Voxie - User Manual

March 31, 2015

1 Table Of Contents

Contents

2 Introduction

This document explains the features of Voxie and how they can be used.

3 Overview

Voxie is a voxel volume viewer designed to process and visualize 3D Images created from tomography.
It's main focus is set on extracting 2 dimensional slice images from slices with arbitrary rotation and position within a 3D Image.

4 User Interface

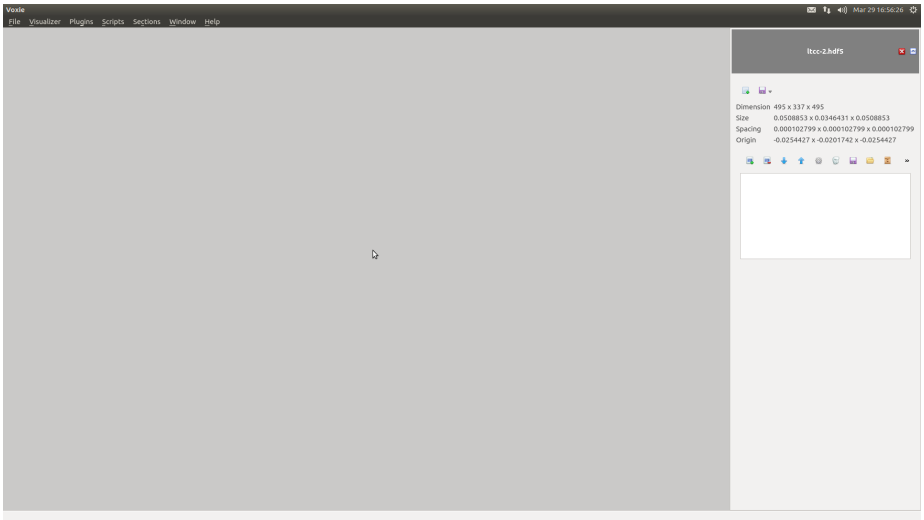
The following chapter will describe the graphical user interface of Voxie.

4.1 Windows

Voxie is an application built around one main window. It contains all main features like showing visualizers and the runtime configuration in the side panel.

4.1.1 Main Window

Figure 1: Main Window



The main window opens when Voxie gets started. It contains the main menu, the side panel and shows all open visualizers in the center.

The side panel is described further in 4.3.

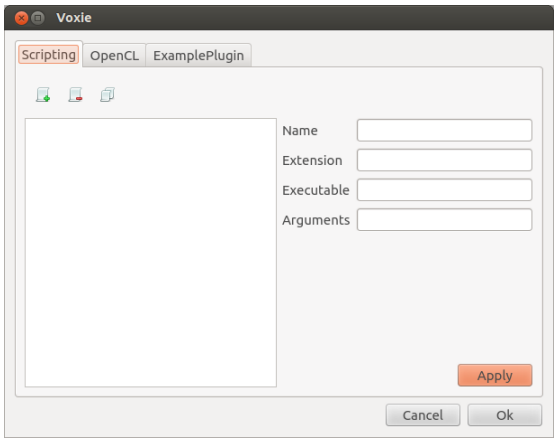
The main menu is described futher in 4.2.

4.1.2 Preferences Window

The preferences window allows the user to modify and customize Voxie for his needs.

It allows setting up custom script types for external script files, select the used OpenCL devices and options provided by plugins.

Figure 2: Preferences Window



4.1.3 Scripting Window

The scripting window provides a small JavaScript console for live scripting. A line of code can be entered in the bottom text box and with a click on *Run!* or a on the return key, the script will be executed. It is possible to navigate through previously executed commands by pressing the up or down arrow key.

It also shows a live debug log from running scripts.

Figure 3: Scripting Window

4.1.4 Visualizer Windows

Visualizer windows are a special kind of window. Each visualizer can be separated from the main window as a standalone window.

Those visualizer windows behave like the integrated windows except they can be positioned anywhere on the work space (including other monitors).

Separating a window can be achieved by clicking the "Pop out" button on top of each visualizer.

4.2 Menus

4.2.1 File

The file menu contains the main control of Voxie as well as options to load files.

