FFI RAPPORT

VOLUVIZ 1.0 REPORT

GAARDER Trond, HELGELAND Anders

FFI/RAPPORT-2002/03449

FFIBM/820/170

Approved
Kjeller 2. September 2002

Bjarne Haugstad
Director of Research

VOLUVIZ 1.0 REPORT

GAARDER Trond, HELGELAND Anders

FFI/RAPPORT-2002/03449

FORSVARETS FORSKNINGSINSTITUTT
Norwegian Defence Research Establishment
P O Box 25, NO-2027 Kjeller, Norway

FORSVARETS FORSKNINGSINSTITUTT (FFI)
Norwegian Defence Research Establishment

UNCLASSIFIED

P O BOX 25
N0-2027 KJELLER, NORWAY

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

REPORT DOCUMENTATION PAGE

1) PUBL/REPORT NUMBER FFI/RAPPORT-2002/03449	2) SECURITY CLASSIFICATION UNCLASSIFIED	3) NUMBER OF PAGES 25
1a) PROJECT REFERENCE FFIBM/820/170	2a) DECLASSIFICATION/DOWNGRADING SCHEDULE -	

4) TITLE
VOLUVIZ 1.0 REPORT

5) NAMES OF AUTHOR(S) IN FULL (surname first)
GAARDER Trond, HELGELAND Anders

6) DISTRIBUTION STATEMENT
Approved for public release. Distribution unlimited. (Offentlig tilgjengelig)

7) INDEXING TERMS

	IN ENGLISH:		IN NORWEGIAN:
a)	VoluViz	a)	VoluViz
b)	Visualization	b)	Visualisering
c)	Direct volume rendering	c)	Direkte volumrendering
d)	Volumizer 2	d)	Volumizer 2
e)		e)	

THESAURUS REFERENCE:

8) ABSTRACT

VoluViz is a direct volume rendering application based on SGI's OpenGL Volumizer 2 and OpenGL, and is designed for interactive viewing of three-dimensional volume data. Its features include trilinear interpolation, interactive color table (both RGBA and HSVA), picking of subsets, clip planes, and visualization of two fields.

The report is in two parts. The first part gives a description of the functionality of VoluViz and treats common usage patterns. The second part is a technical manual discussing implementation issues and program structure. Possible future extensions are also discussed.

9) DATE 2. September 2002	AUTHORIZED BY This page only Bjarne Haugstad	POSITION Director of Research

ISBN-82-464-0669-8

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE
(when data entered)

PREFACE

About this report

This report is divided into two parts. The first part gives a description of the functionality of VoluViz and treats common usage patterns.

The second part is a technical manual discussing implementation issues and program structure. Possible future extensions are also discussed.

Audience of this report

The first part is written for users of the VoluViz application. The focus is on learning to use the application, not on technical details. No previous experience is needed.

The second part is intended for advanced users and developers interested in the internals of the VoluViz application and class library. Experience with C++, Qt, OpenGL, and Volumizer 2 is recommended. Some familiarity with volume visualization principles is also useful.

Note on datasets

The VoluViz application reads volume datasets stored in HDF4 [2] files. There are some restrictions on the format of these files. See Appendix A.1 on page 20 for further details.

CONTENTS

VOLUVIZ 1.0 REPORT

1 USER MANUAL

1.1 Introduction

VoluViz is a volume rendering application designed for viewing three-dimensional volume data. It supports interactive viewing of large (e.g. 512^3 or greater) datasets. VoluViz has many features including trilinear interpolation, interactive color table (both RGBA and HSVA), picking of subsets, clip planes and visualization of two fields. Currently, VoluViz is restricted to handle data defined on uniform grids.

Due to the use of SGI's OpenGL Volumizer 2 [4], Voluviz only runs on SGI hardware with support for 3D texture mapping and color tables (for example the Onyx 3200).

To run the application, enter the following command:

```
% voluviz [filename.hdf]
```

If it is necessary to compile VoluViz, instructions can be found in section 2.3 on page 14.

1.2 Functionality

VoluViz is a GUI (Graphical User Interface) application, and commands are accessed via the menu system (see figure 1.1).

1.2.1 Using the mouse

- To *rotate* the volume, press and hold the left mouse button while moving the mouse.
- To *move* the volume, press and hold the middle mouse button while moving the mouse.
- To *zoom* in and out, press and hold the middle and left mouse buttons while moving the mouse backwards and forwards.

