Developing Audio Utilities in High End Visualization

Adrian Ly   Yusuf Van Gieson   Steven G. Satterfield

Information Technology Lab, National Institute of Standards and Technology

Introduction

NIST uses a software system named High End Visualization (HEV) in a Computer Assisted Virtual Environment (CAVE) to create and interact with visualizations.

Image of NIST CAVE

The current hev-playAudioFiles utility plays audio files with no extra capabilities. The purpose of this project was to add new functions to the utility that would allow more audio manipulation and freedom.

New Capabilities

Blocking
Blocked audio files play in sequence; non-blocked audio files play simultaneously. Multiple fork commands were used to allow non-blocking while the wait command was used to allow blocking.

Channel Volume Control
A stereo audio file requires two output channels: the left and the right. By accessing the audio file data, the volume of each channel can be independently set to a user-defined percentage output of the current volume setting.

Fading
Audio fading is when one channel changes in volume while the other channel experiences the opposite change in volume. Depending whether the user wanted left-to-right or right-to-left, the audio data file was modified to reflect the increasing/decreasing volume of the channel as the audio file played.

Memory
The utility became more efficient by accessing audio file data from memory rather than from the file directory. Upon execution, audio files are accessed from the file directory and the necessary data are loaded into an array that is saved in memory.
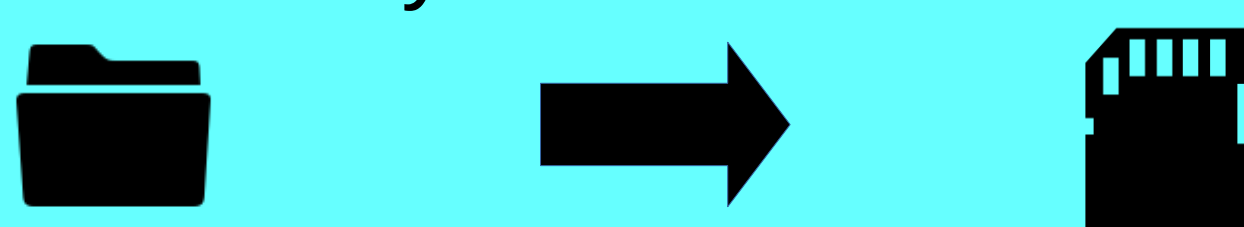
Significance

Humans observe the world with their eyes and ears. As visualization techniques improve, it is crucial to improve audio as well. Incorporating the audio with the visuals will help researchers study the data naturally.
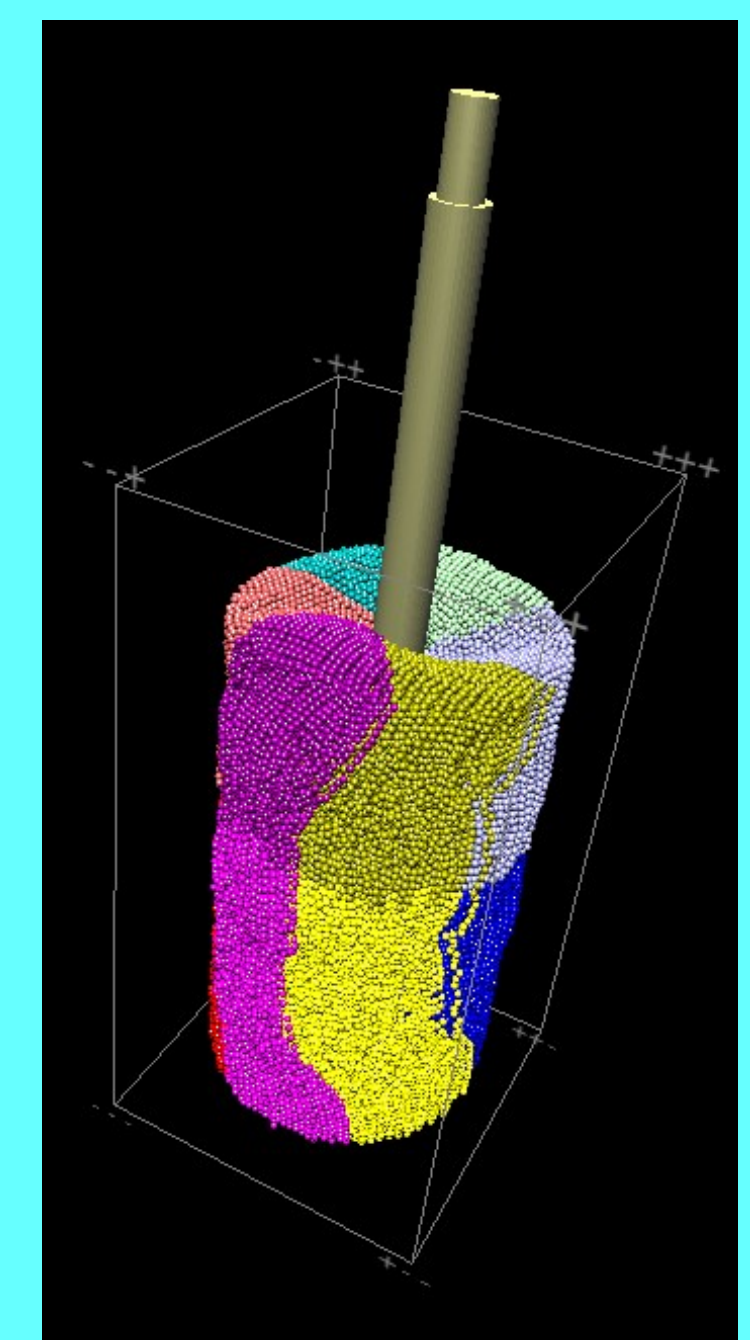
Conclusion