To load a file, there are two options:

- Loading a file
- Using an importer

Loading a file always shows a file dialog (Fig. 4) which allows selection of all plugin supported file types. Most options show an additional dialog after confirming the dialog to make further settings on opening the file.

Example:
The HDF5 importer shows a dialog for selection of the dataset as can bee seen in Fig. 5.

Using importers allows a more flexible way of loading data sets. Importers are not bound to any predefined procedures, they can show any kind of UI and allow advanced loading of data sets, e.g. like loading a data set from multiple files, reading a data set from an external device or even just create a data set on the fly.

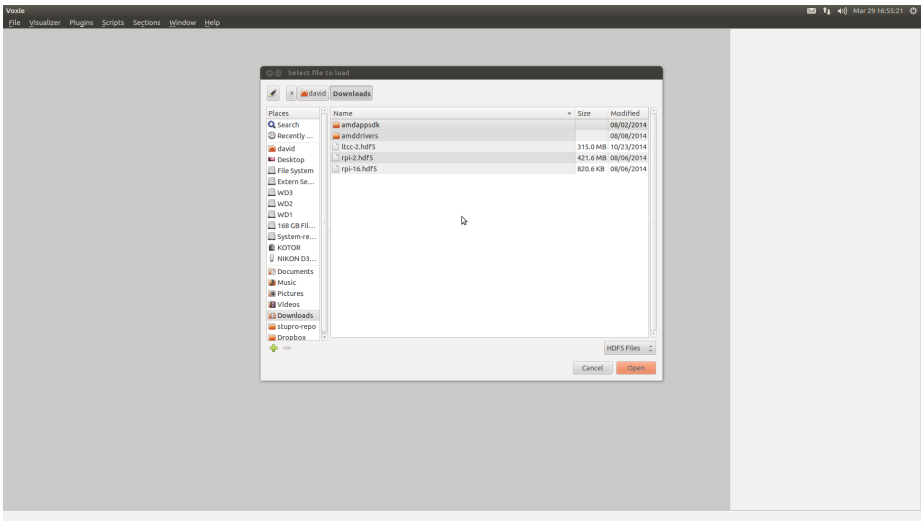
The file menu also contains an option to quit Voxie and another to set up preferences for Voxie itself as well as for plugins that support options.

4.2.2 Visualizer

The visualizer menu contains sorted entries to create new visualizers. Visualizers are ordered in 4 categories:

- *2D*
  Visualize voxel data in 2D.
- *3D*
  Visualize voxel data in 3D.
- *Analytic*
  Analyze the voxel data without a graphical visualization.
- *Miscellaneous*
  Every kind of visualizer that does not fit in the above categories

Figure 4: File Dialog



4.2.3 Plugins

This menu shows an entry for each plugin that supports UI Commands (see **??**). Each entry contains all UI commands for that plugin.

The UI command will be invoked by clicking the command entry.

4.2.4 Scripts

This menu allows to show the scripting console which is explained in 4.1.3.

It also shows all files that reside the folder *scripts* next to the voxie executable. A click on a script will execute it.

JavaScript (*.js) files are shown by default and will be executed by the internal JavaScript scripting engine.

Other scripts can be configured to be shown in the menu by setting up a new script type in the *Scripting*-Section of the preferences window (See 4.1.2). The files will be executed by starting the external application defined there.

