drawn from adjacent surfaces. (default: 0.10 (units of fraction of a grid cell width))

WINDOWHEIGHT Defines the initial window height in pixels. (default: 480)

WINDOWWIDTH Defines the initial window width in pixels. (default: 640)

WINDOWOFFSET Defines a margin offset around the Smokeview scene for use when capturing images to video. (default: 45)

D.3.3 Time and data bounds

This section describes parameters used by Smokeview to 1) modify the time intervals used to load data (keywords beginning with T_), 2) eliminate or chop data from being displayed (beginning with C_) and 3) override the minimum and maximum data values (keywords beginning with V_) used to convert data to color values. By default, Smokeview reads in data for all time values and does not override minimum and maximum data values. Each time and data bound keyword (except for V_PLOT3D) has the format:

```
KEYWORD
minflag minvalue maxflag maxvalue
```

where `minflag` can be either 0 or 1. If it is 1 then, the subsequent number, `minvalue` is used by Smokeview to scale the data otherwise if `minflag` is 0 then `minvalue` is ignored. The next two parameters `maxflag` and `maxvalue` are defined similarly. The `V_PLOT3D` keyword contains data bound entries for each variable in the Plot3D data file. If a Plot3D *speed* variable was constructed by Smokeview then the `V_PLOT3D` keyword will contain six entries instead of five.

[1] Except lines used to draw vents

[2] distance is relative to the maximum grid cell width

C_PARTICLES Defines the minimum and maximum values used to discard particle data in a visualization. To drop particle data below 70°C and above 200°C use:

```
C_PARTICLES 1 70. 1 200.
```

C_PLOT3D Defines the minimum and maximum data values used to drop or chop Plot3D. To cause Smokeview to set the minimum and maximum chop values for the first Plot3D quantity (usually temperature) to 100 and 300 use:

```
C_PLOT3D 5
1 1 100.0 1 300.0
2 0 1.0 0 0.0
3 0 1.0 0 0.0
4 0 1.0 0 0.0
5 0 1.0 0 0.0
```

The integer "`1`" occurring before the "`100`" or "`300`" causes Smokeview to use the next number as a minimum or maximum chop value respectively.

C_SLICE Defines the minimum and maximum values used to discard slice file data in a visualization. To drop slice data below 70°C and above 200°C use:

```
C_SLICE 1 70. 1 200.
```

V_BOUNDARY Defines the minimum and maximum data bounds used to convert boundary data values to color indices. (default: `0 1.0 0 0.0`)

The `V_BOUNDARY` keyword has an optional parameter allowing one to specify to which type of data the bounds should apply. For example, to specify boundary file bounds for temperature (30.0 °C, 600.0 °C) use:

```
V_BOUNDARY
1 30.000000 1 600.000000 TEMP
```

where `TEMP` is the Smokeview colorbar labels displayed when showing the boundary file.

These suffixes are added automatically when the File Bounds dialog box is used to set data bounds.

V_PARTICLES Defines the minimum and maximum data bounds used to convert particle data values to color indices. (default: `0 1.0 0 0.0`)

V_PLOT3D Defines the minimum and maximum data bounds used to convert Plot3D data values to color indices. The default values are given by:

```
V_PLOT3D
5
1 0 1.0 0 0.0
2 0 1.0 0 0.0
3 0 1.0 0 0.0
4 0 1.0 0 0.0
5 0 1.0 0 0.0
```

where the initial 5 indicates the number of subsequent entries, the first integer on each line indicates the Plot3D data variable being specified and all other parameters on each line are defined as above.

To cause Smokeview to set the minimum and maximum data values to for the first quantity (usually temperature) to 20 and 600 use:

```
V_PLOT3D
5
1 1 20.0 1 600.0
2 0 1.0 0 0.0
3 0 1.0 0 0.0
4 0 1.0 0 0.0
5 0 1.0 0 0.0
```

The integer "1" occurring before the "20" or "600" causes Smokeview to use the next number as a minimum or maximum value respectively otherwise if "0" is specified then Smokeview ignores the subsequent min/max value.

In addition to 0 and 1, the V_PLOT3D keyword may use 2 as a bound indicator. In the above example, if 2 rather than 1 is used to define Plot3D bounds, then Smokeview does not draw contour[3] levels smaller than 20 or contours greater than 600. The Plot3D bound line 1 2 360.0 1 420.0 indicates that temperatures below 360 °C are not drawn and that temperatures above 420 °C are drawn with the *highest* color (black in black and white mode, red in color mode). This bound line could also be implemented with the File/Bounds dialog box as illustrated in Fig. 5.3 resulting in a contour plot as illustrated in Fig. 5.2.

V_SLICE Defines the minimum and maximum data bounds used to convert slice data values to color indices. (default: 0 1.0 0 0.0)

The V_SLICE keyword has an optional parameter allowing one to specify to which type of data the bounds should apply. For example, to specify separate slice file bounds for temperature (30.0 °C, 600.0 °C) and the U component of velocity (-1.0 m/s and 1.0 m/s) use:

```
V_SLICE
1 30.000000 1 600.000000 TEMP
V_SLICE
1 -1.0 1 1.0 U-VEL
```

where TEMP and U-VEL are the Smokeview colorbar labels displayed when showing the slice file.

These suffixes are added automatically when the File/Bounds dialog box is used to set data bounds.

V_TARGET

D.3.4 Data loading

The keywords in this section may be used to reduce the memory required to visualize FDS data. Keywords exist for limiting particles and frames. Other keywords exist for compressing particle data and skipping particles and frames.

BOUNDZIPSTEP Defines the number of intervals or steps between boundary file frames when compressed by Smokezip. (default: 1)

[3] This is true for *stepped* or discrete contours. If *continuous* contours are drawn, then "2" and "1" have the same effect.

ISOZIPSTEP Defines the number of intervals or steps between isosurface file frames when compressed by Smokezip. (default: 1)

NOPART Indicates that a particle file should not (1) or should (0) be loaded when Smokeview starts up. This option is used when one wants to look at other files besides the particle file. (default: 1)

SLICEDATAOUT When set to 1 will output data corresponding to any loaded slice files whenever the scene is rendered.

SLICEZIPSTEP Specifies the number of intervals or steps between slice frames when compressed by Smokezip. (default: 1)

SMOKE3DZIPSTEP

D.3.5 View

The keywords in this section define how a scene is viewed. Keywords exist for showing or hiding various scene elements and for modifying how various scene elements appear.

APERTURE Specifies the viewing angle used to display a Smokeview scene. Viewing angles of 30, 45, 60, 75 and 90 degrees are displayed when APERTURE has the value of 0, 1, 2, 3 and 4 respectively. (default: 2)

AXISSMOOTH Specifies whether axis numerical labels should be smoothed (`AXISSMOOTH set to 1`) or not smoothed (`AXISSMOOTH` set to 0). (default: 1)

BLOCKLOCATION Specifies the location or method used to draw blockages. Blockages are drawn either snapped to the nearest grid (5), drawn at locations as specified in the FDS input file (6) or drawn as specified in a compatible CAD package (7)[4]. (default: 5)

CLIP Specifies the near and far clipping plane locations. Dimensions are relative to the longest side of an FDS scene. (default: 0.001 3.000)

COLORBAND Specifies the width in pixels of the black colorband created when the colorbar is selected by the mouse. (default: 1)

COMPRESSAUTO Specifies that files that are to be auto-loaded by Smokeview (and no other files) should be compressed by Smokezip.

CULLFACES Hide (1) or show (0) the back side of various surfaces.

EYEVIEW Specifies whether the scene should be rotated relative to the observer (`EYEVIEW` set to 1) or the scene center (`EYEVIEW`=0). (default: 0)

EYEX, EYEY, EYEZ The parameters `EYEX, EYEY, EYEZ` specify the x, y and z location respectively of the viewing origin (where your eyes are). (default: `0.5 -0.9 1.5`)

FONTSIZE Specifies whether small (0) or large (1) fonts should be used to display text labels. (default: 0)

FRAMERATEVALUE Specifies the maximum rate (frames per second) that Smokeview should display frames. This value is an upper bound. Smokeview will not display frames faster than this rate but may display frames at a slower rate if the scene to be visualized is complex. (default: 1000000 (essentially unlimited))

[4] There are various third party tools that have been developed to help process obstruction data for FDS. See the FDS/Smokeview website, `http://pages.nist.gov/fds/`, for details.

ISOTRANS Specifies transparency state for iso-surfaces. The choices are all iso-surface levels transparent (ALL_TRANSPARENT=1), minimum iso-surface level solid (MIN_SOLID=2), maximum iso-surface level solid (MAX_SOLID=3) and all iso-surface levels transparent (ALL_TRANSPARENT=4) (default: 3)

LABELSTARTUPVIEW Specifies a viewpoint to be applied when smokeview first starts up. Any viewpoint defined by the VIEWPOINT5 ini keyword or the predefined viewpoints external or internal may used to with the LABELSTARTUPVIEW keyword. (default: external)

MSCALE Specifies how dimensions along the X, Y and/or Z axes should be scaled. (default: 1.0 1.0 1.0)

PROJECTION Specifies whether a perspective (0) or orthographic (1) projection is used to draw Smokeview scenes. (default: 0)

P3CONT2D The parameter `P3CONT2D` may be set to 0, 1 or 2. If P3CONT2D is set to 0 then Plot3D color contours are drawn by coloring each node and letting OpenGL interpolate colors between nodes. If P3CONT2D is set to 1 then discrete or stepped shaded contours are drawn. If P3CONT2D is set to 2 then contour lines are drawn. (default: 1)

P3DSURFACETYPE Specifies how Plot3D isosurfaces should be drawn. If P3DSURFACETYPE is set to 1 then Plot3D isosurfaces are drawn using shaded triangles. If P3DSURFACETYPE is set to 2 or 3 then Plot3D isosurfaces are drawn using triangle outlines and points respectively. (default: 1)

P3DSURFACESMOOTH When drawing Plot3D isosurfaces using shaded triangles, this option specifies whether the vertex normals should be averaged (P3DSURFACESMOOTH set to 1) resulting in smooth isosurfaces or not averaged resulting in isosurfaces that have sharp edges (P3DSURFACESMOOTH set to 0). (default: 1)

RENDERFILETYPE Specifies whether PNG (RENDERFILETYPE set to 0) or JPEG (RENDERFILETYPE set to 1) should be used to render images. (default: 1)

RENDEROPTION Records the option used to render images.