1.2.2 File menu

Open loads a HDF volume file. The user must then use the subset dialog to set the desired view volume and press apply to render the volume. (Hotkey: *Ctrl+O*)

Quit quits the VoluViz application. (Hotkey: *Ctrl+Q*)

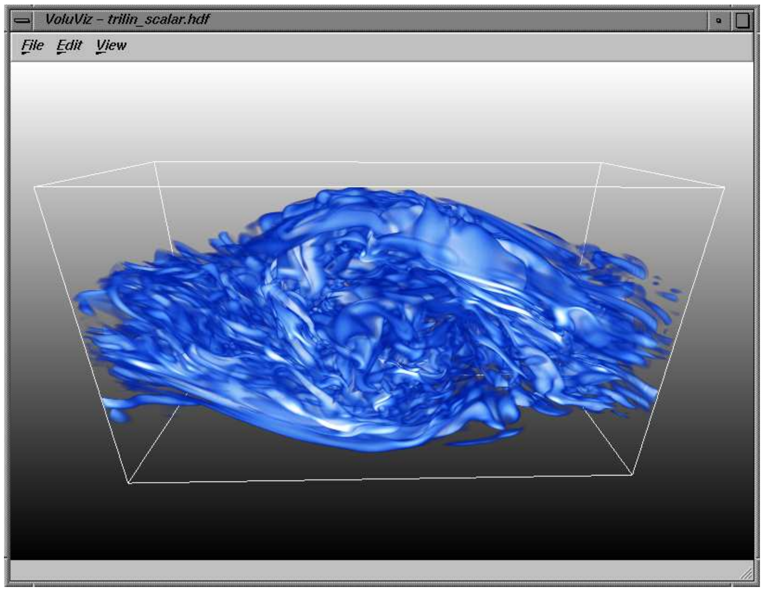


Figure 1.1: The VoluViz GUI with a sample dataset loaded.

1.2.3 Edit menu

Background sets the background color. This can be:

- a single color.
- a vertical gradient between two colors.
- a gradient between four colors defined at the corners of the screen.

The colors are chosen by entering a string of color names (e.g. "black white"). See the Qt documentation [3] for the **QColor** class under `setNamedColor()` for a list of valid colors.

Color editor opens the Color editor. See section 1.3 for information on using the Color editor. (Hotkey: *Ctrl+C*)

Subset (re)displays the subset dialog box. Use the subset dialog box to specify the volume to be rendered. (Hotkey: *Ctrl+S*)

Sampling rate sets the number of slices that are used to render the volume. If the **reduced** checkbox is checked, the number of slices are reduced during user manipulation of the volume (moving, rotating, scaling etc.). This decreases the rendering workload and increases the interactivity of the application. (Hotkey: *Ctrl+R*)

1.2.4 View menu

Axis displays the coordinates of the volume along the x, y and z axis. (Hotkey: *Ctrl+A*)

Bounding box displays an axis aligned bounding box surrounding the volume. (Hotkey: *Ctrl+B*)

Clip plane enables the clip plane. The volume on one side of the plane is displayed while the other side is "clipped". The clip plane can be manipulated using the mouse in conjunction with the keyboard:

- To *rotate* the clip plane press *Shift* and hold the left mouse button while moving the mouse.
- To *slide* the clip plane along the plane normal, press *Shift* and hold the middle mouse button while moving the mouse.
- The clip plane can be aligned with the x, y and z axis by pressing *Shift+X*, *Shift+Y* and *Shift+Z* respectively. Pressing one of these combinations twice in a row flips the clip plane 180 degrees, so that the visible portion of the volume becomes invisble (clipped) and vice versa.
- To *reset* the clip plane to the initial configuration, press *Shift+R*.

(Hotkey: *Ctrl+P*)

1.3 Color editor

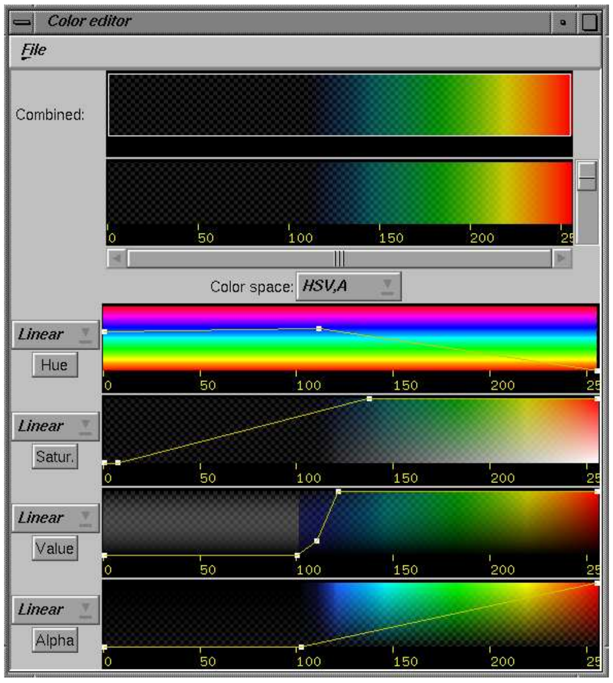


Figure 1.2: The VoluViz color table editor.