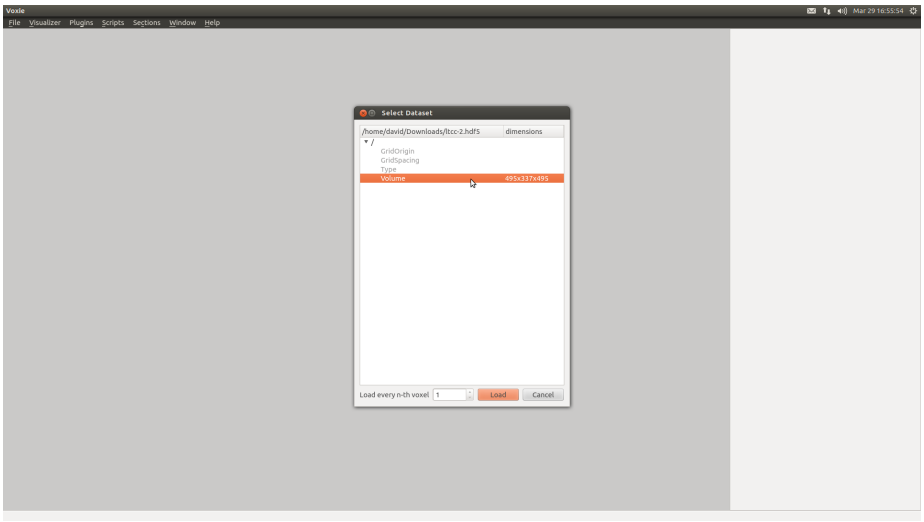
4.2.5 Sections

The *Sections* menu will show all currently active sections. A click on the menu item will show or hide the section in the side panel.

4.2.6 Windows

This menu allows to reorder all currently opened visualizer windows in 4 styles:

Figure 5: HDF5 Dialog

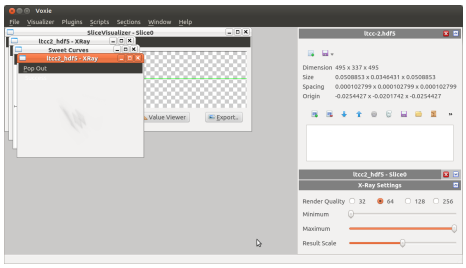


- *Cascade*
  Will cascade the windows from the top left corder to the bottom right corner.
- *Tile*
  Will till the windows in a regular grid.
- *Fill*
  Will maximize the current window.
- *Tabbed*
  Will enable or disable tabbed mode. In tabbed mode, all windows are maximized and can be switched with a tab control.

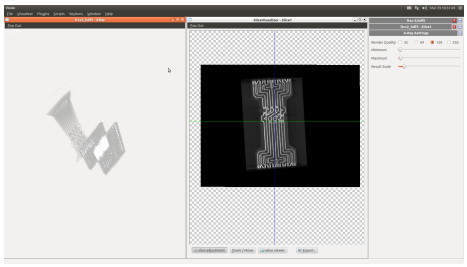
4.2.7 Help

The help menu offers several possibilites to get help with Voxie:

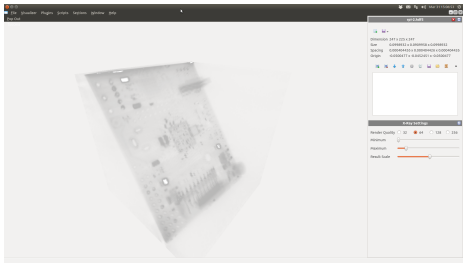
- *Manuel*
  Opens this manual.
- *Wiki*
  Opens the wiki in the default browser.
- *About*
  Shows an about dialog with some quick info about Voxie.



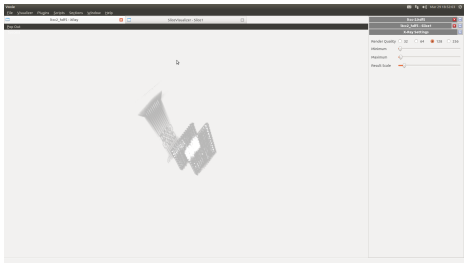
(a) Cascade



(b) Tile



(c) Fill



(d) Tabbed

Figure 6: Window Modes

4.3 Sections

Voxie features the side panel, a multifunctional panel showing context dependent information and settings to the currently active data sets and visualizers.

Sections can be collapsed so they only show their header and may be closable. If a section gets closed, all data connected will be released (e.g. closing a data set will close all attached visualizers and slices as well).

4.3.1 Data Set Section

5 Visualizer

Voxie comes with 2 standard visualizer plugins included. These plugins feature a 2D Slice visualizer for visualizing single slices, and a two 3D visualizers for for visualizing 3D datasets.

5.1 2D

Voxie supports the creation of slices. A slice is a plain intersecting the dataset. Its visual representation is defined by all voxel values it intersects.