SENSORRELSIZE Specifies a scaling factor that is applied when drawing all sensors. (default: 1.0)

SHOWAXISLABELS Specifies whether axis labels should be drawn (1) or not drawn (0) drawn. (default: 0)

SHOWBLOCKLABEL Specifies whether a label identifying the active mesh should be drawn (1) or not drawn (0). (default: 1)

SHOWBLOCKS Specifies how a blockage should be drawn. A value of 0, 1 or 2 indicates that the blockages are invisible, drawn normally or drawn as outlines respectively. (default: 1)

SHOWCADANDGRID

SHOWCEILING Specifies whether the ceiling (upper bounding surface) should be drawn (1) or not drawn (0). (default: 0)

SHOWCOLORBARS Specifies whether the colorbars should be drawn (1) or not drawn (0). (default: 1)

SHOWEXTREMEDATA Specifies whether data exceeding the maximum colorbar label value or less than the minimum colorbar label value should be colored with a different color (black). If SHOWEXTREMEDATA is set to 1 then extreme data is colored black, if SHOWEXTREMEDATA is set to 0 then extreme data is colored as indicated by the maximum and minimum region of the colorbar. (default: 0). .

SHOWFLOOR Specifies whether the floor (lower bounding surface) should be drawn (1) or not drawn (0). (default: 1)

SHOWFRAME Specifies whether the frame surrounding the scene should be drawn (1) or not drawn (0). (default: 1)

SHOWFRAMELABEL Specifies whether the frame number should be drawn (1) or not drawn (0). (default: 1)

SHOWFRAMERATE Specifies whether the frame rate label should be drawn (1) or not drawn (0). (default: 0)

SHOWGRIDLOC Specifies where grid location should be drawn (1) or not drawn (0). (default: 0)

SHOWHRRCUTOFF Specifies whether the HRRPUV cutoff label should be (1) or should not be (0) displayed. (default: 0)

SHOWISO Specifies how an isosurface should be drawn: hidden (0), solid (1), outline (2) or with points (3). (default: 1)

SHOWISONORMALS Specifies whether iso-surface normals are drawn (1) or not drawn (0). (default: 0)

SHOWIGNITION When drawing a temperature boundary file, this option specifies whether ignited materials (regions exceeding the materials ignition temperature) should be drawn (1) or not drawn (0). A second parameter specifies whether only the ignited regions should be drawn (1) or both the ignited regions and other regions should be drawn (0). (default: 0 0)

SHOWLABELS Specifies whether labels should be drawn (1) or not drawn (0). Labels are specified using the *LABEL* keyword described in subsection 14. (default: 0)

SHOWMEMLOAD Specifies (when run on a PC) whether a label giving the memory used should be drawn (1) or not drawn (0). (default: 0)

SHOWOPENVENTS Specifies that open vents should be drawn (1) or not drawn (0). (default: 0)

SHOWDUMMYVENTS Specifies that dummy vents (vents created by FDS) should be drawn (1) or not drawn (0). (default: 0)

SHOWSENSORS Specifies whether sensors should be drawn (1) or not drawn (0). A second parameter specifies whether the sensor's orientation or normal vector should be drawn (1) or not drawn (0). (default: 1 0).

SHOWSLICEINOBST Specifies whether a slice file should be drawn (1) inside a blockage or not drawn (0) inside a blockage. Normally a slice file is not drawn inside a blockage but one would want to draw a slice file inside a blockage if the blockage disappears over the duration of a run. (default: 0).

SHOWSMOKEPART Specifies whether smoke or trace particles should be drawn (1) or not drawn (0). (default: 1)

SHOWSPRINKPART Specifies whether sprinkler droplet particles (if present in the particle file) should be drawn (1) or not drawn (0). (default: 1)

SHOWSTREAK Specify parameters that define streak properties. This keyword has four integer parameters with format:

```
SHOWSTREAK
streak5show,streak5step,showstreakhead,streakindex
```

The `streak5show` parameter may be 0 or 1 and indicates whether a streak is not (0) or is (1) shown. The `streak5step` parameter indicates number of streaks skipped or not displayed. The `showstreakhead` parameter may be 0 or 1 and indicates whether a streak head is not (0) or is (1) shown. The `streakindex` parameter indicates the length of the streak.

SHOWTERRAIN If terrain is present, specifies that terrain should be visualized as a *warped* sheet rather than as a set of FDS blockages.

SHOWTICKS Specifies whether labels should be drawn (1) or not drawn (0). Ticks are specified using the *TICK* keyword described in subsection 14. (default: 0)

SHOWTIMEBAR Specifies whether the timebar should be drawn (1) or not drawn (0). (default: 1)

SHOWTIMELABEL Specifies whether the time label should be drawn (1) or not drawn (0). (default: 1)

SHOWTRANSPARENTVENTS Specifies whether vents specified as being invisible should be shown (1) or not shown (0). (default: 0)..

SHOWHMSTIMELABEL Specifies whether the time label should be drawn (1) or not drawn (0) using the format "h:m:s" where "h" is hours, "m" is minutes and "s" is seconds. (default: 0)

SHOWTITLE Specifies whether the title should be drawn (1) or not drawn (0). (default: 1)

SHOWVENTS Specifies whether vents should be drawn (1) or not drawn (0). (default: 1)

SHOWALLTEXTURES If wall textures are defined in the input .smv file, this option specifies whether to draw (1) or not to draw (0) wall textures. (default: 0)

SHOWWALLS Specifies whether the four walls (four vertical bounding surfaces) should be drawn (1) or not drawn (0). (default: 1)

SMOKERTHICK When the GPU is used to visualize smoke, this parameter specifies a relative thickness scaling factor.

STARTUPVIEW Used by Smokeview to record the view to be displayed at startup (as defined in a previous Smokeview run).

SURFINC Smokeview allows one to display two Plot3D isosurfaces simultaneously. The SURFINC parameter specifies the interval between displayed Plot3D surface values. (default: 0)

TERRAINPARMS Specifies various parameters used to characterize how terrain appears. The parameters are the color at the minimum depth, the color at the maximum depth and scaling factor used to vertically exaggerate the scene.

```
TERRAINPARMS
terrain_rgba_zmin[0],terrain_rgba_zmin[1],terrain_rgba_zmin[2];
terrain_rgba_zmax[0],terrain_rgba_zmax[1],terrain_rgba_zmax[2];
vertical_factor);
```

TIMEOFFSET Specifies that offset time in seconds added to the displayed simulation time. Along with the *SHOWHMSTIMELABEL* keyword, the *TIMEOFFSET* keyword allows one to display *wall clock* rather than simulation time. (default: 0.0)

TITLESAFE Amount in pixels to offset titles when displaying scene in *title safe* mode. (default: 0)

TRANSPARENT Specifies whether 2D and 3D contours should be drawn with solid colors (0) or transparent colors(1). (default: 1)

TWOSIDEDVENTS Specifies whether to draw vents so that they are visible from both sides (1) or visible from only one side (0). (default: 0)

USEGPU If the GPU is available, specifies whether it should be used. (default: 1)

VECLENGTH Specifies vector lengths.

VECTORSKIP Specifies what vectors to draw. For example, if this parameter is set to 2 then every 2nd vector is drawn when displaying vectors. (default: 1)

VIEWPOINT5 Specifies the internal Smokeview parameters used to record a scene's viewpoint and orientation. This parameter is set automatically by Smokeview when a .ini file is created. (default: none)

```
VIEWPOINT5
 eyeview,rotation_index,view_id
 eye\_x,eye\_y,eye\_z,zoom,zoomindex
 view_angle,direction_angle,elevation_angle,projection_type
 xcen,ycen,zcen
 angle_zx[0],angle_zx[1]
 mat[0],mat[1],mat[2],mat[3]
 mat[4],mat[5],mat[6],mat[7]
 mat[8],mat[9],mat[10],mat[11]
 mat[12],mat[13],mat[14],mat[15]
 xyz_clipplane,clip_x,clip_y,clip_z,clip_X,clip_Y,clip_Z
 clip_x_val,clip_y_val,clip_z_val,clip_X_val,clip_Y_valclip_Z_val
 name
```

- eyeview - view method type (0 - general rotations, 1 - first person movement, 2 - level rotations
- eye - coordinates of viewing position
- xcen, ycen, zcen - coordinates of view direction
- mat - viewing transformation matrix
- xyz_clipplane - global clipping flag (on=1, off=0)
- clip_x, clip_y, clip_z - min clipping plane flag ((on=1, off=0)
- clip_X, clip_Y, clip_Z - maxn clipping plane flag ((on=1, off=0)
- clip_x_val, clip_y_val, clip_z_val - min clipping plane values
- clip_X_val, clip_Y_val, clip_Z_val - max clipping plane values
- name - label appearing in Viewpoint menu

XYZCLIP Specifies clip planes in physical coordinates. There are six clipping planes, a minimum and maximum X, a minimum and maximum Y, a minimum and maximum Z. Each clipping plane may be used or not. The first parameter is 1 or 0 and specifies whether clipping is turned on or off. The next three lines specify clipping parameters for the X, Y and Z axes. Each line has the format

```
minflag min-clipval maxflag max-clipval
```

where the two flags, minflag and maxflag are 1 if turned on or 0 if turned off. Clipping is specified with the Clipping dialog box found under the *Options* menu item. (default:

```
0
0 0.0 0 0.0
0 0.0 0 0.0
0 0.0 0 0.0
```

ZOOM Specifies the zoom amount used to display a Smokeview scene using two parameters, an integer zoom index and a floating point zoom amount. If the zoom index is 0->4 then the zoom amount is 0.25, 0.5, 1.0, 2.0 and 4.0 respectively. If the zoom index is negative then the second parameter is used to specify the zoom amount. (default: 0 1.0)

D.3.6 Tour

SHOWPATHNODES Specifies whether the path nodes should (1) or should not (0) be drawn. This is a debugging parameter, not normally used. (default: 0)

SHOWTOURROUTE Specifies whether the tour route should (1) or should not (0) be drawn. (default: 0)

TOURS Keyword used to specify the tours. The format is

```
TOURS
ntours - number of tours
label for tour
nkeyframes - number of key frames for first tour
time xpos ypos zpos 1 az elev bias continuity tension zoom localspeedflag
...
time xpos ... for last key frame
nkeyframes for 2nd tour
...
...
```

If a Cartesian view direction is specified then instead of `1 az elev` above use 0 xview yview zview where xview, yview, zview are the coordinates of the view direction. The *Circle* tour is not stored in the .ini file unless it has been changed by the user. The tour entry created by using the `Add` button in the Tour dialog box is given by

```
TOURS
1
Added Tour 1
2
0.0 -1.0 -1.0 -1.0 0 0.0 0.0 0.0 0.0 0.0 1.0  0
100.0 7.4 9.0 7.4 0 0.0 0.0 0.0 0.0 0.0 1.0 0
```

TOURCOLORS Keyword used to specify the tour colors. The colors as before consist of a red, green and blue component ranging from 0.0 to 1.0 . One can override Smokeview's choice for the path, the path knots for both the selected and un-selected case. One may also specify the color of the time labels and the location of the object or avatar on the tour at the current time. The foreground color is used when a color component less than 0.0 is specified. Default:

```
TOURCOLORS
1.000000 0.000000 0.000000   :selected path line
1.000000 0.000000 0.000000   :selected path line knots
0.000000 1.000000 0.000000   :selected knot
```

```
-1.000000 -1.000000 -1.000000   :path line
-1.000000 -1.000000 -1.000000   :path knots
-1.000000 -1.000000 -1.000000   :text
1.000000 0.000000 0.000000   :avatar
```

TOURCONSTANTVEL Specifies whether the avatar should (1) or should not (0) traverse the path with a constant velocity. (default: 1)

VIEWALLTOURS Specifies whether all (1) tours should be drawn. (default: 0)

VIEWTIMES Specifies the tour start time, tour stop time and number of points to specify a tour. (default: 0.0 100.0 1000)

VIEWTOURFROM Specifies whether the scene should (1) or should not (0) be observed from the selected tour.