The VoluViz Color editor is divided into three main parts (see figure 1.2). The *combined section* include a global view of the color table and a zoomable detailed view.

The *color space* combo box allows you to choose between HSVA and RGBA color spaces.

The *channel views* shows the color table for each color component. The channels will be *hue*, *saturation*, *value* and *alpha*, or *red*, *green*, *blue* and *alpha* depending on the chosen color space.

To modify the color table, the user inserts and manipulates a number of *knots* in the desired channel view.

- To *insert* a knot, press the middle mouse button.
- To *move* an existing knot, move the mouse over the desired knot, and while pressing the left mouse button, drag the knot to its new position.
- To *delete* a knot, move the mouse over the desired knot and press the the right mouse button.

Each channel has an interpolation mode (linear, spline or gauss) that determines the interpolation funtion between knots.

The exact operation of the color table depends on the data set being viewed. Voluviz supports the following texture types:

- *One-component texture*–contains only one component per voxel. In this case all four channels use those values as indices.
- *Two-component texture*–contains two components per voxel (e.g. two field vizualisation), the first component maps to the HSV/RGB channels, while the second component maps to the alpha channel.
- *Four-component texture*–contains four components per voxel and maps these to the RGBA channels respectively. The HSVA color space does not work very well in this scenario. See the section on **vzParameterLookupTable** and **vzParameterVolumeTexture** in [4].

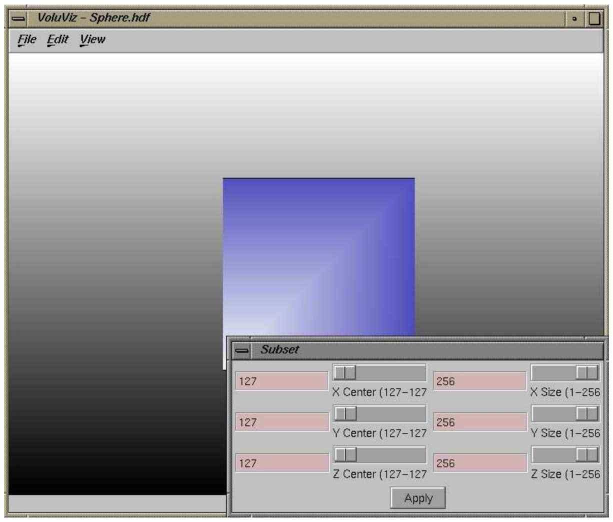
Note that all VoluViz color tables have 256 entries. This is due to the fact that VoluViz only supports unsigned byte textures.

The Color editor also supports saving and loading of color tables. Load and save functionality can be accessed from the Color editor file menu.

1.4 Example

This is a step-by-step example of using VoluViz to view a volumetric data set, using most of the features present in VoluViz.

1. Open a data set by choosing *File* → *Open* (see figure 1.3).



127 X Center (127–127 256 X Size (1–256
127 Y Center (127–127 256 Y Size (1–256
127 Z Center (127–127 256 Z Size (1–256

Figure 1.3: Loading data set.

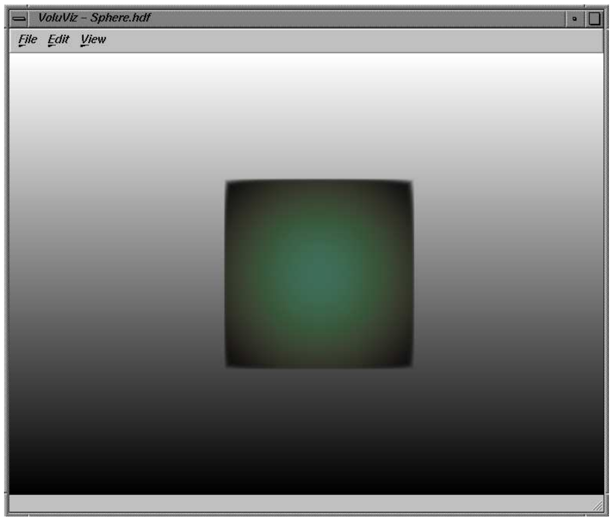


Figure 1.4: Volume loaded and ready.