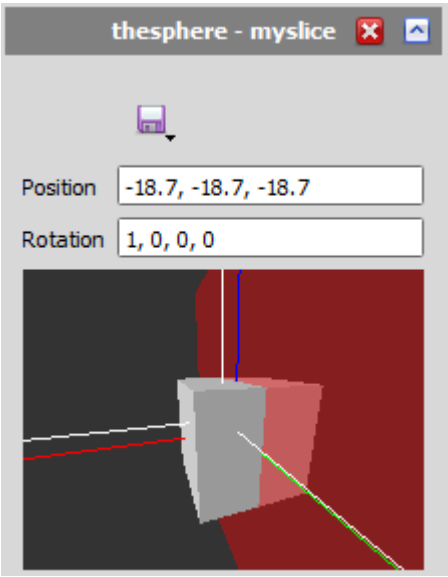
Voxie supports multiple operations on slices. This includes adjusting, filtering, colorizing and analyzing.

5.1.1 Creating a slice

A slice is created by telling the data set (VoxelData) to create a new slice. In the UI this is done by clicking on the *Create Slice* button in the section representing the data set.

After adding a slice a new section will appear displaying the position and rotation of the slice. The 3D image describes the position of the slice within the dataset. Values are coded as x (green), y (blue), z (red). The rotation is displayed as normalized quaternion.

Figure 7: Slice Section



5.1.2 Exporting a slice

The raw values of the slice can be exported in HDF5 format by clicking on the *Save* button in the slice's section.

5.1.3 Displaying a slice

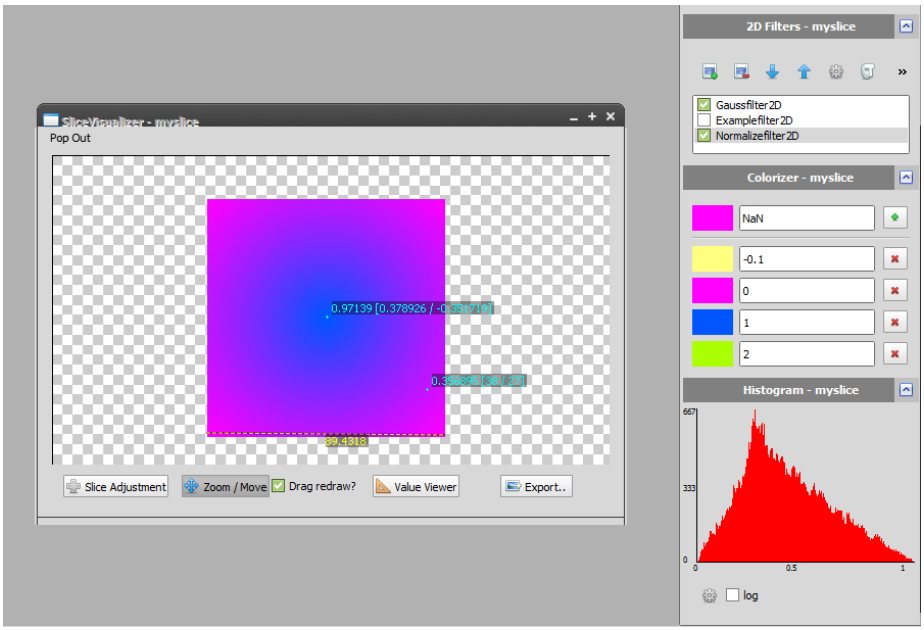
To display a slice Voxie's SliceView plugin has to be used. To add a new Slice View click on Visualizer, then 2D, then Slice.

A dialog will appear. Choosing a slice will create a new window and several new sections.

5.1.4 The slice view

By default the slice view window offers a canvas that displays the slice associated with the slice view. It also allows for tools to be used which can modify the

Figure 8: The Slice View and its sections and tools



slice or display information. Tools can be switched by using the number keys.

5.1.5 Slice Adjustment Tool

The slice adjustment tool allows for interactive adjustments to be made to the slices configuration. When this tool is active the x and y axis of the slice are shown (x = green , y = blue). Notice that x points right and y points down(!).

