# Firefighter Command Training Virtual Environment

Tazama U. St. Julien
stjulien@cc.gatech.edu

Chris D. Shaw
cdshaw@cc.gatech.edu

GVU Center, Georgia Tech
801 Atlantic Dr, Atlanta, GA, 30332

## Abstract

*The Firefighter Command Training Virtual Environment is being developed at Georgia Tech in collaboration with the Atlanta Fire Department. The VE allows the user to: navigate around the environment, viewing a house on fire from any angle; command firefighters and watch them execute those commands; and see realistic fire and smoke behavior reacting to changes in the environment. The VE user is a commanding officer trainee who instructs teams of virtual firefighters to perform different actions to help put out virtual fires. The correct sequence of commands will successfully extinguish the flame with the least amount of danger to the firefighters and the least amount of damage to the home. This simulation was developed using the Simple Virtual Environment (SVE) library, an extensible framework for building VE applications. This is the first example of a firefighter training environment that combines representations of animated firefighters with a reasonable simulation and animation of smoke and fire.*

## 1. Introduction

The Fire Company Officer is the person in command of a 4 to 8 person company of firefighters who are dispatched to a fire emergency. A typical Fire Company Officer has a number of years of experience as a firefighter and will probably have experienced most of the common types of emergencies that firefighters are required to respond to. Aside from on-the-job experience, a Fire Company Officer (FCO) will train for his/her job by attending classroom instruction [Frie72] and by practicing command procedure at the Fire Department's training ground.

There are a number of drawbacks to these methods of training for command. First, not all fire companies see all types of emergencies in equal measure on the job. For example, only 3% of calls to the Atlanta Fire Department are for fires [Mino01]. Second, practicing command on the fire training ground will always take place in exactly the same building. Third, unlike most houses, the training building is fireproof, which means that the commander doesn't have to worry or think about structural issues. Finally, burning materials have an impact in terms of air pollution, cost of materials, and maintaining the safety of the crew.

Promotion and evaluation exams for FCOs may consist of pencil and paper tests followed by an in-depth command interview in which the trainee verbally takes the examiners through the procedure that he/she would use in a real fire [Mino01]. To help solve these problems, we have built a prototype Firefighter Command Training Environment in collaboration with the Atlanta Fire Department. A virtual environment (VE) can provide a variety of firefighting scenarios for instruction and evaluation in a more realistic manner than verbal or written material and with less risk and expense than training with real fires. The Firefighter Command Training Virtual Environment will aid in the training and testing of FCOs.

The VE user is a FCO trainee who instructs teams of virtual firefighters to perform different actions to help put out virtual fires. The VE allows the trainee to navigate around the environment and view the situation from any angle. The trainee can also command firefighters, watch them execute those commands, and see realistic fire and smoke whose behavior changes in reaction to changes in the environment. The correct sequence of commands will successfully extinguish the flame with the least amount of danger to the firefighters and the least amount of damage to the home.

The Firefighter Command Training Virtual Environment can be separated into a few distinct tasks: inputting commands into the VE, graphically displaying the environment and the firefighters executing commands in that environment, and visualizing physically accurate fire and smoke. We are using the Simple Virtual Environment (SVE) [Kess00] library to build our VE, and NIST's Fire Dynamic Simulator (FDS) to simulate accurate fire and smoke behavior. In this paper, we will give a system overview, then discuss in detail the Graphical Command Interface, how we manage the execution of tasks by the firefighters, and the realistic visualization of fire and smoke.

## 2. System Overview

Our system consists of three main components; the Virtual Environment, the Graphical Command Interface, and the NIST Fire Dynamic Simulator (see Figure 1). We are using the Simple Virtual Environments (SVE) library [Kess 00] as the backbone of our application.