2. Press the apply button in the subset dialog box to load and render the volume (see figure 1.4).

3. Open the Color editor by choosing *Edit* → *Color editor*.

4. Create a suitable color table using the Color editor (see figure 1.5).

5. Enable the clip plane by choosing *View* → *Clip plane* (see figure 1.6).

6. Choose *View* → *Bounding box* to display an axis-aligned boundig box around the volume.

7. Position the volume using the mouse (see figure 1.7).

8. Change the background using *Edit* → *Background*.

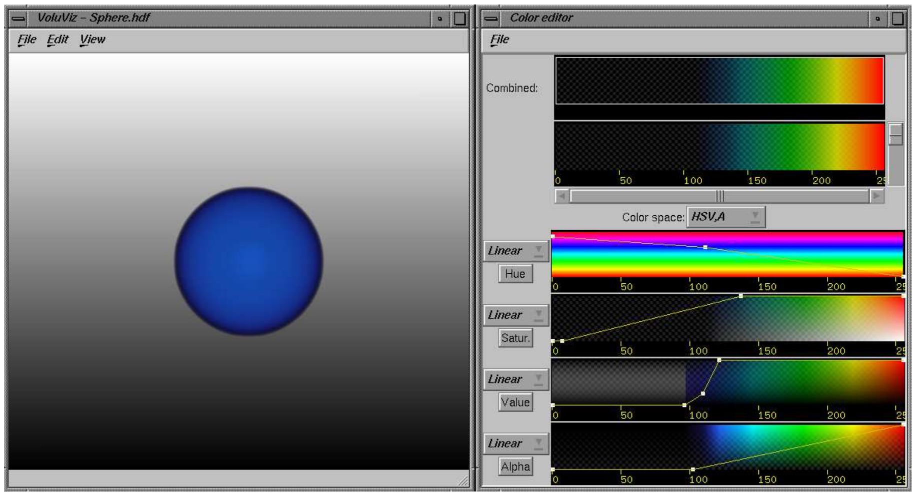


Figure 1.5: Manipulating the color table.

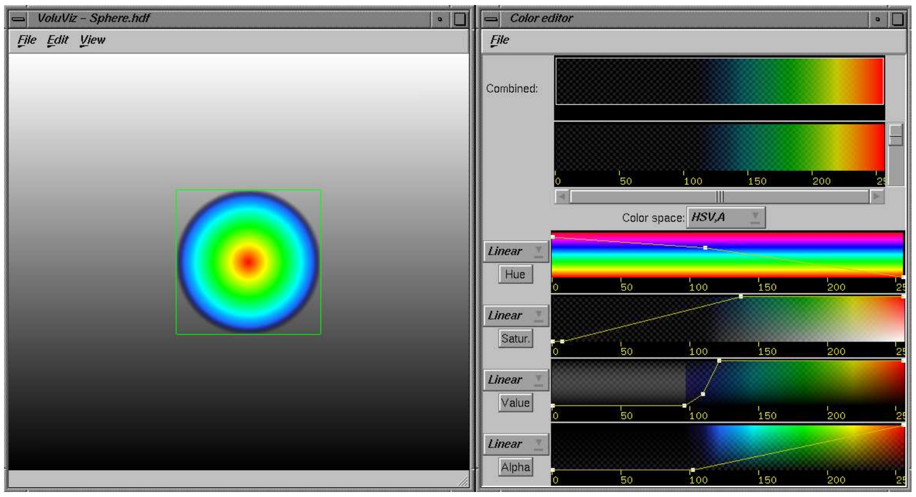


Figure 1.6: Setting a clip plane.