The slice's origin can be moved to another point on the plane by [right-clicking]. This is useful because the slice's origin is the pivot point for its rotations.
The slice's origin can also be moved along its normal vector with the mouse-wheel or [page-up/down] keys. This Movement can be fine adjusted by holding down [shift] simultaneously.

The slice's rotation can be adjusted an variuos ways. To rotate the x-y-plane (rotate arround normal vector) hold down [ctrl] and [drag] the mouse with [left-click] arround the origin.
This way you can also rotate arround the other axes. To rotate the slice arround the x axis hold down [ctrl]+[x] and drag with [left-click], with [ctrl]+[y]+[left-click][drag] you can rotate arround the y axis.
Alternatively you can use the arrow keys [up]/[down]/[left]/[right] to rotate arround the x or y axis. [shift] can be used to slow down the movement with the arrow keys.

For more arbitrary rotation you can "push down" the slice on a specific point with a simple [left-click]. This causes the slice to tilt towards the direction of clicked point (origin is pivot point). To fully understand this imagine the vector that points from the origin to the click point, the vector that is perpendicular to this vector and the normal vector of the plane is the axis of rotation.
The step-size can be reduced by holding down [shift].

5.1.6 Zoom / Move Tool

The zoom and move tool allows one to zoom and move around the displayed image. Holding the left mouse button and dragging will move the image around. The mouse wheel is used to zoom the image. Toggling the checkbox labeled *Drag redraw?* will allow for less processing power to be required as the image will only be altered when the mouse button is released.

5.1.7 Value Viewer Tool

The value viewer tool displays values within the image. The value shown below the curser represents the position and the float value of the slice at the given position. Left-clicking will make the value continue to be displayed. Right-clicking will clear all displayed information. Pressing shift shows the unfiltered value. Pressing control will round the position values. Dragging the mouse will calculate the distance between two points in metric distances (see HDF5 specification). Modifying the slice or the generated image will clear all values.

5.1.8 Mask Selection Tool

The Selection Tool allows the User to select an area on the image and on this selected area the filter is applied to. To use this function a filter in the Filter Chain must be selected and only on this filter the mask is applied on. To create a mask you have to click on the mask symbol, then four new buttons appears on the Slice View, “Rectangle“, “Ellipse“, “Polygon“and “Clear“.

By clicking on the **Rectangle Button**, it allows to create a rectangle shape. To do that click and hold the left mouse button and move the mouse. A yellow rectangle appears, this shows the preview of the selection. By realising the left mouse button it finishes the selection and the yellow rectangle turns into red.

By clicking on the **Ellipse Button**, it allows to create a ellipse shape. To do that click and hold the left mouse Button and move the mouse. A yellow ellipse appears, like before it shows the preview of the selection. The first click defines the center of the ellipse. By realising the left mouse button it finishes the selection and the yellow rectangle turns into red.

By clicking on the **Polygon Button**, it allows to create a polygon shape. To

do that click on the image, every click is one vertex of the polygon. Every click connects the vertices in yellow, this is just the preview. By pressing space, the polygon shape closes automatically. Notice that at least three vertices are needed to close a polygon shape.

By clicking on the **Clear Button**, it deletes every created shape on the mask only for the selected filter.

5.1.9 Export Tool

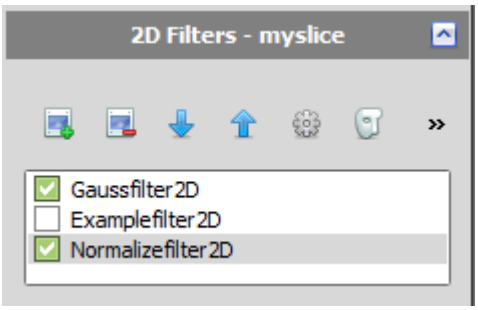
The export tool will allow for the currently displayed image to be saved. There are three modes available; filtered image, filtered and colorized image or the whole canvas (including borders outside the image).

5.1.10 Slice View Sections

A slice view creates three new sections on initialization.

- *2D Filter*
  This section allows for definition of filters, their order and masks. An ordered set of filters is called a filterchain.
- *Colorizer*
  The colorizer section maps values to colors. In between values will be interpolated.
- *Histogram*
  The value distribution is visualized here. It also offers advanced options like logarithmic scaling.