D.3.7 Realistic Smoke Parameters

ADJUSTALPHA The ALPHA parameter in OpenGL/Smokeview is used to specify the transparency of an object. The ALPHA's are computed in FDS and adjusted in SMOKEVIEW according to value of the ADJUSTALPHA keyword. If ADJUSTALPHA is zero, then no adjustments to ALPHA are made. If ADJUSTALHPA is one, then ALPHA's are adjusted for non-orthogonal view directions (ALPHA is increased to account for longer path lengths along non-orthogonal view directions. If ADJUSTALPHA is two, then ALPHA's are adjusted as in the ADJUSTALPHA=1 case and are also set to zero on wall at blockage boundaries (this reduces graphical artifacts). (default: 1)

FIRECOLOR Specifies the color of the fire in red, green, blue coordinates. Each color component is an integer ranging from 0 to 255. (default: 255, 128, 0)

FIREDEPTH Specifies the depth at which the fire is 50 percent transparent. (default: 2.0)

SMOKECULL Cull (or do not draw) smoke if it is outside of the viewing frustum. (default: 1)

SMOKESHADE Grey level of smoke, may range from 0 to 255. (default: 255)

SMOKESKIP To speed up smoke drawing, spatial frames may be skipped. Allowable parameters are (0, 1, 2). (default: 0)

SMOKETHICK The ALPHA transparency parameter may be divided by a power of two ($2^{\texttt{SMOKETHICK}}$) to make the smoke thinner. Parameters may range from 0 to 7. (default: 0)

D.3.8 Zone Fire Modeling Parameters

SHOWHZONE Specifies whether upper layer temperatures should be (1) drawn horizontally or not (0). (default: 0)

SHOWVZONE Specifies whether upper layer temperatures should be (1) drawn vertically or not (0). (default: 1)

SHOWHAZARDCOLORS Specifies whether upper layer temperatures should be (1) drawn in terms of hazard or drawn in terms of a standard color scale (0). (default: 0)

D.3.9 Local Parameters

SCRIPTFILE Specifies the name of a script file either created by hand or created automatically by Smokeview using the script recorder.

D.4 Smokeview Parameter Input File (.smv file)

The FDS software outputs simulation results into the Smokeview input file with extension `.smv` and various output data files whose format is documented in the next section. A `.smv` file is a formatted ascii text file consisting of a set of KEYWORDs followed by DATA describing the FDS case's geometry, data file names and contents, sensor information, etc.

D.4.1 Geometry Keywords

GRID This keyword specifies the number of grid cells in the `X`, `Y` and `Z` directions. For example,

```
GRID
10 20 30
```

specifies that there are 10, 20 and 30 grid cells in the `X`, `Y` and `Z` directions respectively.

OFFSET This keyword specifies signals to Smokeview that a new mesh has begun and also gives values for the front, left bottom corner of the mesh. For example,

```
OFFSET
xmin, ymin, zmin
```

Note that the xmin, ymin and zmin values must be identical to the corresponding values given in the `PDIM` keyword. The `OFFSET` keyword cannot be eliminated from the .smv file (it may seem logical to do this due to the presence of redundant data) because of its role in signaling new meshes.

PDIM This keyword specifies the region where a mesh is located using the same convention as is used in an FDS input file to specify a blockage location. PDIM also specifies a color to use for drawing grids. For example,

```
PDIM
xmin, xmax, ymin, ymax, zmin, zmax, r, g, b
```

where `(xmin, ymin, ymax)` and `(xmax, ymax, zmax)` represent opposite corners of a mesh and r, g and b represent the red, green and blue components (0.0 to 1.0) of grids drawn in the mesh.

Note that the xmin, ymin and zmin values must be identical to the values given in the `OFFSET` keyword.

SHOW_OBST(HIDE_OBST) This keyword specifies when a blockage should be shown(hidden). For example,

```
SHOW_OBST  2
  10 120.1
```

specifies that the tenth blockage in mesh 2 should be opened at 120.1 seconds. This keyword is automatically added to the `.smv` file by FDS.

OBST This keyword specifies internal blockages. A FORTRAN 2003 code segment describing the format of `OBST` data is given by:

```
read(5,*)nb
do i = 1, nb
  read(5,*)x1(i),x2(i),y1(i),y2(i),z1(i),z2(i), id(i),
      ... s1(i),...,s6(i),tx(i),ty(i),tz(i)
end do
do i = 1, nb
  read(5,*)ib1(i), ib2(i), jb1(i), jb2(i), kb1(i), kb2(i),
         ... colorindex(i) blocktype(i),
         ...  red(i), green(i), blue(i), alpha(i)
end do
```

where the parameters are defined in Table D.1. The arrays `x1, ..., z2` and `ib1, ..., kb2` are required. All other arrays are optional and may be omitted.

Table D.1: Descriptions of parameters used by the Smokeview OBST keyword.

Variable(s)	type	Description
nb	integer	number of blockages or entries for the OBST keyword
x1, x2 y1,y2 z1,z2	float	floating point blockage bounds
id	integer	blockage identifier
s1, s2 s3, s4 s5, s6	integer	index of surface (SURF) used to draw blockage sides
tx, ty, tz	float	texture origin
ib1, ib2 jb1, jb2 kb1, kb2	integer	Indices used to define blockage bounds in terms of grid locations.
colorindex	integer	Type of coloring used to color blockage. -1 - default color -2 - invisible -3 - use red, green, blue and alpha to follow (values follow) n>0 - use n'th color table entry
blocktype	integer	Defines how the blockage is drawn. -1 - use surface to obtain blocktype 0 - regular block 2 - outline
red, green, blue alpha	float	Each color value ranges from 0.0 to 1.0 . The alpha *color* represents transparency, alpha=0.0 is transparent, alpha=1.0 is opaque.

TOFFSET The `TOFFSET` keyword defines a default texture origin, (x_0, y_0, z_0) . This origin may be overridden with data provided with the `OBST` keyword. For example,

```
TOFFSET
0.0 0.0 0.0
```

TRNX,TRNY,TRNZ The `TRNX, TRNY, TRNZ` keywords specify grid nodes in the `X, Y, Z` coordinate directions. A FORTAN 2003 code segment describing the format of `TRNX` data is given by:

```
read(5,*)nv
do i = 1, nv
  read(5,*)idummy
end do
do i = 1, nv
  read(5,*)xplt(i)
end do
```

`TRNY` and `TRNZ` data entries are defined similarly. The first `nx` data items are not required by Smokeview.

VENT, CVENT These keywords specify vent coordinates for regular and circular vents. Note that the parameters `x0, y0, z0` and `radius` describing the center and radius of a circular vent only appear with the CVENT keyword. A FORTRAN 2003 code segment describing the format of `VENT` and `CVENT` data is given by:

```
read(5,*)nv
do i = 1, nv
  read(5,*)xv1(i), xv2(i), yv1(i), yv2(i), zv1(i), zv2(i), id(i)
   ... s1(i), tx(i), ty(i), tz(i) % x0, y0, z0, radius
end do
do i = 1, nv
  read(5,*)iv1(i), iv2(i), jv1(i), jv2(i), kv1(i), kv2(i)
   ... index(i), type(i), red(i), green(i), blue(i), alpha(i)
end do
```

where the parameters are defined in Table D.2. The arrays `xv1, ..., zv2` and `iv1, ..., kv2` are required. All other arrays are optional and may be omitted.

OPEN_VENT(CLOSE_VENT) This keyword specifies when a vent should be opened(closed). For example,

```
OPEN_VENT 2
3 15.6
```

specifies that the third vent in mesh 2 should be opened at 15.6 S.

XYZ The `XYZ` keyword defines the .xyz or `Plot3D` grid file name. A FORTAN 2003 code segment describing the format of `XYZ` data is given by:

```
read(5,"(a)")xyzfilename
```

Table D.2: Descriptions of parameters used by the Smokeview VENT and CVENT keywords.

Variable(s)	type	Description
nv	integer	number of vents or entries for the VENT keyword
xv1, xv2 yv1,yv2 zv1,zv2	float	floating point bounds
id	integer	vent identifier
s1	integer	index of surface (SURF) used to draw vent
tx, ty, tz	float	texture origin
iv1, iv2 jv1, jv2 kv1, kv2	integer	Indices used to define vent bounds in terms of grid locations.
index	integer	Type of coloring used to color vent. -99 or +99 - use default color -n or +n - use n'th palette color <0 - do not draw boundary file over vent >0 - draw boundary file over vent
type	integer	Defines how the vent is drawn. 0 - solid surface 2 - outline -2 - hidden
red, green, blue alpha	float	Each color value ranges from 0.0 to 1.0 . The alpha *color* represents transparency, alpha=0.0 is transparent, alpha=1.0 is opaque.
x0, y0, z0	float	circular vent origin
radius	float	if positive, radius of circular vent

where xyzfilename is a character variable containing the name of the .xyz file.

D.4.2 File Keywords

BNDF The BNDF keyword defines the .bf file name along with character labels used to describe the data contents of the boundary file.