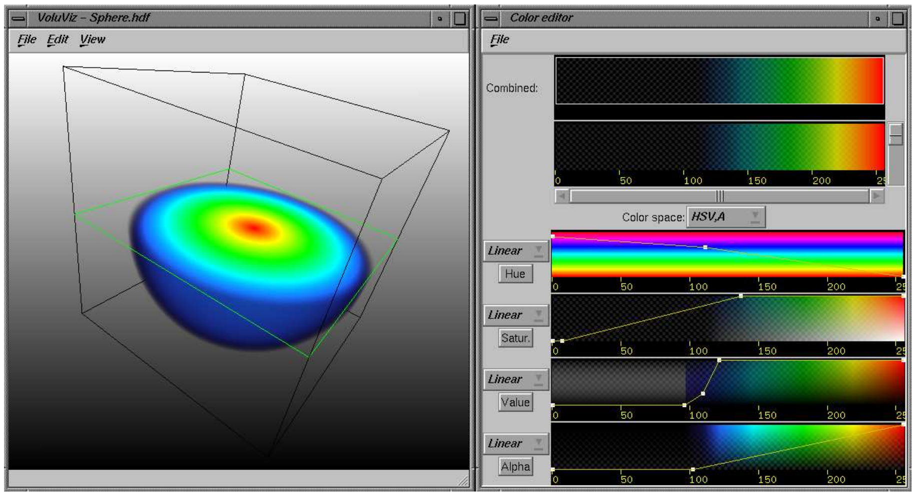
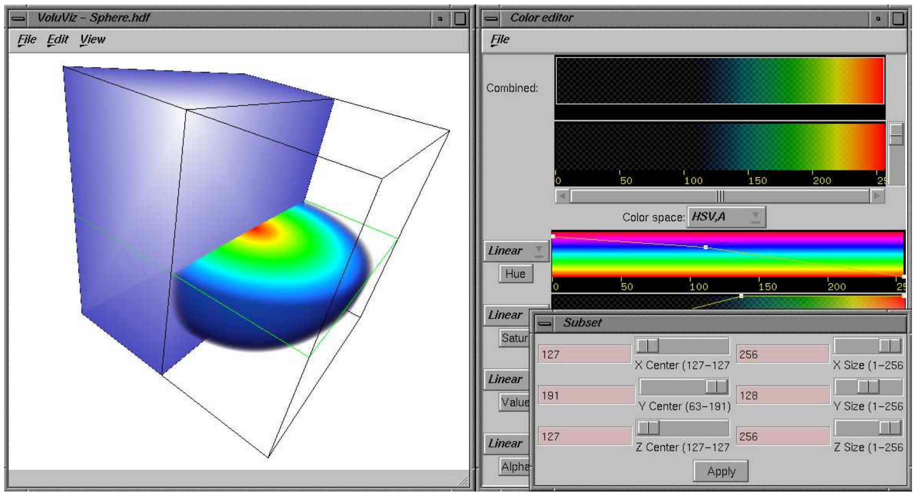


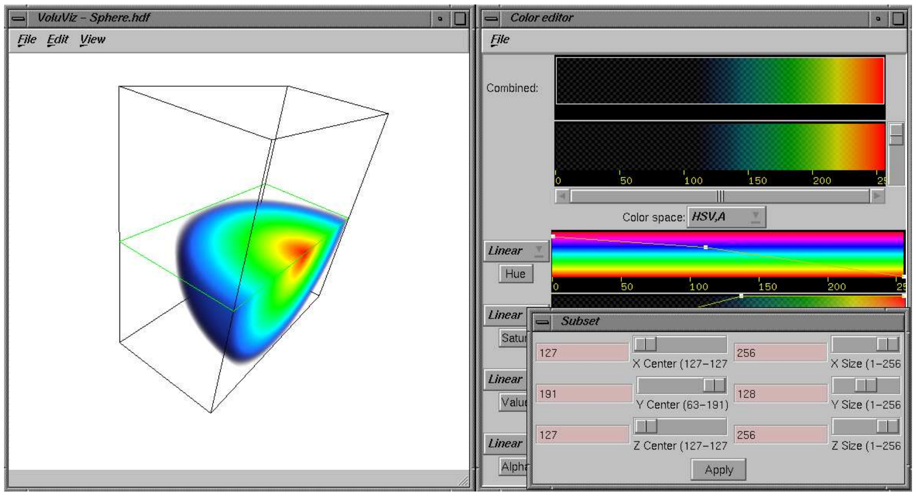
Figure 1.7: Positioning the volume.



Color space: HSV,A

127 X Center (127–127
256 X Size (1–256
191 Y Center (63–191)
128 Y Size (1–256
127 Z Center (127–127
256 Z Size (1–256

Figure 1.8: Picking a subset.



Color space: HSV,A

127 X Center (127–127
256 X Size (1–256
191 Y Center (63–191)
128 Y Size (1–256
127 Z Center (127–127
256 Z Size (1–256

Figure 1.9: Loading the selected subset.

9. Select a subset of the original volume by using the subset dialog box (see figure 1.8).
10. Press the apply button (see figure 1.9).
11. Repeat until satisfied.

2 TECHNICAL MANUAL

2.1 Introduction

VoluViz is a direct volume rendering application based on SGI Volumizer 2 and OpenGL. Trolltech's Qt GUI library is used to create the user interface. Note that Qt only exists in a 32-bit configuration, therefore VoluViz must be compiled as a 32-bit application. Due to limitations in the HDF4 file format, the size of the volume texture can not exceed 2GB.

