This document is a a very brief description of WISH. More info and documentation will come. See my PhD thesis for more background.

/ Erik Vidholm


# The WISH software

The core of the WISH software consists of a stand-alone C++ class library for image analysis, visualization, and volume haptics.

## *Image representation and file I/O*

The image representation in WISH is template based, meaning that it is theoretically possible to handle images with arbitrary data types. The *wishImage* class contains the dimensions of the image data, the raw data array itself, and functionality for accessing the data. Data access includes fast tri-linear interpolation and gradient computation. Reading and writing of image data is performed by the classes *wishImageReader* and *wishImageWriter* that can be extended to handle different file formats. The VTK file format is useful for volumetric. WISH also includes a slice sequence reader based on the FreeImage library that can read sequences of the common image file formats.

## *WISH and the H3D API*

The core functionality of WISH is integrated into the multi-sensory 3D visualization software H3D API (version 1.5) from SenseGraphics. H3D API is an open source, cross-platform API for 3D haptics and graphics based on the X3D scene-graph standard. The core of the API is written in C++, graphics rendering is performed with OpenGL, and the haptic rendering is performed with OpenHaptics from Sensable.

The interface between the WISH core and the H3D API is called WISHH3D, and consists of H3D API scene graph nodes for the image processing filters, segmentation algorithms, visualization algorithms, and volume haptics algorithms in WISH. All WISHH3D nodes are implemented in C++. The main node is the *wishH3DVolume* node. It is the parent for all other nodes and contains functionality for reading and writing images, setting up the coordinate system, and tracking the position of the haptic device in the volume. The nodes are compiled into a separate dll that is imported into the main H3D API application at startup. In this way, the WISHH3D nodes can be used and controlled through X3D or Python. Python's TkInter library is used for the GUI.

## *Visualization*

The 3D image visualization tools available in WISH are multi-planar reformatting (MPR) and hardware accelerated volume rendering. The MPR node is called *wishH3DVolumeSlicer* and consists of three orthogonal planes that are controlled with the haptic device. Adjustment of contrast and brightness is made with a standard transfer function that defines the width and center level of a gray-level window. The volume slicer also supports rendering of overlays. The volume rendering node *wishH3DVolumeRenderer* contains a hardware accelerated volume rendering engine implemented in GLSL. Colors and opacities are controlled with transfer functions, e.g., sigmoid or window/level functions. There are three compositing modes: MIP (color-correct), front-to-back alpha compositing, and iso-surface rendering with shading. For shading of isosurfaces, normals are computed with a gradient filter and stored as colors in an RGB or RGBA texture.

## *Volume haptics*

The volume haptics nodes *wishH3DVolumeHaptics* and *wishH3DHapticMode* implements proxy-based volume haptics rendering (Ikits03, LundinPalmerius07). The interface between the volume haptics engine in WISH and the one in the H3D API is realized through a customized *H3DForceEffect* structure. There are four different modes for the haptic rendering: Surface mode, viscosity mode, potential mode, and vector mode. The different modes can be controlled through haptic transfer functions.

## *Image analysis tools*

There are a number of image analysis and processing tools within the toolkit. The segmentation algorithms include fast marching (*wishH3DFastMarchingFilter*), fuzzy connectedness (*wishH3DFuzzyConnectednessFilter*), live-marching (*wishH3DLiveMarching*) and deformable simplex meshes (*wishH3DDeformableSimplexMesh*). There are also a number of image processing filters implemented, e.g., bilateral filtering (*wishH3DBilateralFilter*), Gaussian filtering (*wishH3DGaussian* Filter*), and gradient vector flow (*wishH3DGVFFilter*).