HRRPUVCUT The HRRPUVCUT keyword defines the heat release rate per unit volume cutoff value. When displaying realistic smoke and fire, fire is displayed above this cutoff and smoke is displayed below.

INPF The INPF keyword specifies a file containing a copy of the FDS input file.

ISOF The ISOF keyword defines the .iso file name along with character labels used to describe the data contents of the isosurface file.

PART,PRT5 The PART and PRT5 keywords define the .part file name along with character labels used to describe the data contents of the particle file.

PL3D The PL3D keyword defines the .q file name along with character labels used to describe the data contents for each Plot3D variable.

SLCF,SLCT The SLCF and SLCT keywords define the .sf file name along with character labels used to describe the data contents of the slice file. The SLCT keyword is used for wildland urban interface fire simulations performed over terrains. Smokeview allows one to visualize slice files for these types of simulations that conform to the terrain.

D.4.3 Device (sensor) Keywords

DEVICE A *device* generalizes the notion of a sensor, sprinkler or heat detector. The DEVICE keyword defines the device location and name. This name is used to access the set of instructions for drawing the device. These instructions are contained in a file named objects.svo. The default location of this file on the PC is C:\Program Files\FDS\FDS5\bin\objects.svo . This file may be customized by the user by adding instructions for devices of their own design (i.e., custom devices may be added to a Smokeview visualization without re-programming Smokeview). See Chapter 6 for more information on how to do this. The format for the DEVICE keyword is

```
DEVICE
device_name
x y z xn yn zn 0 0 % PROP_ID
```

where device_name is the entry in the objects.svo file used to draw the device and (x,y,z) and (xn,yn,zn) are the location and direction vector of the device. The label PROP_ID (prepended by a %) is the PROP entry used to list of other properties used for drawing the device (see the PROP entry in this section for more details). Note, the two 0 0 numbers are used for backwards compatibility.

DEVICE_ACT The DEVICE_ACT keyword defines the activation time for a particular device. The format for the DEVICE_ACT keyword is

```
DEVICE_ACT
idevice time state
```

where `time` is the activation time of the `idevice`'th device. State is the state of the device, 0 for off or in-active and 1 for on or active. If the device may be drawn more than two ways then more than 2 states may be used with this keyword.

HEAT The `HEAT` keyword defines heat detector location data. A FORTAN 2003 code segment describing the format of `HEAT` data is given by:

```
read(5,*)nheat
do i = 1, nheat
  read(5,*)xheat(i),yheat(i),zheat(i)
end do
```

where `nheat` is the number of heat detectors and `xheat, yheat, zheat` are the `x, y, z` coordinates of the heat detectors.

HEAT_ACT The `HEAT_ACT` keyword defines heat detector activation data. A FORTAN 2003 code segment describing the format of `HEAT_ACT` data is given by:

```
read(5,*)iheat, heat_time
```

where `heat_time` is the activation time of the `iheat`'th heat detector.

PROP The `PROP` keyword specifies a list of general properties used by Smokeview to customize the drawing of devices defined in the objects.svo file. The format of the `PROP` keyword is

```
PROP
 prop_id           (character string)
 smokeview_id      (character string)
 number of keyword/value pairs  (integer)
 keyword1=value1   (character string)
 ..
 ..
 keywordn=valuen   (character string)
 number of texture files (integer) (0 or 1 for now)
 texture file 1
 ...
 ...
 texture file n
```

SPRK The `SPRK` keyword defines sprinkler location data. A FORTAN 2003 code segment describing the format of `SPRK` data is given by:

```
read(5,*)nsprink
do i = 1, nsprink
  read(5,*)xsprink(i),ysprink(i),zsprink(i)
end do
```

where `nsprink` is the number of sprinklers and `xsprink, ysprink, zsprink` are the `x, y, z` coordinates of the sprinklers.

SPRK_ACT The `SPRK_ACT` keyword defines sprinkler activation data. A FORTAN 2003 code segment describing the format of `SPRK_ACT` data is given by:

```
read(5,*)isprink, sprink_time
```

where `sprink_time` is the activation time of the `isprink`'th sprinkler.

THCP The THCP keyword defines thermocouple location data. A FORTAN 2003 code segment describing the format of THCP data is given by:

```
read(5,*)ntherm
do i = 1, ntherm
  read(5,*)xtherm(i),ytherm(i),ztherm(i)
end do
```

where ntherm is the number of thermocouples and xtherm, ytherm and ztherm are the x, y and z coordinates of the thermocouples.

D.4.4 Zone Modeling Keywords

This section contains documentation for keywords used to describe features found in a zone fire model, features such as rooms, vents, fires, etc. Smokeview also supports a number of the keywords described above to support visualization of slice files, isosurface files, and devices for zone fire models.

FIRE The FIRE keyword defines the location and room number of a zone fire modeling fire. The format for the FIRE keyword is

```
FIRE
 i x y z
```

where i is the room number containing the fire and (x,y,z) is the location within the room of the base of the fire. One FIRE entry is specified for each fire. The order of the FIRE entries found in the .smv file should correspond to the ordering of the fires (i.e., The n'th fire is specified with the n'th .smv FIRE entry.)

VENTGEOM The VENTGEOM keyword defines the location, orientation and size of a zone fire modeling vent (with horizontal flow). The format for the VENTGEOM keyword is

```
VENTGEOM
 from to face width ventoffset bottom top r g b
```

where from and to are the room indices (ranging from 1 to the number of rooms) of the "from" and "to" rooms, face is the index of the wall (or face) where the vent is located (front wall=1, right wall=2, back wall=3, left wall=4), width is the vent width, bottom is the elevation (relative to the floor) of the vent sill, top is the elevation (relative to the floor) of the vent soffit and r, g, b are the red, green and blue components (ranging from 0.0 to 1.0) of the vent color. The order of the VENTGEOM entries found in the .smv file should correspond to the ordering of the vents (i.e., The n'th vent is specified with the n'th .smv VENTGEOM entry.) Note that the VENTGEOM keyword is not used in CFAST 7 files and has been replaced by HVENTPOS VVENTPOS and MVENTPOS keywords described below.

HVENTPOS defines the location, orientation, size, color, and initial opening for a zone fire modeling wall vent. The format for the HVENTPOS keyword is

```
HVENTPOS
 from to x1 x2 y2 y2 z1 z2 red green blue opening
```

where `from` and `to` are the room indices (ranging from 1 to the number of rooms) of the "from" and "to" rooms, `x1 x2 y1 y2 z1 z2` define lower left and upper right coordinates of the vent relative to the lower left/front corner of the `from` compartment, `red green blue` (optional) specify the color of the drawn vent (color values range from 0.0 to 1.0), and `opening` (optional) is the initial opening area of the vent.

VVENTPOS defines the location, orientation, size, color, and initial opening for a zone fire modeling ceiling/floor vent. The format for the `VVENTPOS` keyword is

```
VVENTPOS
 from to x1 x2 y2 y2 z1 z2 red green blue opening
```

where `from` and `to` are the room indices (ranging from 1 to the number of rooms) of the "from" and "to" rooms, `x1 x2 y1 y2 z1 z2` define lower left and upper right coordinates of the vent relative to the lower left/front corner of the `from` compartment, `red green blue` (optional) specify the color of the drawn vent (color values range from 0.0 to 1.0), and `opening` (optional) is the initial opening area of the vent.

MVENTPOS defines the location, orientation, size, color, and initial opening for a zone fire modeling wall vent. The format for the `MVENTPOS` keyword is

```
MVENTPOS
 from to x1 x2 y2 y2 z1 z2 red green blue opening
```

where `from` and `to` are the room indices (ranging from 1 to the number of rooms) of the "from" and "to" rooms, `x1 x2 y1 y2 z1 z2` define lower left and upper right coordinates of the vent relative to the lower left/front corner of the `from` compartment, `red green blue` (optional) specify the color of the drawn vent (color values range from 0.0 to 1.0), and `opening` (optional) is the initial opening area of the vent.

ROOM The `ROOM` keyword defines the size and location of a zone fire modeling compartment or room. The format for the `ROOM` keyword is

```
ROOM
 x y z
 x0 y0 z0
```

ROOM where (x,y,z) is the width, depth and height of the room respectively and $(x0,y0,z0)$ is the location of the left, front, bottom corner of the room. The order of the `ROOM` entries found in the .smv file should correspond to the ordering of the rooms (i.e., The n'th room is specified with the n'th .smv `ROOM` entry.)

ZONE The `ZONE` keyword defines the file used to store zone fire modeling data and the types of data found within the file. The format for the `ZONE` keyword is

```
ZONE
 file
 long label
 short label
 unit
 long label
 short label
 unit
 long label
```

```
short label
unit
long label
short label
unit
```

where `file` is the name of the file containing the zone fire modeling data. `long label short label` and `unit` describe the columns of data in the `ZONE file`Ṅote, Smokeview does not use the label or unit data with CFAST 6 or later. This data is found within the spread sheet data now used to store zone fire modeling data.

D.4.5 Miscellaneous Keywords

REVISION The `REVISION` keyword defines the GIT revision or build number for the version of FDS that ran the case being visualized. The FDS and Smokeview revisions are displayed in the Help menu.

TITLE1/TITLE2 The `TITLE1` and `TITLE2` keywords allow one to specify extra information documenting a Smokeview case. These keywords and associated labels are added by hand to the Smokeview (`.smv) file using the format`:

```
TITLE1
first line of descriptive text
TITLE2
second line of descriptive text
```

D.5 CAD/GE1 file format

A program called DXF2FDS, written by David Sheppard of the US Bureau of Alcohol, Tobacco and Firearms (ATF), creates an FDS input file and a Smokeview geometric description file (.GE1) given a CAD description of the building being modeled. The CAD description must be in a *dxf* format and created using *3DFACE* commands. The .GE1 file has a simple text format and is described below. DXF2FDS specifies that .GE1 filename on the `&DUMP` line in an FDS input file using the `RENDER_FILE` keyword, as in

```
&DUMP RENDER_FILE='Capecod.GE1' /
```

FDS to perform computations and a CAD view.

```
[APPEARANCE]
nappearances
string (material description)
index r g b twidth, theight, alpha, shininess, tx0, ty0, tz0
tfile
:
:   The above entry is repeated nappearances-1 more times
:
[FACES]
nfaces
x1 y1 z1 x2 y2 z2 x3 y3 z3 x4 y4 z4 index
The above line is repeated nfaces-1 more times.
```

nappearances Number of appearance entries to follow Each appearance entry has 3 lines.

string A material description is written out by DX2FDS but is ignored by Smokeview.

index An index number starting at 0.

r, g, b Red green and blue components of the CAD face used when a texture is not drawn. The values of r, g and b range from 0 to 255. If a color is not used then use -1 for each color component. In this case, the CAD face is opaque regardless of the alpha value specified.

twidth, theight Textures are tiled or repeated. The characteristic width and length of the texture file is twidth and theight respectively.

alpha Opaqueness value of the cad element being drawn. Values may range from 0.0 (completely transparent) to 1.0 (completely opaque. (default: 1.0)

shininess Shininess value of the cad element being drawn. Values may be larger than 0.0 . (default: 800.0)

tx0, ty0, tz0 x, y and z values in physical coordinates of the offset used to apply a texture to a cad element. (default: 0.0, 0.0, 0.0)

tfile The name of the texture file. If one is not used or available then leave this line blank.

nfaces Number of face entries to follow. Each face entry has one line.

x1/y1/z1/.../x4/y4/z4 x,y,z coordinates of a quadrilateral. T he four corners of the quad must lie in a plane or weird effects may result when Smokeview draws it. (This is a requirement of OpenGL). The four points should be in counter-clockwise order.

index Points to a material in the [APPEARANCE] section.