The src/docs directory contains documentation generated by Doxygen [1] in several formats (html, TeX, etc.).

2.2 Prerequisites

The following software must be installed on your system:

- OpenGL
- SGI OpenGL Volumizer 2 [4] (tested with version 2.2)
- Qt [3] (tested with version 2.3.1)
- HDF4 [2]
- C++ compiler and gmake

2.3 Compiling and Running the Application

To compile the application, go to the VoluViz root directory and type

```
% gmake
```

Modification of the Makefile might be necessary depending on where the prerequisite packages are located.

To run the application, enter the following command:

```
% voluviz [filename.hdf]
```

2.4 Program Components

VoluViz consists of the following main components:

- The main Qt GUI application
- The renderer

- File and dataset handling
- The color table editor
- Misc. dialogs and support code

2.5 The main Qt GUI application

The *main Qt GUI application* handles the user interaction and controls the other VoluViz components.

The following files are included in the *main Qt GUI application* component:

- main.cpp
- VoluViz.h
- VoluViz.cpp

2.6 The renderer

The *renderer* draws the volume data and other graphical elements. It also handles mouse manipulation of the scene and clip plane (translating, rotating, scaling).

The following files are included in the *renderer* component:

- Viewer.h
- Viewer.cpp

The **Viewer** class is the heart of VoluViz. It renders the volume and geometry as well as supporting geometry like bounding boxes and axes. SGI's Volumizer 2 API is used to render the volume, while the rest of the rendering code is standard OpenGL. For an overview of the Volumizer 2 rendering process see [4]. Currently, the VoluViz renderer uses the **vzTMLUTShader** to render the volume, and uses the **vzBlock**[1] to define the volume geometry. Future extensions might include using more advanced geometries (e.g. unstructured tetrahedral meshes).

All OpenGL code is located in classes that inherit from **QGLWidget** [3]. Such a class must re-implement the three virtual functions `initializeGL()`,`paintGL()` and `resizeGL()`. The **Viewer** class inherits from **QGLWidget** and is defined in *Viewer.h*:

[1]The vzBlock object represents a simple axis-aligned cube. By default, the extents of the cube are set to (0,0,0) and (1,1,1)

```
class Viewer : public QGLWidget
{
  Q_OBJECT // must include this if you use Qt signals/slots

public:
  Viewer( QWidget *parent, const char *name);
  virtual ~Viewer(void);
  ...

protected:
    /// No copying allowed
    Viewer(const Viewer& c) : m_vsctrl(0,0), m_vscClipPlane(0,0) {}
    /// No assignment allowed
    Viewer& operator=(const Viewer& rhs) {return *this;}
    ...
    void initializeGL(void);
    void resizeGL( int w, int h );
    void paintGL(void);
    ...
};
```

The VoluViz rendering code is mostly located in the `paintGL()` function. This function renders the scene and is called whenever the widget is redrawn.

```
void Viewer::paintGL()
{
  ...
  // Set Background
  if (m_dataset == NULL)
    return;
  ...
  // Transform to viewport and move the volume to center.
  ...
  // Draw ROI box
  if (m_drawROIbox) {
    drawROIbox();
  }
  // Draw axis
  if (m_axisActive) {
    drawAxis();
  }
  // Clip Plane
  if (m_clipPlaneActive) {
    //setClipPlane
  }

  /*******************************************************************/
  /*       Additional geometry in physical coordinates can be inserted here       */
  /*******************************************************************/

  //  Transform the volume to physical coordinates
  if (m_boundingBoxActive) {
    //drawBoundingbox
  }
  // Render the volume data
  renderVolumeData();
}
```

2.7 File and dataset handling

The *File and dataset handling* controls the loading of the HDF files and creates the textures (**vzParameterVolumeTexture** [4]).

The following files are included in the *File and dataset handling* component:

- HDFLoader.h
- HDFLoader.cpp
- Dataset.h
- Dataset.cpp

2.8 The color table editor

The *color table editor* serves as the user interface to the underlying Volumizer 2 lookup table. It consists of a Qt widget and assorted helper classes.

The following files are included in the *color table editor* component:

- color/CEditor.h

- color/CEditor.cpp
- color/ChannelEditor.h
- color/ChannelEditor.cpp
- color/TFChannel.h
- color/TFChannel.cpp
- color/TFControlPoint.h
- color/TFControlPoint.cpp
- color/VVColorFile.h

CEditor is the main user interface and consists of several Qt widgets and six **ChannelEditor** widgets. Four of these represent each of the HSVA/RGBA channels, while the remaining two show the combined lookup table.