The Firefighter Training Virtual Environment includes a furnished single family home, a fire truck, a fire hydrant, various firefighting tools, and firefighters. To put out the fire, the trainee issues commands to the virtual firefighters. The command entry into the system is performed by an operator who translates the verbal command from the trainee into commands on the Graphical Command Interface. The firemen then execute the appropriate character animation for their assigned task. In order for our virtual environment to display fire and smoke, we

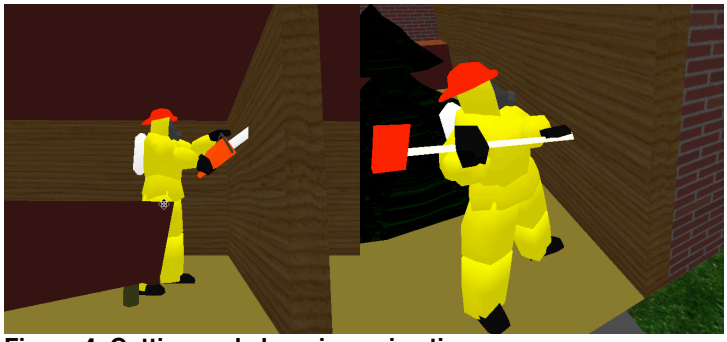

**Figure 4. Cutting and chopping animations**

precompute fire and smoke behavior using the NIST Fire Dynamic Simulator (FDS). The time varying behavior of the smoke and fire is stored in a series of data files for use by the virtual environment.

## 3. Graphical Command Interface

The Graphical Command Interface (GCI) for our virtual environment reduces input errors, provides fast and easy entry of commands into the system, and allows the operator to monitor the status of the fire teams in the VE. Before the GCI was implemented, command entry was done using typed instructions. This type of interface requires user knowledge of all possible commands, is very error prone, and makes it difficult to understand the state of the virtual environment and what commands have been issued. The GCI is a standalone application that uses TCP/IP sockets to send messages to the virtual environment and receive status messages from the VE.

## 4. Firefighter Control

Firefighter command execution can be broken down into three parts. A path from the firefighter's current position to their destination point that avoids walls and other obstacles must be found. The firefighter must also be translated at a set velocity along the path while executing an appropriate looped character travel animation (e.g. walking, carrying a ladder, or crawling). The firefighter will then perform the commanded task at the destination point with a looped task animation.

### 4.1. Pathfinding

We are currently using the A* pathfinding algorithm to plan the path of the firefighter [Russ95]. By using this technique, the firefighters can successfully navigate around walls and other obstacles. A* is a heuristic search that ranks each node by an estimate of the best route that traverses through that node. A* is guaranteed to find the shortest path, as long as the heuristic estimate, h(n), is acceptable, that is, it is never greater than the true remaining distance to the goal. It makes the most efficient use of the heuristic function of all other pathfinding algorithms. However, the quality of A* search depends on the quality of the heuristic estimate. If h(n) is very close to the true cost of the remaining path, its efficiency will be high.

### 4.2. Models and Animations

Using 3D Studio Max, we have modeled the house, the furnishings in the house, the fire truck, the fire hydrant, fire hoses, the firefighters, and their firefighting tools. We export our models as Wavefront files. A motion vocabulary of character animations is created in 3D Studio Max and used in the virtual environment.

To complete their assigned tasks, firefighters perform these character animations. The vocabulary of character animations includes: cutting, chopping, walking, crawling, climbing, pulling and spraying a hose, and carrying a variety of tools (See figures 4-6). By performing the correct combinations of these tasks the firefighter can successfully put out the fire.

In order to simplify the animation process, we have created an animation file format that can be read by a parser and used to perform the translations and rotations required for these animations. We use a custom 3D Studio Max plug-in that outputs animation files. These animation files contain absolute position changes and relative rotation changes for each key frame of the firefighter objects. The animation parser uses these values to create the animations we see in our virtual environment. The parser interpolates the transformations of each object over time and makes the appropriate small change to the object every frame.

## 5. Realistic Fire and Smoke

In order for our virtual environment to be effective in training and testing FCOs, there must be a realistic visual representation of fire and smoke. Command trainees have prior experience and expectations for the behavior of fire and smoke, and the progress of fire and smoke in a burning building affects the decisions they should make. We face four main problems in visualizing fire and smoke. Where do we get the fire and smoke data, how do we deal with the sheer volume of that data, how do we visualize that data, and how do we make the fire and smoke react to changes in the environment?