D.6 Objects.svo

```
// ************ object file format ********************

//  1. comments and blank lines may be placed anywhere
//  2. any line not beginning with "//" is part of the definition.
//  3. the first non-comment line after OBJECTDEF is the object name
//  4. an object definition may contain, labels, numerical constants
//     a number), string constants (enclosed in " ") and/or
//     commands (beginning with a-z)
//  5. a label begins with ':' as in :dx
//  6. the label :dx may be accessed afterward using $dx
//  7. An object may contain multiple frames or states.  A new frame within
//     an object is defined using NEWFRAME

// OBJECTDEF // OBJECTDEF begins the object definition

//   object_name // name or label for object
//   :var1 ... :varn  // a series of labels may be specified for use by
//                    // the object definition.  Data is copied to these
//                    // label locations using the SMOKEVIEW_PARAMETERS
//                    // &PROP keyword or from a particle file. The data
//                    // in :varn may be referenced  elsewhere in the
//                    // definition using $varn

//   // A series of argument/command pairs are specified on one or
```

```
//   // more lines.

//   arg1 ... argn command1 arg1 ... argn command2 ...

//   // An argument may be a numerical constant (e.g. 2.37), a string
//   // (e.g. "SKYBLUE"), a label (e.g. :var1),  or a reference to a
//   // label located elsewhere (e.g. $var1)

//  NEWFRAME    // beginning of next frame
//   more argument/command pairs for the next object frame
//   ....

   // ************ static object definitions - single frame/state ********************

OBJECTDEF
 debug_thermocouple
 "BLUE" setcolor
 push 0.00625 0.00625 0.075 scalexyz 1.0 1.0 drawdisk pop push 0.0 0.0 0.0375
     translate 0.008 drawsphere pop
 "RED" setcolor
 push .075 0.00625 0.00625 scalexyz 1.0 1.0 drawdisk pop push 0.0375 0.0 0.0
     translate 0.008 drawsphere pop
 "GREEN" setcolor
 push 0.00625  0.075 0.00625 scalexyz 1.0 1.0 drawdisk pop push 0.0 0.0375 0.0
     translate 0.008 drawsphere pop

 0 0 0 setrgb
 90.0 rotatex
 0.005 0.005 0.005 scalexyz
 2.0 drawsphere
 153 153 153 setrgb
 push 3.4 0.0 3.5 translate 90.0 rotatey 0.5 5.0 drawdisk pop
 push  45.0 rotatey 0.5 5.0 drawdisk pop
 push 3.4 0.0 -3.5 translate 90.0 rotatey 0.5 5.0 drawdisk pop
 push  135.0 rotatey 0.5 5.0 drawdisk pop

OBJECTDEF
 thermocouple
 0 0 0 setrgb
 90.0 rotatex
 0.005 0.005 0.005 scalexyz
 2.0 drawsphere
 153 153 153 setrgb
 push 3.4 0.0 3.5 translate 90.0 rotatey 0.5 5.0 drawdisk pop
 push  45.0 rotatey 0.5 5.0 drawdisk pop
 push 3.4 0.0 -3.5 translate 90.0 rotatey 0.5 5.0 drawdisk pop
 push  135.0 rotatey 0.5 5.0 drawdisk pop

OBJECTDEF
 target
 153 153 153 setrgb
 0.0 0.0 -0.005 translate 0.2 0.01 drawdisk

OBJECTDEF  // used by smokeview to display smoke thickness
 smokesensor
 "WHITE" setcolor
 0.15 drawsphere

OBJECTDEF  // draw a plane intersecting through FDS meshes
```

```
 plane
 "GREEN" setcolor
 0.038 drawsphere

   // ************ static object definitions - multiple frames/states
       ********************

OBJECTDEF
 sensor
0 255 0 setrgbval
0.25 SCALEGRID 1.0 drawsphere
 NEWFRAME
 255 0 0 setrgbval
0.25 SCALEGRID 1.0 drawsphere
 NEWFRAME
 0 0 255 setrgbval
0.25 SCALEGRID 1.0 drawsphere
 NEWFRAME
 255 255 255 setrgbval
0.25 SCALEGRID 1.0 drawsphere

OBJECTDEF
 heat_detector         // label, name of object

 // The heat detector has three parts
 //   a disk, a truncated disk and a sphere.
 //   The sphere changes color when activated.

 204 204 204 setrgb  // set color to off white
 180.0 rotatey 0.0 0.0 0.03 translate
 push 0.0 0.0 -0.02 translate 0.127 0.04 drawdisk pop
 push 0.0 0.0 -0.04 translate
 0.06 0.08 0.02 drawtrunccone pop
 "GREEN" setcolor
 push 0.0 0.0 -0.03 translate  0.04 drawsphere pop
 // push and pop are not necessary in the last line
 //   of a frame.  Its a good idea though to prevent
 //   problems if parts are added later.
 NEWFRAME  // beginning of activated definition
 204 204 204 setrgb
 180.0 rotatey 0.0 0.0 0.03 translate
 push 0.0 0.0 -0.02 translate 0.127 0.04 drawdisk pop
 push 0.0 0.0 -0.04 translate
   0.06 0.08 0.02 drawtrunccone pop
 "RED" setcolor
 push 0.0 0.0 -0.03 translate
   0.04 drawsphere pop

OBJECTDEF
 sprinkler_upright
 180.0 rotatey 0.0 0.0 -0.04 translate
 "BRICK" setcolor
 push  0.0 0.0 -0.015 translate 0.03 0.03 drawdisk  pop
 push  0.0105 0.0 0.055 translate -22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
 push -0.0105 0.0 0.055 translate  22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
 push  0.019 0.0 0.02 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
```

```
 push -0.019 0.0 0.02 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
 push 0.0 0.0 0.07 translate
   0.010 0.017 0.020 drawtrunccone pop
 push 0.0 0.0 0.089 translate
   0.064 0.002 0.004 -1.0 drawnotchplate pop
 "GREEN" setcolor
 push 0.00 0.0 0.04 translate
   0.4 0.4 1.0 scalexyz 0.03 drawsphere pop
 NEWFRAME
 "BRICK" setcolor
 180.0 rotatey 0.0 0.0 -0.04 translate
 push  0.0 0.0 -0.015 translate 0.03 0.03 drawdisk  pop
 push 0.0105 0.0 0.055 translate -22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
 push 0.0190 0.0 0.020 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
 push -0.0105 0.0 0.055 translate 22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
 push -0.0190 0.0 0.020 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
 push 0.0 0.0 0.07 translate
   0.01 0.017 0.02 drawtrunccone pop
 push 0.0 0.0 0.089 translate
   0.064 0.002 0.004 -1.0 drawnotchplate pop
 "BLUE" setcolor
 push 0.0 0.0 0.015 translate  0.015 drawsphere pop

OBJECTDEF
 sprinkler_pendent
 "BRICK" setcolor
 0.0 0.0 -0.04 translate
 push 0.0 0.0 -0.015 translate 0.03 0.03 drawdisk pop
 push 0.0105 0.0 0.055 translate -22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
 push 0.019 0.0 0.02 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
 push -0.0105 0.0 0.055 translate 22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
 push -0.019 0.0 0.02 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
 push 0.0 0.0 0.07 translate
   0.01 0.017 0.02 drawtrunccone pop
 push 0.0 0.0 0.089 translate
   0.064 0.002 0.008 1.0 drawnotchplate pop
 "GREEN" setcolor
 push 0.00 0.0 0.04 translate
   0.4 0.4 1.0 scalexyz 0.03 drawsphere pop
 NEWFRAME
 "BRICK" setcolor
 push
 0.0 0.0 -0.04 translate
 push 0.0 0.0 -0.015 translate 0.03 0.03 drawdisk pop
 push 0.0105 0.0 0.055 translate -22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
 push 0.019 0.0 0.02 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
 push -0.0105 0.0 0.055 translate 22 rotatey
   0.0085 0.004 0.05 scalexyz 1.0 drawcube pop
```

```
 push -0.019 0.0 0.02 translate
   0.0085 0.004 0.03 scalexyz 1.0 drawcube pop