Each of these **ChannelEditors** contain a **TFChannel** object that keeps track of the control points (**TFControlPoint**) and interpolation mode. The TF* classes were "borrowed" from a Volumizer 2 sample application (original source in Volumizer2/src/tfeditor).

The **CEditor** class is able to load and save the lookup table in *VoluViz Color table file Format* (vvcf, see appendix A.2 page 20).

The **CEditor** class combines the individual ChannelEditors into one combined lookup table, that is then passed to the renderer as an array of floats in the range [0, 1] on the form (R, G, B, A, R, G, B, ...). The size of the lookup table array is currently 4*256.

2.9 Misc. dialogs and support code

This category includes many support classes that are used by the VoluViz application.

The following files are included in the *Misc. dialogs and support code* component:

- Subset.h
- Subset.cpp
- StringDialog.h
- StringDialog.cpp
- rotation/vector3.h
- rotation/vector3.cpp
- rotation/matrix3.h

- rotation/matrix3.cpp
- rotation/matrix4.h
- rotation/matrix4.cpp
- rotation/quaternion.h
- rotation/quaternion.cpp
- rotation/transform.h
- rotation/transform.cpp
- rotation/Ball.h
- rotation/Ball.cpp
- rotation/VSController.h
- rotation/VSController.cpp

Subset and **StringDialog** are Qt widgets used by the main application. The **VSController** class is a "virtual trackball" [5] used to rotate the volume and clip plane in the viewer. The rest of the files in the rotation subdirectory are support files for **VSController**.

2.10 Future work

VoluViz should be extended to use more of the advanced features in Volumizer 2. In particular, VoluViz should support data in formats other than **vzBlock**. It may also be desirable to support different Volumizer 2 shaders, other than **vzTMLUTShader**. Possible formats and shader classes are listed in [4].

Volumizer 2 supports several different texture types (byte, short, float, etc.) while VoluViz only supports unsigned byte textures. VoluViz should take advantage of the possibilities present in Volumizer 2, and support several texture types.

Lookup tables are currently restricted to 256 entries. Volumizer 2 is capable of handling larger lookup tables so this limit should be removed.

Users should be able to create animations (movies) of their VoluViz sessions and take snapshots for presentation purposes.

An effort to integrate volume and geometric shapes using Coin has started.

VoluViz should support the HDF5 file format in addition to HDF4.

For a list of known bugs and possible future extensions, see the `todo` file in the VoluViz root directory.

A APPENDIX

A.1 Volume file format

VoluViz loads files in the HDF4 [2] file format developed by NCSA. The HDF4 library comes with its own routines for handling HDF files. VoluViz uses HDF4 *Scientific Data Sets* (SD API) to open and read files. A Scientific Data Set (SDS), is a group of data structures used to store and describe multidimensional arrays of scientific data. A HDF file can contain several Scientific Data Sets.

A VoluViz HDF file contains at least one SDS containing byte data (luminance). Multi-field data sets contain additional SDSs (e.g. luminance-alpha). They may also contain a SDS describing the domain coordinates. The coordinate type[2] (uniform, unstructured, etc.) is stored in a SD attribute called **coord_type**. Note that the SDS containing the coordinate data must be the first SDS in the HDF file. The *HDFLoader.cpp* file contains the VoluViz volume file loading code.

A.2 VoluViz Color table file Format (VVCF)

The VoluViz Color table file Format (.vvcf) is defined in the *VVColorFile.h* located in the *src/color* directory. The latest version, version 2, is included here:

```
/// Magic number for identifying vvcf files.
const unsigned short VVCF_MAGICNUMBER = 0x00008365;

/// Version number for this file format.
const unsigned short VVCF_VERSION = 0x00000002;

/// Identifies either a RGBA or a HSVA colorspace.
enum vvColorSpace
{
  VVCF_CS_RGBA,
  VVCF_CS_HSVA,
};

/// Identifies a colormap type.
/**
 * Identifies a LUMINANCE (one channel), LUMINANCE_ALPHA (two channels) or
 * RGBA (four channels) colormap.
 */
```