### 5.1 Fire Simulation

We are using NIST's Fire Dynamic Simulator (FDS) to compute realistic physical fire and smoke behavior and output volumetric data inside of our house. We use volume rendering techniques to render this volumetric fire and smoke data. This allows our environment to display fire and smoke with very realistic behavior. FDS predicts smoke and/or air flow movement caused by fire, wind, ventilation systems and other physical factors. It is a computational fluid dynamics model of fire-driven fluid flow, and numerically solves the Navier-Stokes equations appropriate for low speed, thermally driven airflow with an emphasis on smoke and heat transport from fires. We model the house offline with FDS and compute fire and smoke data for the entire house volume at one-second increments. We use this pre-computed data to visualize and animate the fire and smoke in our virtual environment.

We have created a script that will read in a Wavefront object file and output the house model in FDS input data file format. Once the house is modeled in FDS, we lay out a timeline for the simulation to follow, and add parameters for the fire, and the hoses. A Plot3D file is created by FDS for every second of data and contains computed values for each voxel in the houses volume; the heat release rate per unit volume (hrrpuv), and the soot density. On a 1GHz PC

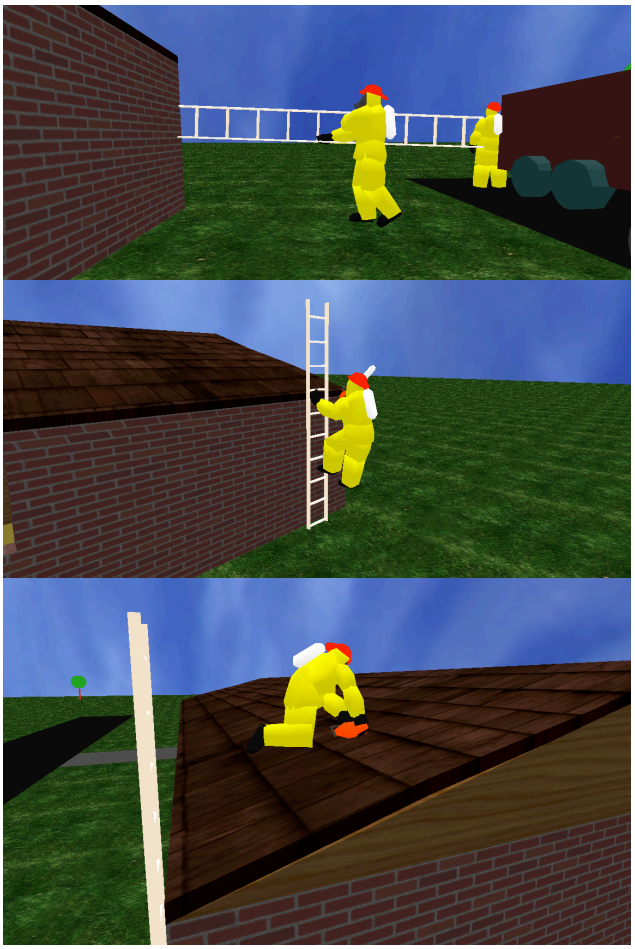

**Figure 6. Ladder carry, climb, and roof cut animations**

with 512 MB of RAM, FDS takes about 8 hours to compute one minute of data.

With FDS, there is a tradeoff between processing time and the accuracy and detail of the simulation. As the grid's voxel size decreases, processing time increases but the simulation becomes more detailed and accurate. Increasing the voxel size creates an inaccurate simulation due to heat loss. Our voxel size is 20 $cm^3$ and is the result of a compromise between the detail of the fire and smoke simulation and processing time. Currently our house's dimensions are 165 x 80 x 20 and contains 264,000 voxels. Plot3D files contain 5 binary floating-point values for each voxel. Because the voxel grid is regular, there is no need to store voxel position information. Therefore, each file, representing one second of data, is about 5.3 MB. This is too large to be used efficiently by a real-time application. Read times for these data files range from 0.25 to 1.3 seconds. Of the 5 values in the data files, we are only interested in 2 (about 2 MB). Therefore, to minimize storage requirements and to improve run-time performance for volumetric fire and smoke visualization, the data files are compressed.