Figure 9: 2D Filter Section



5.1.11 Adding 2D Filters

A 2D filter can be added by clicking on the *Add Filter* button will open a dialogue allowing one to choose what kind of filter is to be added. Filters are automatically active after creation.

5.1.12 Removing 2D Filters

Filters can be removed by highlighting a filter in the section then clicking the *Remove Filter* button.

5.1.13 Choosing which Filters to apply

Filters can be temporarily deactivated by toggling the checkbox next to their name.

5.1.14 Ordering 2D Filters

Filters are applied in order from top to bottom. Their ordering can be altered by highlighting a filter then pressing the array buttons to move them up or down.

5.1.15 Configuration of 2D Filters

Some filters posses a configuration dialogue. The dialogue can be opened by highlighting the desired filter then pressing the *Options* button (*Wheel*).

5.1.16 Loading and saving Filterchains

Filterchains including the individual filter settings can be exported and imported using the *Export Filterchain* and *Import Filterchain* buttons. The filterchain will be represented by a human readable xml file.

5.1.17 Filter Masks

After activating the filter feature by adding filters, the *Filter Mask* button will activate the filter mask feature in the slice view. This allows for the *Mask Selection Tool* to be used. See 5.1.8 for more information.

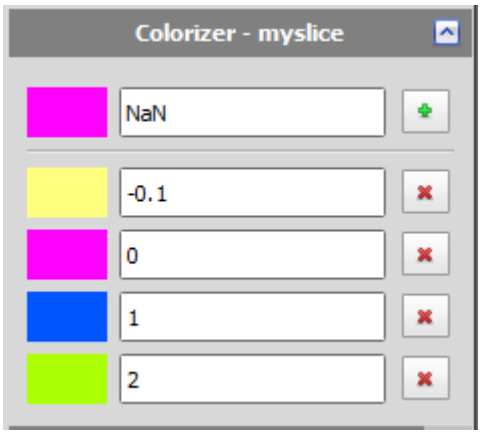
5.1.18 Colorizer

The colorizer maps values to colors. Colors in between the given mappings will be interpolated. Mapping can be added by pressing the *Add Mapping* button in the first column. This will add a new mapping to the list of mappings below. The colors can be changed by clicking on the displayed color. This will invoke the operating system specific color picker.
To remove a mapping simply click on the *Remove Mapping* button.
Please note that mappings with the same value will be automatically merged, the new mapping having priority.

5.1.19 Histogram

The histogram section shows a visualization of the value distribution of the slice. The scale can be set to be linear or logarithmic. Upper and lower bound as well as the maximum count value can be set via dialogue.

Figure 10: Colorizer Section



5.1.20 Displaying the difference between two slices

If two Slices should be compared, the DiffView plugin has to be used. To add a new DIff View click on Visualizer, then 2D, then DiffView.

A dialog will appear. Choosing two slices will create a new window and several new sections.

The DiffView is build similar to the SliceView, thereforce the following subsctions only describe the changes between the DiffView and the SliceView.

5.1.21 Slice Adjustment Tool

The Slice Adjustment Tool within the DiffView brings two new buttons: "Switch Slice" to switch the slice that is currently adjusted and "Select Both" to adjust both slices at the same time.

The Origin of the currently not altered Slice will be shown as DashLine.

5.1.22 Value Viewer Tool

The value viewer for the DiffView will always display two values, as there are two Slices, except if the Shift-Key is pressed, then the value for the filtered Image (which is not connected to a slice anymore) will be shown.

5.2 3D

3D visualizers allow the user to get a an idea of the data set they are working with. The visualizers don't allow exact measurement or inspection but they provide enough information to get how the data set look.

5.3 Isosurface

The isosurface visualizer allows display of an isosurface. Isosurfaces are surfaces in data sets with equal density. This allows displaying structures of similar or equal density as a solid object.