[2]Currently, VoluViz only supports uniform grids.

```
enum vvColorMapType
{
  VVCF_RGBA,
  VVCF_LUMINANCE_ALPHA,
  VVCF_LUMINANCE
};

/// Identifies component type.
enum vvComponentType
{
  VVCF_UNSIGNED_BYTE,
  VVCF_UNSIGNED_SHORT,
  VVCF_UNSIGNED_INT,
  VVCF_FLOAT,
  VVCF_BYTE,
  VVCF_SHORT,
  VVCF_INT
};

/// Identifies method of interpolation.
/**
 * Values are interpolated from knots using this interpolation
 * method.
 */
enum vvInterpolation {
  VVCF_INTER_LINEAR,
  VVCF_INTER_SPLINES,
  VVCF_INTER_GAUSSIAN,
  VVCF_INTER_PAINT,

  //VVCF_INTER_COUNT
};

/// VoluViz Color table file Format channel header.
/**
 * Carries metadata for one channel.
 */
struct VVCF_CHANNEL_HEADER
{
  /// Interpolation mode used by the channel.
  vvInterpolation interpolation;
  /// Number of points (knots) in the channel.
  unsigned int numpoints;
};
```

```
/// VoluViz Color table file Format header.
/**
 * This struct defines the VoluViz Color table file Format (.vvcf).
 * This struct is designed to be read from and written to file
 * using binary read/write functions, for example fstream::read()
 * and fstream::write() or the C standard library.
 */
struct VVCF_HEADER
{
  /// Magic number
  unsigned short magic;

  /// Version number
  unsigned short version;

  /// RGBA or HSVA colorspace.
  vvColorSpace colorspace;

  /// Type of colormap
  /**
   * \note Only VVCF_RGBA is supported in this version.
   */
  vvColorMapType colormaptype;

  /// Component type
  /**
   * \note Only VVCF_UNSIGNED_BYTE is supported in this version.
   */
  vvComponentType comptype;

  /// Number of entries int the color map.
  unsigned int numindices;
};
```

A vvcf file consists of a **VVCF_HEADER**, one or more **VVCF_CHANNEL_HEADER**s depending on the value of **colormaptype**, and a number of control points. A control point consists of an integer index and a floating point value. After the last **VVCF_CHANNEL_HEADER**, the control points are stored as a block of **numpoints** indices, and then a block of **numpoints** values. Finally, a block containing the entries in the lookup table is stored. This block consists of **numindices** tuples. The size of these tuples depends on the value of **colormaptype**.

References

[1] Doxygen webpage. http://www.doxygen.org.

[2] HDF webpage. http://hdf.ncsa.uiuc.edu.

[3] Qt webpage. http://www.trolltech.com.

[4] K. JONES. *OpenGL Volumizer^{TM} 2 Programmer's Guide*. Silicon Graphics. http://www.sgi.com/software/volumizer/documents.html.

[5] K. SHOEMAKE. ARCBALL: A user interface for specifying three-dimensinal orientation using a mouse. In *Proceedings of Graphics Interface '92*, pages 151–156, May 1992.

2.11

DISTRIBUTION LIST

FFIBM **Dato:** 2. september 2002

RAPPORTTYPE (KRYSS AV) [✓ X] RAPP ☐ NOTAT ☐ RR	RAPPORT NR. 2002/03449	REFERANSE FFIBM/820/170	RAPPORTENS DATO 2. september 2002
RAPPORTENS BESKYTTELSESGRAD Unclassified		ANTALL EKS UTSTEDT 30	ANTALL SIDER 25
RAPPORTENS TITTEL VOLUVIZ 1.0 REPORT		FORFATTER(E) GAARDER Trond, HELGELAND Anders	
FORDELING GODKJENT AV FORSKNINGSSJEF Bjarne Haugstad		FORDELING GODKJENT AV AVDELINGSSJEF: Jan Ivar Botnan	

EKSTERN FORDELING

ANTALL	EKS NR	TIL
		www.ffi.no

INTERN FORDELING

ANTALL	EKS NR	TIL
14		FFI-Bibl
1		Adm direktør/stabssjef
1		FFIE
1		FFISYS
1		FFIBM
1		FFIN
2		Forfattereksemplerer
9		Restopplag til FFI-Bibl
		ELEKTRONISK FORDELING:
		FFI-veven
		Øyvind Andreassen, FFIBM, OyA
		Thor Gjesdal, FFIBM, ThG
		Jan Olav Langseth, FFIBM, JOL
		Atle Ommundsen, FFIBM, AOm
		B. Anders P. Reif, FFIBM, BRe
		Carl Erik Wasberg, FFIBM, CEW

FFI-K1 Retningslinjer for fordeling og forsendelse er gitt i Oraklet, Bind I, Bestemmelser om publikasjoner for Forsvarets forskningsinstitutt, pkt 2 og 5. Benytt ny side om nødvendig.