## 5.2 Volume Compression

Run Length Encoding (RLE) compression is used to compress our data files to a manageable size. First, we quantize the smoke density and hrrpuv values of the data file to 8 bits per voxel, then we use RLE compression on the quantized data. By compressing the data we save disk space and improve the performance of our simulation. Using this compression technique we can reduce the file sizes from about 2MB to 160KB per data file, and the compressed files takes less time to read and decompress than the uncompressed files took to read. Input times for the compressed data range from about 0.05 to 0.08 seconds versus the 0.25 to 1.3 seconds required for uncompressed data.

## 5.3 Fire and Smoke Visualization

To draw the fire and smoke, we are implementing a voxel-based splatting renderer [West90] similar to Jang [Jang 02]. To do this, we use the values from the FDS output file for each time step to draw fire and smoke. For each voxel in the house's volume where smoke or fire is present, we draw a billboarded polygon with a gaussian texture (Figure 7) of the appropriate color texture mapped onto it. The opacity and color of each screen-aligned smoke and fire splat is dependent upon the soot density value and heat release rate per unit volume of each voxel respectively. Currently we traverse through each voxel of the house and analyze its values. If its hrrpuv is above 200 $^{kW}/_{m^3}$, we draw a fire splat of the appropriate color. Otherwise, we analyze its soot density. If it is above 90 $^{mg}/_{m^3}$, we will draw smoke at its appropriate opacity and color. The larger the soot density, the darker and less transparent the smoke will be. Therefore, we have created a ramping function that takes the soot density as input and outputs the appropriate color and opacity for the splat. Smoke splats range from 10% opacity and light gray to 75% opacity and completely black. We also map hrrpuv to a range of bright fire colors, from red to orange to yellow (see Figure 8).

**Figure 7. Gaussian Texture**

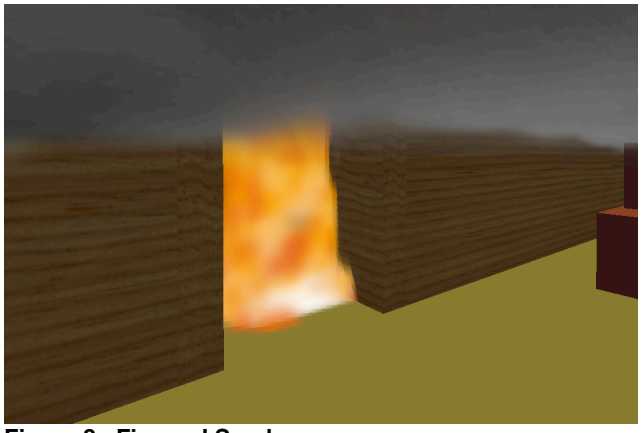

**Figure 8. Fire and Smoke**

In order to render our gaussian splats, we are using a per-frame callback function imbedded in a single SVE object. This callback function is executed when that object is traversed in the scene graph. This allows for the use of direct openGL calls to draw all of our splats in one SVE object. This technique results in improved performance.

Table 1 lists performance times for rendering fire and smoke on a PC with a 1GHz Intel Pentium III processor, 512 MB of RAM and a nVidia GeForce 3 video card. (With a 2.4 GHz Pentium 4 and GeForce 4, the virtual environment updates at 8 frames per second with 50,591 splats. We plan to furthor optimize volume rendering with large volumes of smoke.)

**Table 1. Performance measures for rendering splats**

| Time into simulation | Splat Count | Frames/Sec Display List | Frames/Sec No Disp. List |
|---|---|---|---|
| 15 seconds | 1,055 | 62.11 | 19.37 |
| 36 seconds | 5,504 | 28.41 | 12.02 |
| 50 seconds | 10,591 | 17.21 | 9.18 |
| 152 seconds | 50,591 | 4.12 | 2.84 |

### 5.4 Interactive Fire and Smoke

For our virtual environment to be interactive, we must have fire and smoke data available for every sequence of actions that the firefighters can perform. When a firefighter sprays water on the fire, the blaze should subside and more smoke/steam should be produced. When a door is opened, the fire should react accordingly. Therefore, we must use FDS to precompute every possible scenario encountered in the virtual environment. The trainee will be able to make interactive choices that will affect the behavior of the fire and smoke in 15-second increments.

