Volume Visualization Tool (PRISM)

As of January 10, 2019.

This document is to provide documentation as to how to use the Volume Visualizer tool (VVT). This README is written in markdown.

Scope and Software Versions

- I will only cover the usage of the desktop version of the tool. The VR and CAVE versions of the tool are deprecated currently. Several updates in Unity have led to what would likely amount to numerous shifts in how VR is handled in the VIVE version of the tool.
- I will only be covering the newest version of the VVT, which uses SIEVAS to stream the brick data.
- The PRISM and SIEVAS branches associated with this version are both called `VolViz_Stream.`
- For further details on some of the concepts and complications associated with the VVT, see VolumeVizTool.pdf. This is a discussion on the theory and issues surrounding the VVT.
- I used Unity 2018.2.1f1 (64 bit) to develop the VVT. The VVT (desktop) should be back compatible with older versions of Unity to a degree. The Frames Per Second prefab may or may not work on all versions of Unity.

- I used VisualStudio2017 for the C#/HLSL development.
- I used Netbeans 8.2 and Java 8 for the Java development.
- All development was done on Windows 10.
- I used Git Bash for my console.

Starting

Begin by opening Unity. We will be using the `VolumeVisualizationDesktop` project. In the Game view, you should see the SIEVAS login screen. Each field of the SIEVAS login screen should be auto populated. Depending on your usage, you may need to change these fields. To change these fields in Unity (and store them for further re-use), change

```
cvsLogin ->
          Panel ->
                 ifUsername -> (Or ifPassword or ifServerURL)
                 [Inspector]->
                             Input Field (Script) ->
                                                  Text
```

The default credentials are Username: `user` and Password: `password`. The name of the volume to render can be changed under `Main Camera -> Volume Controller (Script) -> Volume Name`. This field is not case sensitive. The `Data Path` field can also be altered as necessary.

Now start the backend to SIEVAS using the following Maven make command (It is assumed that SIEVAS has already been installed and set up properly):

```
[~SIEVAS/backend] $ mvn spring-boot:run
```

If all of your dependencies are compliant with the SIEVAS backend POM file, then you should see a line in the console that states "HELLO HANDLER" followed by about 20 lines of INFO. The last line of info should be something along the lines of

```
<timestamp>  INFO 10956 --- [  restartedMain] gov.inl.SIEVAS.Application:
  Started Application in 16.22 seconds (JVM running for 17.7)
```

If the make command fails due to an inability to create beans with the bean maker, the dependencies for SIEVAS and its backend may be out of date (or too up to date!). Fixing this can be rather tricky and is beyond the scope of this README. Every case is going to be slightly different.

Volume Visualizer Tool Controls

Assuming that the make command has executed successfully, you can now press "play" in Unity. SIEVAS should be set to auto login. The volume should now be rendered. Note that some volumes are surrounded by padding that obstructs the view of the volume until the transfer function is appropriately manipulated. (See PiggyBank for an example).

Volume manipulation

The volume itself is manipulated by clicking and pulling/pushing intuitively with the mouse. To zoom in/out, use the scroll wheel on the mouse. Most movement of the volume should be pretty instinctive if you know how to use a computer mouse.

'Volume Visualization' Panel Controls

There are four sliders on this panel.

- **Max Steps:** Defaults to 128. This controls the maximun number of steps that each ray will make into the volume. Once this number is reached by a ray, the ray trace terminates, even if the aggregated alpha value for the associated pixel is less than one.
- **Norm Per Ray:** Defaults to one (1). This controls the intensity scaling of the RGBa values associated with each ray. Larger values lead to more color saturation.
- **Min HZ Level:** Defaults to one (1). This controls the minimum HZ level that a voxel may be rendered at. This value will often be overridden by the LOD culling that occurs. (See VolumeVizTool.pdf for more details).
- **Lambda:** Defaults to 0.5. This controls the lambda value for the LOD culling detailed in VolumeVizTool.pdf.

'Transfer Function Menu' Controls

There are several components to this panel.

- **Color:** Defaults to black transitioning to white. This color spectrum bar is for use as a reference for the tranfer function control points. Users can add more colors to the spectrum by clicking on the color spectrum bar and using the RGB sliders on the color palette control panel to set and adjust the colors. To alter an existing color bar control point, click on the control point itself.
- **Alpha-Isovalue 2-way Control:** Defaults (under most transfer functions) to a diagonal line of slope one. Click and drag control points to adjust the emphasis of certain isovalues.
- **Transfer Function Selector:** Defaults to `Black_To_White`. Select the transfer function you wish to use and load it. Take note that the save button will overwrite the current transfer function file with the current values of the color spectrum bar and the transfer function controls. If you do not wish to overwrite the predefined transfer functions with new values, first create a separate transfer function file that can be safely written over.

Termination and Reset of SIEVAS

At times it is useful to be able to terminate and reset the connection to SIEVAS that the VVT is using. This is especially useful when doing development. On a standard Wondows machine, the backend of SIEVAS runs on port 61616. If the backend enters an error state, one must terminate the backend processes. This can be accomplished using a short shell script like the following:

```
#!/bin/sh


#echo "Which port do you want to kill? (Usually 61616)"
```

```
# Use the backtick ` to signal an external command.
# awk allows us to have multiple parsing conditions.
PID=`netstat -ano | awk '$2 ~ "127.0.0.1:61616" && $4 == "LISTENING"' | awk '{print $5}'`

#&& $4 == "LISTENING"


echo "This is PID: $PID"

if ["$PID" = ""]; then
echo "Port is not currently running."
exit 1
fi

`TASKKILL //PID $PID //F`
```

Author and Contact

- *Author:* Randall Reese
- *Email:* randall.reese@inl.gov
- *PRISM Developers:* Randall Reese, James Money, Marko Sterbentz, Nathan Morrical, Landon Woolley, Thomas Szewczyk.