The improved hev-playAudioFiles utility includes multiple user-defined features with a convenient and accessible GUI that will allow the CAVE to produce more audio-sophisticated demos and increase the comprehensibility of the displayed data.

Testing

The new capabilities were tested with a rheometer demo. A stress value was associated with every frame in the animation. audioTest.c was written to convert the current frame number playing to access the stress value in the current frame into a line of arguments that is fed into the utility. Low pitch notes faded right-to-left represent low stress values and high pitch notes faded left-to-right represent high stress values.
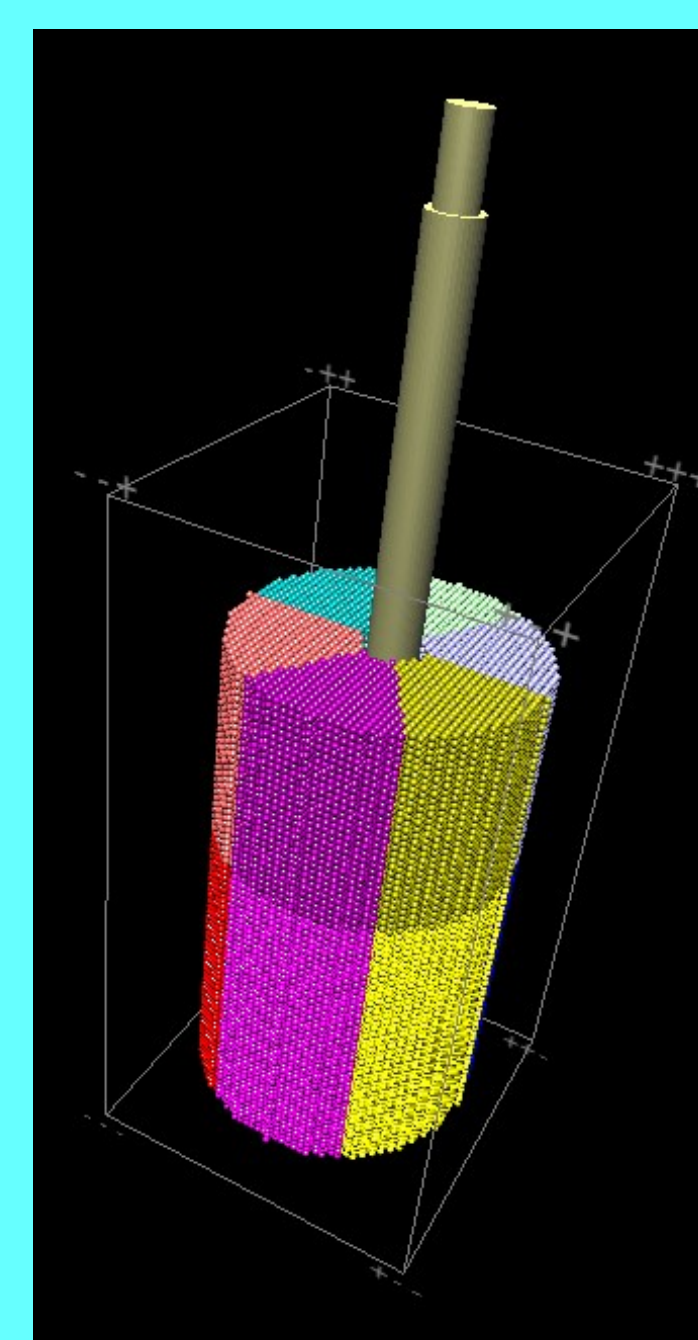
Initial rheometer state   Running rheometer state

hev-readInts: current animated frame in the running visualization
audioTest.c: note + volume/fading (input for hev-playAudioFiles)
hev-playAudioFiles: plays the audio file with the given argument$_s$