 push 0.0 0.0 0.07 translate
   0.01 0.017 0.02 drawtrunccone pop
 push 0.0 0.0 0.089 translate
   0.064 0.002 0.008 1.0 drawnotchplate pop
 "BLUE" setcolor
 push 0.0 0.0 0.015 translate 0.015 drawsphere pop
 pop

OBJECTDEF
 smoke_detector
 204 204 204 setrgb
 180.0 rotatey 0.0 0.0 0.02 translate
 push 0.0 0.0 -0.025 translate 0.127 0.05 drawdisk pop
 "GREEN" setcolor
 push 0.0 0.0 -0.02 translate 0.04 drawsphere pop
 26 26 26 setrgb
 push 0.0 0.0 -0.028 translate 0.10 0.11 0.02 drawring pop
 push 0.0 0.0 -0.028 translate 0.07 0.08 0.02 drawring pop
 push 0.0 0.0 -0.028 translate 0.04 0.05 0.02 drawring pop
 NEWFRAME
 204 204 204 setrgb
 180.0 rotatey 0.0 0.0 0.02 translate
 push 0.0 0.0 -0.025 translate 0.127 0.05 drawdisk pop
 "RED" setcolor
 push 0.0 0.0 -0.02 translate 0.04 drawsphere pop
 26 26 26 setrgb
 push 0.0 0.0 -0.028 translate 0.10 0.11 0.02 drawring pop
 push 0.0 0.0 -0.028 translate 0.07 0.08 0.02 drawring pop
 push 0.0 0.0 -0.028 translate 0.04 0.05 0.02 drawring pop


OBJECTDEF
 nozzle
 0.0 0.0 -0.041402 translate
 "BRICK" setcolor
 0.022225 0.0127 drawhexdisk
 push 0.0 0.0 0.0127 translate 0.01905 0.01905 drawdisk pop
 push 0.0 0.0 0.031751 translate 0.01905 0.009525 drawhexdisk pop
 204 204 204 setrgb
 push 0.0 0.0 0.035052 translate 0.00635 0.00635 drawdisk pop
 NEWFRAME
 0.0 0.0 -0.041402 translate
 "BRICK" setcolor
 0.022225 0.0127 drawhexdisk
 push 0.0 0.0 0.0127 translate 0.01905 0.01905 drawdisk pop
 push 0.0 0.0 0.031751 translate 0.01905 0.009525 drawhexdisk pop
 "BLUE" setcolor
 push 0.0 0.0 0.035052 translate 0.00635 0.00635 drawdisk pop
 push 0.0 0.0 0.035052 translate 0.00635 drawsphere pop
 push "BLUE" setcolor
   0.0 0.0 0.0414 translate 0.012 drawsphere pop

OBJECTDEF
 arrow
push 0.0 0.0 1.0 translate 0.4 0.6 drawcone pop
push 0.1 0.1 1.0 scalexyz 1.0 1.0 drawdisk pop
```

```
OBJECTDEF
truck
 push
 push
 255 26 0 setrgb
 0.25 0.25 0.25 scalexyz
 0.0 0.0 -0.35 translate
 180.0 rotatez
 0.0   0.0 0.125 3.0 1.4 0.5     drawboxxyz
 0.625 0.0 0.625 0.5 1.4 0.0 0.5 drawprismxyz
 1.125 0.0 0.625 0.5 1.4 0.5     drawboxxyz

 0 0 0 setrgb
 push .375 0.075 0.2 translate 0.5 0.15 drawwheel pop
 push .375 1.485 0.2 translate 0.5 0.15 drawwheel pop
 push  2.5 0.075 0.2 translate 0.5 0.15 drawwheel pop
 push  2.5 1.485 0.2 translate 0.5 0.15 drawwheel pop

 pop
 pop

OBJECTDEF
car
 push
 push
 0 144 255 setrgb
 0.25 0.25 0.25 scalexyz
 0.0 0.0 -0.35 translate
 180.0 rotatez

 0.0   0.0 0.125 0.875 1.4 0.375 0.5   drawprismxyz
 0.875 0.0 0.125 0.375 1.4 0.5   0.625 drawprismxyz
 1.25  0.0 0.125 0.75  1.4 0.625 0.625 drawprismxyz
 2.0   0.0 0.125 0.375 1.4 0.625 0.5   drawprismxyz
 2.375 0.0 0.125 0.5   1.4 0.5   0.375 drawprismxyz

 0 0 0 setrgb
 push .375 0.075 0.2 translate 0.5 0.15 drawwheel pop
 push .375 1.485 0.2 translate 0.5 0.15 drawwheel pop
 push  2.5 0.075 0.2 translate 0.5 0.15 drawwheel pop
 push  2.5 1.485 0.2 translate 0.5 0.15 drawwheel pop

 pop
 pop

   // ************ object definitions used for FDS-EVAC ********************


OBJECTDEF
 evacbox
// draws a square "railings"
// smv file: xyz is the (min_x, min_y, z) corner
//           orientation vector (0,0,1)
// Red Green Blue  width(x) depth(y) diameter
 :R=0 :G=0 :B=0 :DX :DY :D
 $R $G $B setrgb
 push  0.0 0.0 0.0 translate -90.0 rotatex $D $D $DY scalexyz 1.0 1.0 drawdisk pop
 push  $DX 0.0 0.0 translate -90.0 rotatex $D $D $DY scalexyz 1.0 1.0 drawdisk pop
 push  0.0 0.0 0.0 translate  90.0 rotatey $D $D $DX scalexyz 1.0 1.0 drawdisk pop
```

```
 push  0.0 $DY 0.0 translate  90.0 rotatey $D $D $DX scalexyz 1.0 1.0 drawdisk pop
// push  $DX 0.0 0.0 translate -90.0 rotatex $D $D $DY scalexyz 1.0 drawcubec pop

OBJECTDEF
 evacdoor         // label, name of object
 :SX :SY :SZ :R=0 :G=0 :B=0 :DX :DY :DZ
 // Evacuation input: door or exit namelist
 push  $DX $DY $DZ translate -180.0 rotatex
    34 139 3 setrgb 0.0 0.0 -0.6 translate 0.4 0.6 drawcone pop // draw an arrow
 -90.0 rotatez -90.0 rotatex
 push  $SX $SY $SZ scalexyz $R $G $B setrgb
    0.0 -0.5 0.0 translate 1.0 drawcube pop // front half of door (user specified
        color)
 push  $SX $SY $SZ scalexyz 34 139 3 setrgb
    0.0 0.5 0.0 translate 1.0 drawcube pop // back half of door (forest green)
NEWFRAME
 :SX :SY :SZ :R=0 :G=0 :B=0 :DX :DY :DZ
 -90.0 rotatez -90.0 rotatex
 push  $SX $SY $SZ scalexyz $R $G $B setrgb
    0.0 -0.5 0.0 translate 1.0 drawcube pop // front half of door (user specified
        color)
 push  $SX $SY $SZ scalexyz "RED" setcolor
    0.0 0.5 0.0 translate 1.0 drawcube pop // back half of door (red)

OBJECTDEF
 evacincline         // label, name of object
 :SX :SY :SZ :R=0 :G=0 :B=0
 // Evacuation input: evss namelist
 -90.0 rotatez -90.0 rotatex
 push  $SX $SY $SZ scalexyz $R $G $B setrgb
    0.0 0.5 0.0 translate 1.0 drawcube pop // incline (user specified color)

OBJECTDEF
 evacentr         // label, name of object
 :SX :SY :SZ :R=0 :G=0 :B=0
 // Evacuation input: entr namelist
 -90.0 rotatez -90.0 rotatex
 push  $SX $SY $SZ scalexyz $R $G $B setrgb
    0.0 -0.5 0.0 translate 1.0 drawcube pop // front half of door (user specified
        color)
 push  $SX $SY $SZ scalexyz 135 206 235 setrgb
    0.0 0.5 0.0 translate 1.0 drawcube pop // back half of door (sky blue 135 206 235)
NEWFRAME
 :SX :SY :SZ :R=0 :G=0 :B=0
 -90.0 rotatez -90.0 rotatex
 push  $SX $SY $SZ scalexyz $R $G $B setrgb
    0.0 -0.5 0.0 translate 1.0 drawcube pop // front half of door (user specified
        color)
 push  $SX $SY $SZ scalexyz "RED" setcolor
    0.0 0.5 0.0 translate 1.0 drawcube pop // back half of door (red)

   // ************ dynamic particle object definitions ********************
   //          (modifable using data obtained from FDS)

OBJECTDEF // object for particle file sphere
 sphere
 :R=0 :G=0 :B=0 :D=0.1
 $R $G $B setrgb
 $D drawsphere
```

```
OBJECTDEF
 box
 :R=0 :G=0 :B=0 :DX :DY :DZ
 $R $G $B setrgb
 $DX $DY $DZ scalexyz 1.0 drawcubec

OBJECTDEF // object for particle file tube
 tube
 :R=0 :G=0 :B=0 :D=0.1 :L=0.1 :RANDXY=0.0 :RANDXZ=0 :RANDYZ=0 :RANDXYZ=0.0 :DIRX=0.0
     :DIRY=0.0 :DIRZ=0.0
 $R $G $B setrgb
 $RANDXY randxy $RANDXZ randxz $RANDYZ randyz $RANDXYZ randxyz $DIRX $DIRY $DIRZ
     orienx 90.0 rotatey $D $L drawcdisk

OBJECTDEF // object for particle file tube
 cylinder
 :R=0 :G=0 :B=0 :D=0.1 :L=0.1
 $R $G $B setrgb
 90.0 rotatey $D $L drawcdisk

OBJECTDEF // object for particle "egg"
 egg
 :R=0 :G=0 :B=0 :D=0.1 :DX     // data obtained from an FDS input file
 $R $G $B setrgb
 $DX $D $D scalexyz 1.0 drawsphere

OBJECTDEF // object for particle file tube
 veltube
 :R=0 :G=0 :B=0 :D=0.1 :L=0.1 :U-VEL=1.0 :V-VEL=1.0 :W-VEL=1.0 :VELMIN=0.01
     :VELMAX=0.2
 $R $G $B setrgb
 $U-VEL :UV abs $UV $VELMAX :U div $U 0.0 1.0 :CU clip
 $V-VEL :VV abs $VV $VELMAX :V div $V 0.0 1.0 :CV clip
 $W-VEL :WV abs $WV $VELMAX :W div $W 0.0 1.0 :CW clip