5.3.1 Settings

The Isosurface visualizer can be set up via its side panel section. The section has multiple options:

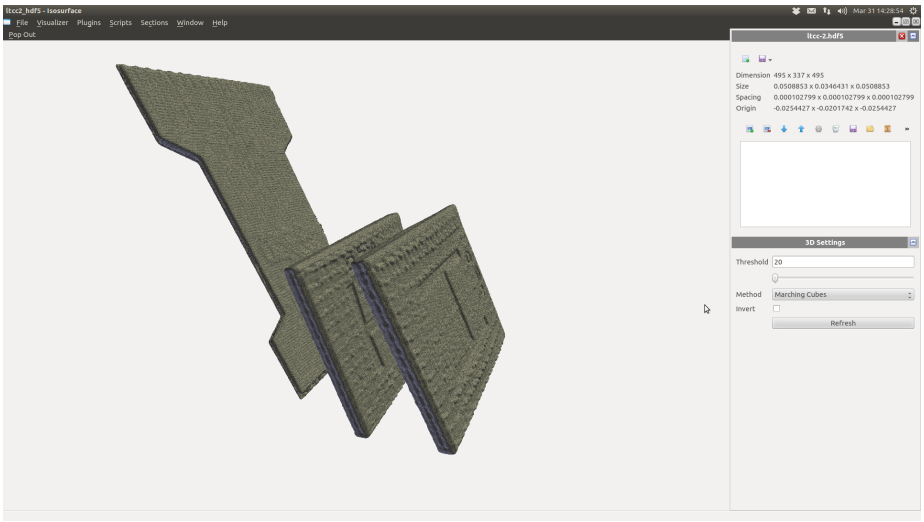
- *Threshold*
  The threshold value for the isosurface.
- *Method*
  The method used for generating the isosurface.
- *Invert*
  A flag that determines if the surface is built against values below the threshold or above the threshold.
- *Refresh*
  Refreshes the isosurface model.

The Isosurface visualizer needs some time to generate an isosurface so it does not view anything by default. The visualizer needs to be set up and then the *Refresh*-Button must be pressed to regenerate the isosurface model.

5.3.2 Controls

The data set can be rotated by dragging the mouse over the visualizer window while holding down the left mouse button. Scrolling the mouse wheel will zoom in or out the view.

Figure 11: Isosurface



5.4 X-Ray

The X-Ray visualizer allows seeing the data set as a whole piece. It accumulates the voxels it finds while raycasting into the scene and thus shows the interior of the data set.

It is useful to gather information about where specific structures are located in the data set, if there are cavities in the data set or points with a high density.

5.4.1 Settings

The X-Ray visualizer can be set up via its side panel section. The section has multiple options:

- *Render Quality*
  The number of samples used by the raycaster. Lower values are faster, higher values give more details.

- *Minimum*
  The minimum value considered as black/transparent. Can be used as an additional high-pass filter.

- *Maximum*
  The maximum value considered as white/opaque. Can be used as a range adjustment.

- *Result Scale*
  Scales the result from the range options by a given amount. Can be used to tweak the image brightness.

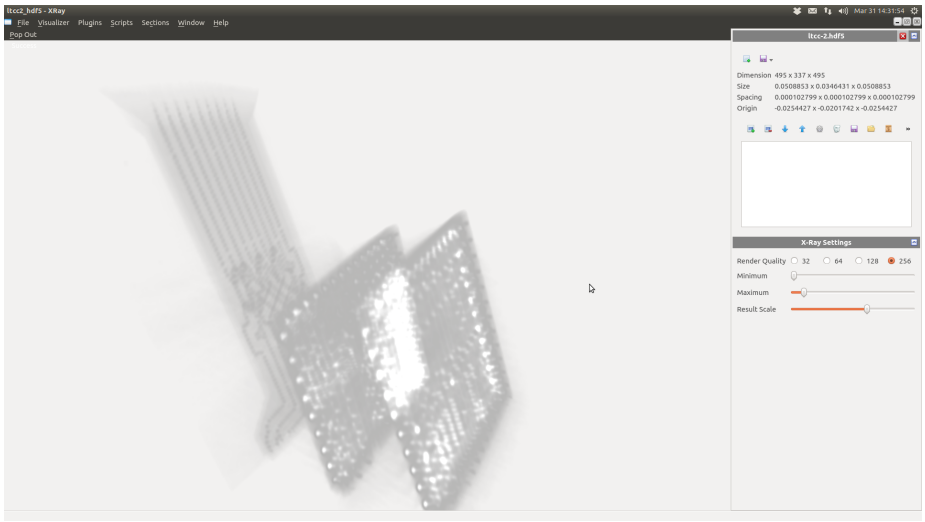
The settings the of visualizer are unit-less as the visualizer only provides a visual representation that must be tweaked by eye.

