VTK-Unity

Medical Viewer Asset

v1.0 - Google Docs Link

Contact: kitware@kitware.fr

Table of contents

Overview

The role of this asset is to demonstrate the Visualization Toolkit (VTK) rendering and processing capabilities inside a Unity application. It provides the minimal infrastructure required to create a medical image viewer in Unity. The VTK scene is rendered into Unity's rendering pipeline using Unity low-level native plug-in interface.

The asset is organized with the following directory structure:

- **Plugins**: Native and managed code providing VTK C++ API in Unity.
- **Scripts**: MonoBehaviour scripts used to synchronize and update VTK rendering. See section 3. for a description of each script.
- **StreamingAssets**: Data files necessary for the demo scene. You will need to copy the contents of this inner directory to the StreamingAssets directory located in the Assets folder of your project.
- **Scenes**: Example scene using files from the StreamingAssets directory to showcase VTK visualization and computation capabilities. An exhaustive description of the features available in the demo scene is given in section 4.

Limitations

- **OS support**: The native plugin has been implemented and tested for Windows only. Support for Linux and MacOS might come in future versions.
- **Unity Graphics API**: VTK relies on the OpenGL rendering backend, thus Unity graphics API must be set to OpenGLCore.
- **Missing functionalities**: This asset only provides a small subset (section 4.) of the VTK functionalities available in C++. While future versions of this asset plan to extend the list of available functionalities, it is possible to access the whole set of functionalities offered by VTK by using Activiz which provides VTK API in C#. Please contact kitware@kitware.fr for details.

Monobehaviour scripts components

The following script components are provided for users to setup a VTK scene within their Unity project.

Scripts within the Core folder are provided to synchronise the two scenes:

- *VTKCamera.cs*: This script should be attached to Unity's Camera to trigger VTK rendering. It uses Unity's command buffers to call the VTK rendering callback implemented in the native plugin.

- *VTKLight.cs*: This script should be added to a Unity light object to setup the corresponding VTK scene light and synchronize light parameters. VTK supports illumination with Directional, Point and Spot light types. Light color, Shadows, indirect lighting and cookies are unsupported for now.

- *VTKNativePlugin.cs*: This script provides a helper class for users to call VTK functionalities. It gathers every functionality offered by this asset.

Scripts within the Scene folder provides proxy classes to design and update the VTK scene:

- *VTKVolumeProxy.cs*: This script is used to modify the appearance of VTK scene objects. It allows users to show or hide VTK objects, and offers control over VTK filters parameters to alter processing of data. Best practice is to attach this script to an empty GameObject in Unity.

The full description of actions that can be applied on VTK objects is available in section 4.

To use the plugin, create a new Unity 3D project and proceed as follow:

- Change rendering backend to OpenGLCore

  - Edit->Project Settings->Player->Other Settings

  - Untick "Auto Graphics API for Windows"

  - Remove current API and add "OpenGLCore"

- Copy the Plugin folder into the Assets folder of your project

- Copy the Scripts folder into the Assets folder of your project

  - Add VTKCamera.cs to the main Camera of your Unity project

  - Add VTKLight.cs to every light component in your Unity project

  - Add VTKVolumeProxy.cs to an empty game object in the Unity scene

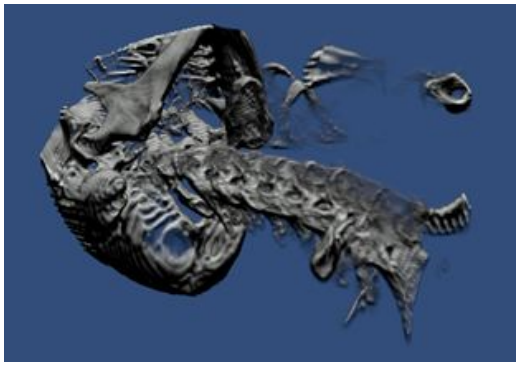
- Copy input images to be visualized in the StreamingAssets folder located in the Assets folder of the unity project.

Example Scene description

The following functionalities are presented in the example scene shipped with this asset and can be accessed through the VTKVolumeProxy.cs script.

Volume Rendering

Volume Rendering is used to display a series of 2D slices as a 3D volume.



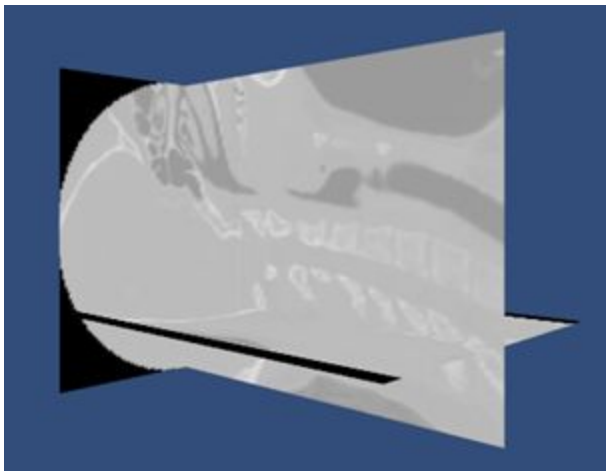
The public variables from the Volume Rendering section allow for the following modification.