 $CU $CV $CW rotatexyz $CU $CV $CW scalexyz $D $L drawcdisk

OBJECTDEF // color with FDS quantity, stretch with velocity
 velegg
 :R=0 :G=0 :B=0 :D=1.0 :U-VEL=1.0 :V-VEL=1.0 :W-VEL=1.0 :VELMIN=0.01 :VELMAX=0.2 //
     data obtained from an FDS input file
 $R $G $B setrgb
 $U-VEL :UV abs $UV $VELMAX :U div $U 0.0 1.0 :CU clip
 $V-VEL :VV abs $VV $VELMAX :V div $V 0.0 1.0 :CV clip
 $W-VEL :WV abs $WV $VELMAX :W div $W 0.0 1.0 :CW clip
 $CU $CV $CW rotatexyz $CU $CV $CW scalexyz $D drawsphere

OBJECTDEF // object for particle "egg"
 tempegg
 :R=0 :G=0 :B=0 :D=0.1 :DX :temp :rot_rate    // data obtained from an FDS input file
 $temp 700.0 :tempd28 div $tempd28 $G $B setrgb
 $rot_rate 0.0 :rotz multiaddt $rotz rotatez 0.2 $tempd28 0.2 scalexyz 1.0 drawsphere

OBJECTDEF
 block
 :R=0 :G=0 :B=0 :DX=1.0 :DY=1.0 :DZ=1.0 :ZANGLE=0.0
 $R $G $B setrgb
 $ZANGLE rotatez $DX $DY $DZ scalexyz 1.0 drawcubec
```

```
OBJECTDEF // object for a general ball
 ball
 :R=0 :G=0 :B=0 :DX :DY :DZ :D=-.1
 $D 0.0 :DGT0 GT
 $R $G $B setrgb
 $DGT0 IF
  $D drawsphere
  ELSE
  $DX $DY $DZ scalexyz 1.0 drawsphere
 ENDIF
 NO_OP

OBJECTDEF
 face_eye
  :R=0 :G=0 :B=0 :W :H
 $R $G $B setrgb
 rotateeye $W $H 1.0 scalexyz 1.0 drawsquare

OBJECTDEF // object for dynamic textured sphere
 movingsphere
 :R=0 :G=0 :B=0 // sphere color
 :X0 :Y0 :Z0    // sphere origin
 :VX :VY :VZ    // sphere velocity
 :ROTATE_RATE    // rotation rate
 :D=0.1          // sphere diameter
 :tfile          // texture file
 $R $G $B setrgb
 $VX $X0 :vvx multiaddt
 $VY $Y0 :vvy multiaddt
 $VZ $Z0 :vvz multiaddt
 $vvx $vvy $vvz translate $ROTATE_RATE 0.0 :rotz multiaddt
   $rotz rotatez 180.0 rotatey $tfile :textureindex gettextureindex $textureindex $D
       drawtsphere

OBJECTDEF // object for a moving box
 movingbox
 :R=255 :G=0 :B=0 // box color
 :X0 :Y0 :Z0    // lower left front box corner
 :VX :VY :VZ    // box velocity
 :DX :DY :DZ    // box size
 :XMAX :YMAX :ZMAX
 $R $G $B setrgb
 $VX $X0 :vvx multiaddt $vvx 0.0 $XMAX :CLIPX mirrorclip
 $VY $Y0 :vvy multiaddt $vvy 0.0 $YMAX :CLIPY mirrorclip
 $VZ $Z0 :vvz multiaddt $vvz 0.0 $ZMAX :CLIPZ mirrorclip
 $CLIPX $CLIPY $CLIPZ gtranslate $DX $DY $DZ scalexyz 1.0 drawcube

OBJECTDEF // object for dynamic textured sphere
 demosphere
 :R=0 :G=0 :B=0 // sphere color
 :X0 :Y0 :Z0    // sphere origin
 :VX :VY :VZ    // sphere velocity
 :ROTATE_RATE    // rotation rate
 :D=0.1          // sphere diameter
 :tfile          // texture file
 :XMAX :YMAX :ZMAX // box bounds
 $R $G $B setrgb
 $VX $X0 :vvx multiaddt $vvx  0.0 $XMAX :CLIPX mirrorclip
 $VY $Y0 :vvy multiaddt $vvy  0.0 $YMAX :CLIPY mirrorclip
```

```
 $VZ $Z0 :vvz multiaddt $vvz  0.0 $ZMAX :CLIPZ mirrorclip
 $CLIPX $CLIPY $CLIPZ gtranslate $ROTATE_RATE 0.0 :rotz multiaddt
   $rotz rotatez $tfile :textureindex gettextureindex $textureindex $D drawtsphere

OBJECTDEF // object for dynamic textured sphere
 ttest
 :texture_file
 $texture_file :textureindex gettextureindex $textureindex 1.0 drawtsphere

OBJECTDEF // object to test IF, LE, GT and AND operators
 conditional_ball
 :DX :DY :DZ         // parameters passed from FDS in SMOKEVIEW_PARAMETERS array
 :time gett          // get the current time
 $time 3.0 :LE_L LE  // is time .le. 3
 $time 3.0 :GE_L GT  // is time .gt. 3
 $time 6.0 :LE_H LE  // is time .le. 6
 $time 6.0 :GT_H GT  // is time .gt  6
 $LE_L IF
   "RED" setcolor   // set the color to red if t .le. 3.0
 ENDIF
 $GE_L $LE_H :ANDTEST AND $ANDTEST IF
   "GREEN" setcolor   // set the color to green if t .gt. 3.0 and t .le.6
 ENDIF
 $GT_H IF
    "BLUE" setcolor  // set the color to blue if t > 6.0
 ENDIF
 $DX $DY $DZ scalexyz 1.0 drawsphere

OBJECTDEF
 fan
  :HUB_R=0 :HUB_G=0 :HUB_B=0 :HUB_D=0.1 :HUB_L=0.12
  :BLADE_R=128 :BLADE_G=64 :BLADE_B=32 :BLADE_ANGLE=30.0 :BLADE_D=0.5 :BLADE_H=0.09
  :ROTATION_RATE=360.0
 $HUB_L -2.0 :HUB_LD2 div
 $BLADE_H -2.0 :BLADE_HD2 div
 $HUB_R $HUB_G $HUB_B setrgb
 $ROTATION_RATE 0.0 :rotz multiaddt $rotz rotatez
 push
   0.0 0.0 $HUB_LD2 translate
   $HUB_D $HUB_L drawdisk
 pop
 push
   $BLADE_R $BLADE_G $BLADE_B setrgb
   0.0 0.0 $BLADE_HD2 translate
   $BLADE_ANGLE $BLADE_D $BLADE_H drawarcdisk
 pop
 push
  120.0 rotatez
  0.0 0.0 $BLADE_HD2 translate
  $BLADE_ANGLE $BLADE_D $BLADE_H drawarcdisk
 pop
 push
  240.0 rotatez
  0.0 0.0 $BLADE_HD2 translate
  $BLADE_ANGLE $BLADE_D $BLADE_H drawarcdisk
 pop

OBJECTDEF
 vent
```

```
  :R=0 :G=0 :B=0 :W :H :ROT=0.0
 $ROT rotatez $R $G $B setrgb
 $W $H 1.0 scalexyz 1.0 drawsquare
 NEWFRAME
  :R=0 :G=0 :B=0 :W :H :ROT=0.0
 $ROT rotatez $R $G $B setrgb
 $W $H drawvent

OBJECTDEF
 cone
  :R=0 :G=0 :B=0 :D=0.4 :H=0.6
 $R $G $B setrgb
 $D $H drawcone

OBJECTDEF // object for dynamic textured sphere
 tsphere
  :R=0 :G=0 :B=0 :AX0 :ELEV0 :ROT0 :ROTATION_RATE :D=0.1 :tfile
 $R $G $B setrgb
 90.0 rotatey $AX0 rotatex $ELEV0 rotatey
 $ROTATION_RATE $ROT0 :rotz multiaddt $rotz rotatez
   $tfile :textureindex gettextureindex $textureindex $D drawtsphere

   // ************ dynamic tree object definitions ********************

OBJECTDEF // object for tree trunk
 TREE
 :TRUNK_D :TRUNK_H :TRUNK_BASE_H   // trunk variables
 :TRUNK_R=138 :TRUNK_G=69 :TRUNK_B=18
 :CANOPY_D :CANOPY_H :CANOPY_BASE_H // canopy variables
 :CANOPY_R=25 :CANOPY_G=128 :CANOPY_B=0
 $TRUNK_R $TRUNK_G $TRUNK_B setrgb
 push 0.0 0.0 $TRUNK_BASE_H translate $TRUNK_D $TRUNK_H drawdisk pop
$CANOPY_R $CANOPY_G $CANOPY_B setrgb
 0.0 0.0 $CANOPY_BASE_H translate $CANOPY_D $CANOPY_H drawcone

OBJECTDEF // object for tree trunk
 TRUNK
 :TRUNK_BASE_H :TRUNK_D :TRUNK_H :R=138 :G=69 :B=18
 $R $G $B setrgb
 0.0 0.0 $TRUNK_BASE_H translate $TRUNK_D $TRUNK_H drawdisk
NEWFRAME
 :TRUNK_BASE_H :TRUNK_D :TRUNK_H
 0.0 0.0 0.0 setrgb
 0.0 0.0 $TRUNK_BASE_H translate $TRUNK_D $TRUNK_H drawdisk
NEWFRAME
 :TRUNK_BASE_H :TRUNK_D :TRUNK_H
 0.0 0.0 0.0 setrgb
 0.0 0.0 $TRUNK_BASE_H translate $TRUNK_D $TRUNK_H drawdisk
NEWFRAME
 0.0 0.0 0.0 translate

OBJECTDEF // object for tree canopy
 CANOPY
:CANOPY_BASE_H :CANOPY_D :CANOPY_H :R=25 :G=128 :B=0
$R $G $B setrgbval
 0.0 0.0 $CANOPY_BASE_H translate $CANOPY_D $CANOPY_H drawcone
NEWFRAME
:CANOPY_BASE_H :CANOPY_D :CANOPY_H
0.0 0.0 0.0 setrgbval
```

```
 0.0 0.0 $CANOPY_BASE_H translate $CANOPY_D $CANOPY_H drawcone
NEWFRAME
 0.0 0.0 0.0 translate
NEWFRAME
 0.0 0.0 0.0 translate

OBJECTDEF // object for house
 HOUSE
 :R=0 :G=255 :B=0 :LENGTH=0.2 :DEPTH=0.2 :HEIGHT1=0.1 :HEIGHT2=0.05 :ANGLEZ=0.0
$R $G $B setrgb
$ANGLEZ rotatez $LENGTH $DEPTH $HEIGHT1 scalexyz 0.0 0.0 0.5 translate 1.0 drawcubec
    -.5 -0.5 0.5 translate 1.0 1.0 drawtriblock

OBJECTDEF // object for house
 MHOUSE
 :LENGTH=0.2 :DEPTH=0.2 :HEIGHT1=0.1 :HEIGHT2=0.05
255 0 0 setrgb
$LENGTH $DEPTH $HEIGHT1 scalexyz 0.0 0.0 0.5 translate 1.0 drawcubec -.5 -0.5 0.5
    translate 1.0 1.0 drawtriblock
 NEWFRAME
 :LENGTH=0.2 :DEPTH=0.2 :HEIGHT1=0.1 :HEIGHT2=0.05
0 255 0 setrgb