We are currently in the process of creating a decision tree that loads the appropriate precomputed FDS data corresponding to the sequence of choices the trainee makes. We monitor the state of the house and the effects of the user's choices on the state of the house, so that every 15 seconds a new group of data files is selected and used to visualize the fire and smoke. This way, the fire and smoke reacts realistically and becomes a visual clue to the trainee of the house's current state.

## 6. Conclusion

In this paper we discussed the Firefighter Command Training Virtual Environment, which allows a fire company officer trainee to navigate around the environment and view a house on fire. The virtual environment presents a volumetrically rendered CFD simulation of fire and smoke in combination with virtual firefighters. The trainee commands these firefighters to take actions to put out the fire. This is the first example of a firefighter training environment that combines representations of animated firefighters with a reasonable simulation and animation of smoke and fire.

## 7. Future Work

The Firefighter Training Virtual Environment is a prototype and is still in development. We are currently working on improving some aspects of the application and adding new features.

We are refining our implementation of pathfinding. We plan to segment the houses grid into connected rooms in order to perform a high-level search with each node representing an entire room. After finding the path of rooms, low-level searches can be performed on each room. We expect this will improve the performance of our pathfinding algorithm.

Currently, we can playback precomputed fire and smoke data, but our decision tree is too simple. We are in the process of building a richer decision tree structure.

## 8. Acknowledgements

We thank the Atlanta Fire Department for their support since June 2000, and the students that have contributed to this project over the years.

## 9. References

[Blis97] Bliss J. P., Tidwell P. D., Guest M. A., (1997). "The effectiveness of Virtual Reality for administering Spatial Navigation Training to Firefighters". Presence: *Teleoperators and Virtual Environments Journal*, Vol. 6, No. 1, February 1997, pp 73 - 86.

[Broo99] F. Brooks. "What's Real About Virtual Reality?" *IEEE CG&A*, Vol. 19, No. 6, 1999

[Buko97] R. Bukowski and C. Sequin. "Interactive simulation of fire in virtual building environments". *Proceedings of SIGGRAPH '97,* Los Angeles, 1997.

[Egse93] Egsegian, R., Pittman, K., Farmer, K., & Zobel, R. (1993). "Practical Applications of Virtual Reality to Firefighter Training". *In Proceedings of the 1993 Simulation Multiconference on the International Emergency Management and Engineering Conference*, Tenth Anniversary: Research and Applications, (pp. 155-160). San Diego, CA: SCS.

[Frie72] Emmanual Fried, *Fireground Tactics* Atlanta, Ga. : Argus, 1972.; 372 p. : ill. 9th printing, 1994

[Hall99] Haller M., Holm R., Volkert J., and Wagner R. "A VR based safety training system in a petroleum refinery", In Eurographics'99, 20th Annual Conference of the European Association for Computer Graphics, Sept. 7th-11th 1999, Milano.

[Jang02] J. Jang, W. Ribarsky, C. D. Shaw, and N. Faust. "View-Dependent Multiresolution Splatting of Non-Uniform Data". In *Proceedings of IEEE Visualization Symposium*, Barcelona, 2002.

[Kess00] D. Kessler, D. Bowman, L. Hodges. "The Simple Virtual Environment Library: An Extensible Framework for Building VE Applications", *Presence: Teleoperators and Virtual Environments*, 2000, pp. 187-208.

[Mino01] Chief Winston Minor, Atlanta Fire Departement, Personal Communication, 2001.

[Russ95] Stuart J. Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach* 932 pages, Prentice Hall, New York, 1995.

[Star02] http://www.startechcorp.com

[SWRI 02] http://www.tss.swri.edu/technology/vr_interactive_simulation/subsection_fire.shtm

[Tate97] D. Tate, L. Sibert, and T. King, "Virtual Environments for Shipboard Firefighting Training," *Proc. IEEE Virtual Reality Ann. Int'l Symp.*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1997, pp. 61-68.

[West90] L. Westover. "Footprint Evaluation for Volume Rendering". In Computer Graphics, *Proceedings of SIGGRAPH 90*, pages 367--376. August 1990.