For a computationally under-performing computer it is recommended to not use the higher render quality settings as they draw a lot of computational power.

5.4.2 Controls

The data set can be rotated by dragging the mouse over the visualizer window while holding down the left mouse button. Scrolling the mouse wheel will zoom in or out the view.

Figure 12: X-Ray



6 Scripting

Voxie provides a scripting interface that allows the user to extend the features of the Voxie by custom sequences.

Scripting comes in two flavours: Internal scripting with access to all objects inside of Voxie or external scripting via D-Bus.

6.1 Objects

Voxie provides several objects for scripting. Each object has its own responsibility and represents a part from the Voxie ecosystem.

6.1.1 Common API

- *objectName*
  The name of the object. Can be changed if wanted.
- *list()*
  Returns a QStringList with the names of all children
- *explore()*
  Returns a string containing a description of the API.

6.1.2 Root

The root object is the central piece of Voxie. It allows access to all other objects as well as some basic functionality to control Voxie itself.

- *plugins*
  Object that contains all loaded plugins.
- *dataSets*
  Object that contains all current data sets.
- *log(value)*
  Writes a value to the script console
- *quit()*
  Shuts down voxie.
- *exec(script)*
  Runs the script parameter in the internal script engine.
- *execFile(fileName)*
  Runs the script file in the internal script engine.
- *resolve(name)*
  Resolves an object by the script path.

6.1.3 Plugin

The plugin representation inside voxie. Contains container properties that allow access to all functionality of a plugin.

- *visualizer*
  Object that contains visualizers.
- *extension*
  Object that contains plugin extensions.
- *filters2d*
  Object that contains 2d filters.

- *filters3d*
  Object that contains 3d filters.
- *loaders*
  Object that contains loaders.
- *importers*
  Object that contains importers.
- *voxelExporters*
  Object that contains voxel exporters.
- *sliceExporters*
  Object that contains slice exporters..

6.1.4 Data Set

TBD

6.1.5 Slice

TBD

6.2 Internal Scripting

Voxie supports scripting with an integrated JavaScript engine.

The engine allows access to the most important objects of voxie including all plugins, data sets and slices.

6.2.1 Example

The following example creates a new function called *listPlugins()* that will list all plugins together with their contents.

```
function inspect(obj)
{
        log("Inspecting " + obj)
        for(var prop in obj)
        {
                log(prop + " = " + obj[prop])
        }
}

function listPlugins()
{
  var pluginNames = plugins.list();
  log("Plugins:")
  for(var key1 in pluginNames)
  {
```

```
    var pluginName = pluginNames[key1]
    var plugin = plugins[pluginName]
    log("  Plugin: " + pluginName)
    var items = function(key)
    {
      for(var item in plugin[key].list())
      {
        log("      " + plugin[key].list()[item])
      }
    }

    log("    Visulizers:"); items("visualizer")
    log("    Extensions:"); items("extensions")
    log("    Filters 2D:"); items("filters2d")
    log("    Filters 3D:"); items("filters3d")
    log("    Loaders:"); items("loaders")
    log("    Importers:"); items("importers")
  }
}
```

6.3 External Scripting

Voxie also allows to be scripted via D-Bus. The first instance of Voxie will be known under the service name *org.voxie.app*, all other instances will only be accessible by the bus object name.

It exposes two main objects with their children to D-Bus: */org/voxie/window* and */org/voxie/root*.

/org/voxie/window is the main window of the application.

/org/voxie/root is the object also accessible via the internal JavaScript engine.

```
import sys
import dbus

bus = dbus.SessionBus()

proxy = bus.get_object(
    sys.argv[1],
    '/org/voxie/root')
iface = dbus.Interface(
    proxy,
    dbus_interface='org.voxie.core.Root')
iface.quit()
```