VFire: Virtual Fire in Realistic Environments

Roger V. Hoang ◦ Joseph D. Mahsman ◦ David T. Brown ◦ Michael A. Penick ◦ Frederick C. Harris Jr. ◦ Timothy J. Brown

Department of Computer Science and Engineering, University of Nevada, Reno

Desert Research Institute, Center for Advanced Visualization, Computation and Modeling (CAVCaM), Reno, Nevada

Abstract

VFire is an immersive wildfire visualization application geared towards scientific analysis as well as training and land management. Given the large datasets involved in wildfire simulations, we adopt a GPU-centric approach to visualization. By taking advantage of the programmable shader and other features of modern graphics hardware, we increase the verisimilitude of our visualization and the range user interactivity while maintaining a real-time frame rate.

Introduction

The devastation left in the wakes of wildfires has fueled the development of mathematical models in order to better understand fire behavior. VFire provides an immersive visualization of FARSITE, a fire behavior simulator based on one such model. In addition to running on a desktop computer, the visualization is also capable of running in a CAVE environment. The long term goal of VFire is to allow researchers to verify and refine fire models and experiment with new fire mitigation and prevention techniques; additionally, VFire can be utilized for training as well as community planning.

In working towards these goals and target audiences, our recent efforts have been focused on improving the visual presentation of the simulation as well as increasing the degree of user interactivity. We achieve these goals without sacrificing system responsiveness by exploiting particular features of modern graphics hardware.

Terrain

Visualizing a wildfire typically involves rendering large expanses of land. We use a level-of-detail approach to achieve high rendering speeds without a noticeable loss in visual quality. With FARSITE data stored as vertex textures, a vertex shader is used to texture the terrain based on fuel load data and to generate a glow and scorching effect as an area is burned.

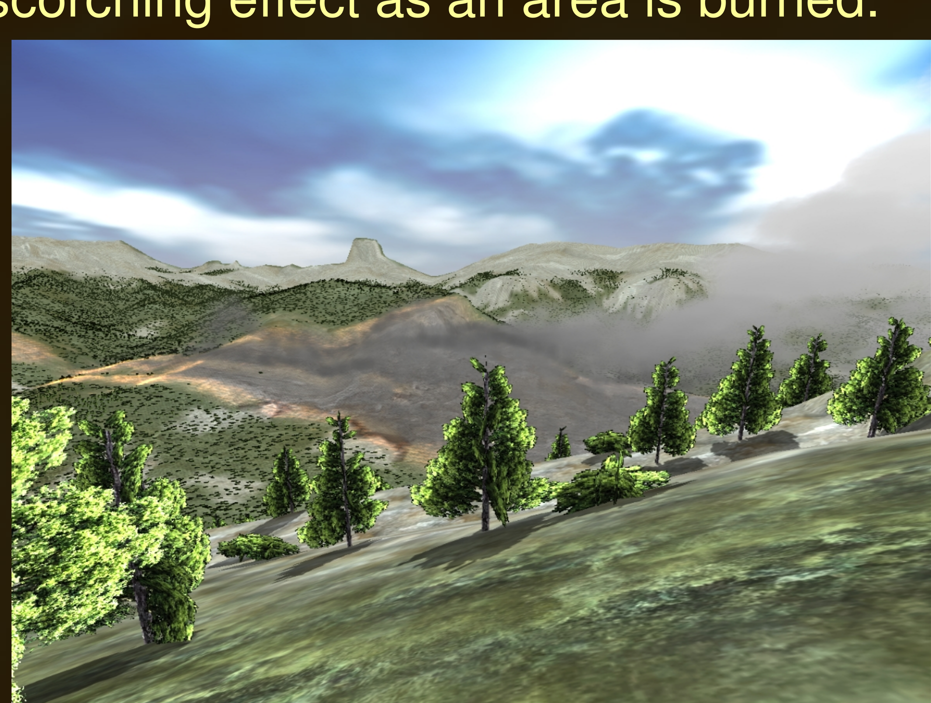

Figure 1. Terrain being scorched in our prototypical dataset of Kyle Canyon, Nevada.

Vegetation

Two interchangeable solutions are provided for vegetation rendering. The first consists of procedurally generated trees; burn effects are achieved by having a vertex shader determine how burned a vertex is by accessing ignition time data stored as a vertex attribute. The second solution uses SpeedTree. Burn effects are produced in a similar manner; however, the ignition times are determined by using a vertex texture.

Figure 2. SpeedTree vegetation with a burn effect applied at various times.

Fire and Smoke

Fire and smoke are visualized using particle systems. Particle data is stored and updated on the GPU, with lock-step updating and CPU-side particle emissions in order to maintain consistency between screens. Billboarding and animation are done in a vertex shader, with spread data skewing the fire particles. High dynamic range rendering is used to control the visual effects of the fire depending on the time of day.