$LENGTH $DEPTH $HEIGHT1 scalexyz 0.0 0.0 0.5 translate 1.0 drawcubec -.5 -0.5 0.5
    translate 1.0 1.0 drawtriblock
 NEWFRAME
 :LENGTH=0.2 :DEPTH=0.2 :HEIGHT1=0.1 :HEIGHT2=0.05
0 0 0 setrgb
$LENGTH $DEPTH $HEIGHT1 scalexyz 0.0 0.0 0.5 translate 1.0 drawcubec -.5 -0.5 0.5
    translate 1.0 1.0 drawtriblock
 NEWFRAME
 :LENGTH=0.2 :DEPTH=0.2 :HEIGHT1=0.1 :HEIGHT2=0.05
0 0 0 setrgb

OBJECTDEF // object for tree canopy
 MCANOPY
 :CANOPY_BASE_H :CANOPY_D :CANOPY_H
25 128 0 setrgb
 0.0 0.0 $CANOPY_BASE_H translate $CANOPY_D $CANOPY_H drawcone
 NEWFRAME
 :CANOPY_BASE_H :CANOPY_D :CANOPY_H
153 51 0 setrgb
 0.0 0.0 $CANOPY_BASE_H translate $CANOPY_D $CANOPY_H drawcone
 NEWFRAME
 :CANOPY_BASE_H :CANOPY_D :CANOPY_H
153 153 153 setrgb
 0.0 0.0 $CANOPY_BASE_H translate $CANOPY_D $CANOPY_H drawcone
 NEWFRAME
 :CANOPY_BASE_H :CANOPY_D :CANOPY_H
26 26 26 setrgb
 0.0 0.0 $CANOPY_BASE_H translate $CANOPY_D $CANOPY_H drawcone
 NEWFRAME
 :CANOPY_BASE_H :CANOPY_D :CANOPY_H
0 0 0 setrgb

   // ************ avatar object definitions ********************

AVATARDEF
 human_fixed          // label, name of avatar
 :DUM1 :DUM2 :DUM3 :W :D=0.1 :H1 :SX :SY :SZ :R=0 :G=0 :B=0 :HX :HY :HZ
```

```
 90.0 rotatez
 "TAN" setcolor // head color  TAN 210 180 140
 0.3 0.3 0.3 scalexyz
 0.0 0.0 0.0 translate
 push  0.0 0.0 5.2 translate 1.1 drawsphere
   "BLUE" setcolor // eye color BLUE
   push -0.25 -0.4 0.05 translate 0.2 drawsphere pop // eye
   push  0.25 -0.4 0.05 translate 0.2 drawsphere pop // eye
   pop // head
 28 64 140 setrgb  // body color
 push  0.0 0.0 3.55 translate 0.5 0.3 1.0 scalexyz 2.5 drawsphere pop // trunk
 "TAN" setcolor // arm color TAN 210 180 140
 push -0.9 0.0 3.5 translate  35.0 rotatey 0.2  0.2  1.0 scalexyz 3.0 drawsphere pop
     // arm
 push  0.9 0.0 3.5 translate -35.0 rotatey 0.2  0.2  1.0 scalexyz 3.0 drawsphere pop
     // arm
 39 64 139 setrgb // leg color ROYAL BLUE4: 39 64 139
 push -0.5 0.0 1.3 translate  30.0 rotatey 0.25 0.25 1.0 scalexyz 3.0 drawsphere pop
     // leg
 push  0.5 0.0 1.3 translate -30.0 rotatey 0.25 0.25 1.0 scalexyz 3.0 drawsphere pop
     // leg

AVATARDEF
 human_altered_with_data         // label, name of avatar
 :DUM1 :DUM2 :DUM3 :W :D=0.1 :H1 :SX :SY :SZ :R=0 :G=0 :B=0 :HX :HY :HZ
 90.0 rotatez
 "TAN" setcolor // head color  TAN 210 180 140
 $SX $SY $SZ scalexyz //  scale by data height
 1.0 1.0 0.579 scalexyz
 0.3 0.3 0.3 scalexyz
 push  0.0 0.0 5.2 translate 1.1 drawsphere
   "BLUE" setcolor // eye color BLUE
   push -0.25 -0.4 0.05 translate 0.2 drawsphere pop // eye
   push  0.25 -0.4 0.05 translate 0.2 drawsphere pop // eye
   pop // head
 $R $G $B setrgb  // body color
 push  0.0 0.0 3.55 translate $W $D $H1 scalexyz 1.334 1.33 1.0 scalexyz 2.5
     drawsphere pop // trunk, scale by width and depth
 "TAN" setcolor // arm color TAN 210 180 140
 push -0.9 0.0 3.5 translate  35.0 rotatey 0.2  0.2  1.0 scalexyz 3.0 drawsphere pop
     // arm
 push  0.9 0.0 3.5 translate -35.0 rotatey 0.2  0.2  1.0 scalexyz 3.0 drawsphere pop
     // arm
 39 64 139 setrgb // leg color ROYAL BLUE4: 39 64 139
 push -0.5 0.0 1.3 translate  30.0 rotatey 0.25 0.25 1.0 scalexyz 3.0 drawsphere pop
     // leg
 push  0.5 0.0 1.3 translate -30.0 rotatey 0.25 0.25 1.0 scalexyz 3.0 drawsphere pop
     // leg

AVATARDEF
 ellipsoid         // label, name of object
 :DUM1 :DUM2 :DUM3 :W :D=0.1 :H1 :SX :SY :SZ :R=0 :G=0 :B=0 :HX :HY :HZ
 90.0 rotatez
 $HX $HY $HZ translate $SX $SY $SZ scalexyz $W $D $H1 scalexyz
  push 0.0 -1.0 0.0 translate 1.0 5.0 0.5 scalexyz "BLUE" setcolor 0.4 drawsphere pop
  $R $G $B setrgb 1.0 drawsphere

AVATARDEF
 disk         // label, name of object
```

```
 :DUM1 :DUM2 :DUM3 :W :D=0.1 :H1 :SX :SY :SZ :R=0 :G=0 :B=0 :HX :HY :HZ
 90.0 rotatez
 0.0 0.0 1.0 translate $W $D $H1 scalexyz
  push 0.0 -0.25 0.05 translate 0.3 2.5 0.3 scalexyz "CYAN" setcolor 0.2 drawsphere
     pop
  $R $G $B setrgb 1.0 0.05 drawdisk

AVATARDEF
 fire_fighter
 "TAN" setcolor // head color  TAN 210 180 140
 0.3 0.3 0.3 scalexyz
 0.0 0.0 0.0 translate
 push  0.0 0.0 5.2 translate 1.1 drawsphere
   "BLUE" setcolor // eye color BLUE
   push -0.25 -0.4 0.05 translate 0.2 drawsphere pop // eye
   push  0.25 -0.4 0.05 translate 0.2 drawsphere pop // eye
   pop // head
 "YELLOW" setcolor  // body color
 push  0.0 0.0 3.55 translate 0.5 0.3 1.0 scalexyz 2.5 drawsphere pop // trunk
 "YELLOW" setcolor // arm color
 push -0.9 0.0 3.5 translate  35.0 rotatey 0.2  0.2  1.0 scalexyz 3.0 drawsphere pop
    // arm
 push  0.9 0.0 3.5 translate -35.0 rotatey 0.2  0.2  1.0 scalexyz 3.0 drawsphere pop
    // arm
 "BLUE" setcolor // leg color
 push -0.5 0.0 1.3 translate  30.0 rotatey 0.25 0.25 1.0 scalexyz 3.0 drawsphere pop
    // leg
 push  0.5 0.0 1.3 translate -30.0 rotatey 0.25 0.25 1.0 scalexyz 3.0 drawsphere pop
    // leg

OBJECTDEF
 airpack
 96 96 96 setrgb
 push 180.0 rotatey 0.2 drawhsphere pop
 0.2 0.55 drawdisk
 0.0 0.0 0.55 translate 0.2 drawhsphere
OBJECTDEF  // used by smokeview to display smoke thickness
 helmit
 255 51 51 setrgb
 0.3 drawhsphere
 0.4 0.02 drawdisk
AVATARDEF
 fire_fighter_with_gear          // label, name of avatar
 push "fire_fighter" include pop
 push 0.0 0.0 1.65 translate 1.0 1.0 1.0 scalexyz "helmit" include pop
 push 0.0 0.2 0.80 translate "airpack" include pop

   // ************ Elementary object definitions *******************

   //  These definitions are used to illustrate the basic building blocks
   //  used to create more complex objects

OBJECTDEF
 drawaxisxyz
 "RED" setcolor
 push  90.0 rotatey 0.05 0.05 0.4 scalexyz 1.0 1.0 drawdisk pop
 "GREEN" setcolor
 push -90.0 rotatex 0.05 0.05 0.4 scalexyz 1.0 1.0 drawdisk pop
 "BLUE" setcolor
```

```
 push 0.05 0.05 0.4 scalexyz 1.0 1.0 drawdisk pop

OBJECTDEF
 drawaxis2
 "BLACK" setcolor
 push 0.05 0.05 0.4 scalexyz 1.0 1.0 drawdisk pop
  "BLACK" setcolor
 push 90.0 rotatey 0.05 0.05 0.6 scalexyz 1.0 1.0 drawdisk pop

OBJECTDEF
 drawaxis
 "BLACK" setcolor
 push 0.00625 0.00625 0.075 scalexyz 1.0 1.0 drawdisk pop
 "BLACK" setcolor
 push 90.0 rotatey 0.00625 0.00625 0.075 scalexyz 1.0 1.0 drawdisk pop

OBJECTDEF
 drawcone
 "BRICK" setcolor
 0.50 0.30 drawcone

OBJECTDEF
 drawcube
 "BRICK" setcolor
 0.25 drawcube

OBJECTDEF
 drawdisk
 "BRICK" setcolor
 0.25 0.50 drawdisk

OBJECTDEF
 drawcdisk
 "BRICK" setcolor
 0.25 0.50 drawcdisk

OBJECTDEF
 drawhexdisk
 "BRICK" setcolor
 0.50 0.25 drawhexdisk

OBJECTDEF
 drawnotchplate
 "BRICK" setcolor
 0.5 0.1 0.2 1 drawnotchplate

OBJECTDEF
 drawnotchplate2
 "BRICK" setcolor
 0.5 0.1 0.2 -1 drawnotchplate

OBJECTDEF
 drawpolydisk
 "BRICK" setcolor
 5 0.35 0.15 drawpolydisk

OBJECTDEF
 drawring
 "BRICK" setcolor
```

```
 0.3 0.5 0.1 drawring

OBJECTDEF
 drawsphere
 "BRICK" setcolor
 0.25 drawsphere

OBJECTDEF
 drawtrunccone
 "BRICK" setcolor
 0.5 0.2 0.4 drawtrunccone

OBJECTDEF
 drawarcdisk
 "BRICK" setcolor
 60.0 0.6 0.2 drawarcdisk
```