- **Volume File Name**: Specify the filename of the 3D input image (volume) to visualize. The filename is relative to the StreamingAssets folder located in the Assets folder of the Unity project. The following extensions are supported as input:
    - VTK XML Image (.vti)
    - MetaImage (.mha/.mhd)
    - DICOM folder (Path to a folder containing DICOM images)
- **Volume Visibility**: Show/Hide the volume actor in the scene.
- **Auto Adjust Sample Distance**: When enabled, the sample distance used to raycast the volume is automatically adapted to keep an interactive framerate.

- **Sample Distance**: Specify the sample distance used to raycast the volume. The smaller it is, the more accurate the rendered volume is. This has no effect when Auto Adjust Sample Distance is enabled.

Volume Slices

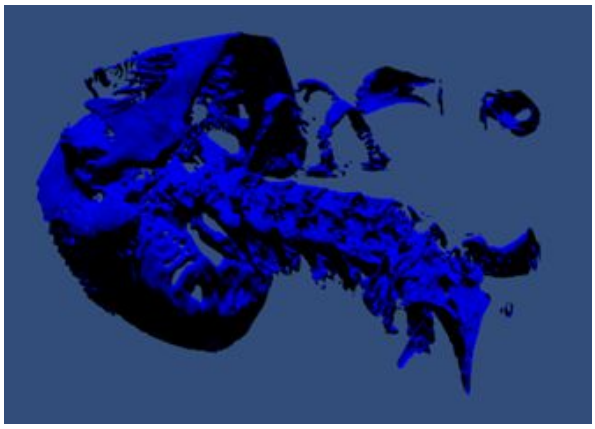
It is possible to display the anatomical planes of the input volume.



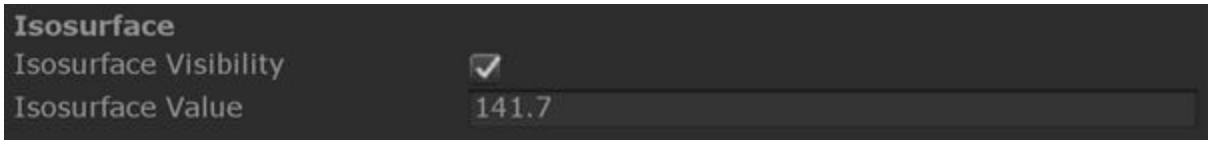
Each of the anatomical planes (Axial, Sagittal, Coronal) can be displayed or hidden using the corresponding Visibility option. In addition, the index of the slice to be displayed can be controlled in real time. If the specified slice index is out of the range defined by the image dimensions, the change will not have any effect.

Isosurfaces

Isosurface visualization allows users to extract contours of the data where all values of the data field are equal to a specified value.



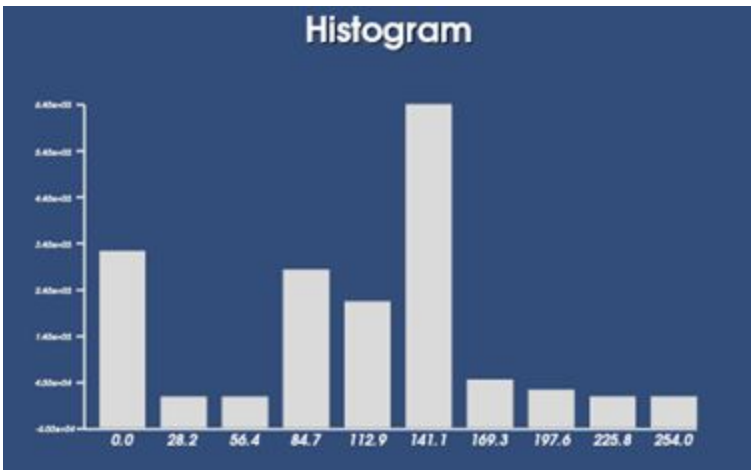
The public variables from the Isosurface section allow for the following modification.



- **Isosurface Visibility**: Show/Hide the isosurface actor.
- **Isosurface Value**: Specify the data value of the extracted contour.

Histogram

The histogram represents the distribution of data in the input image. It can be useful to identify the number of pixels assigned to a particular value for isosurface extraction.



- **Histogram Visibility**: Show/Hide the histogram.

License

VTK is an open-source toolkit licensed under the BSD license. The following license applies to all files shipped with this asset.