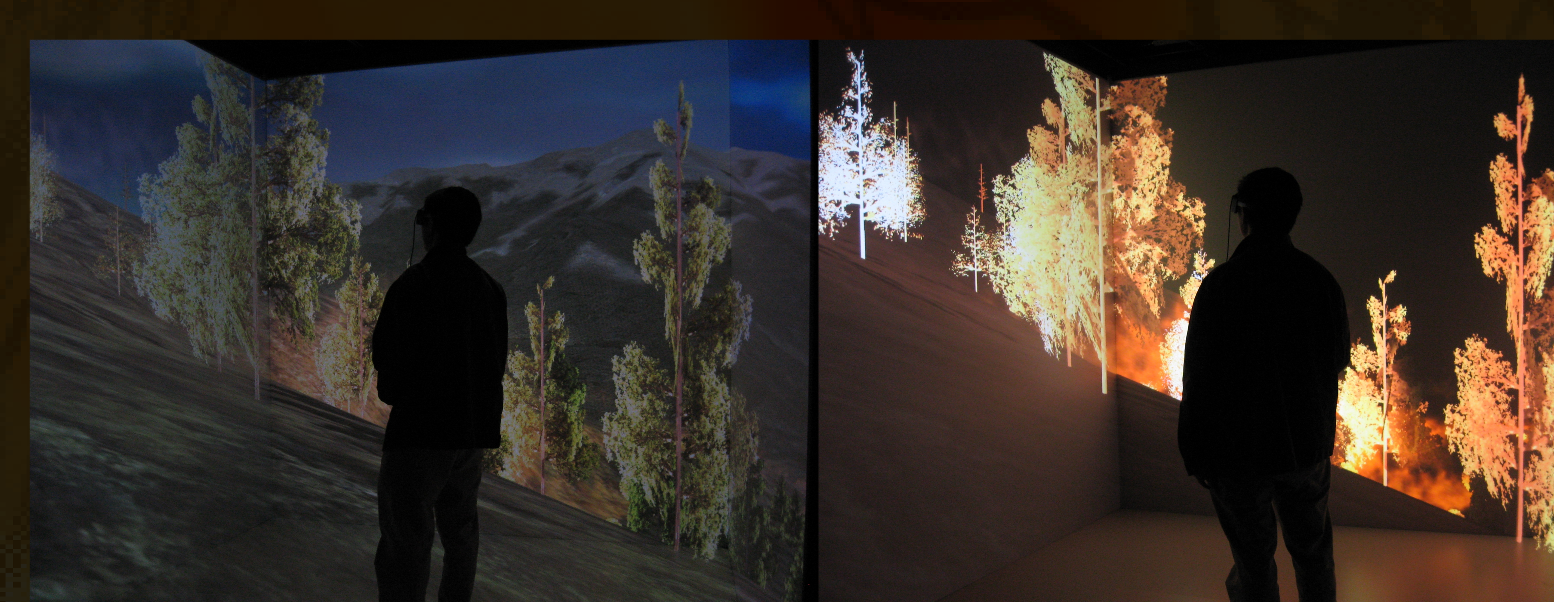

Figure 3. VFire in a CAVE environment using day (left) and night (right) lighting parameters.

Interaction

In addition to allowing the user to fly about the virtual world or trek on the terrain to gain different perspectives, VFire allows the user to manipulate the flow of time. This is achieved by deriving the state of a vertex purely as a function of simulation time; as such, time can be sped up, slowed down, stopped, and even reversed by shifting a single variable.

Figure 4. Time manipulation in VFire. The time multiplier is displayed at the top of the wheel while an hourglass indicates the direction of time.

Conclusions

VFire provides an immersive visualization of an established fire behavior model. In addition to being a model verification tool, VFire can reveal crucial aspects of fire behavior to analysts and visualizes data in such a way that it can be more easily understood by a broader audience.

By exploiting features of modern graphics hardware, we were able to increase the visual fidelity of our visualization without reducing interactivity; on the contrary, by creating a system where every vertex knows when and how to destroy itself, the user has been granted more control over the visualization.

Future Work

Improving the accuracy of the visualization will continue to be a main objective. We are developing ways to incorporate more data such as fireline intensity and wind into the visualization. Additionally, incorporating a physically-based smoke model will be integral for atmospheric analysis. To more closely replicate the real world, we are also working on the ability to extract vegetation and objects from satellite images in conjunction with other pieces of data using image processing techniques.

We also plan to integrate the FARSITE software directly into VFire in order to allow users to experiment with various parameters and increase VFire's utility as a training tool. With respect to this, the GPU-centric nature of VFire suggests pushing FARSITE directly onto the graphics hardware; graphics hardware now provides features that bring such a move closer to reality.

This work was partially supported by the US Army PEO-STRI, NAVAIR Contract N61339-04-C-0072