Volume Rover\textsuperscript{2.0}

May 29, 2013

Computational Visualization Center
Institute for Computational Engineering and Sciences
\& Department of Computer Sciences
The University of Texas at Austin

http://cvcweb.ices.utexas.edu
Contents

1 Welcome to Volume Rover
   1.1 Installation ................................................. 3
      1.1.1 Dependency with other libraries ........................................ 3
      1.1.2 Build .................................................. 4
   1.2 Tabs ......................................................... 5
      1.2.1 Viewers ................................................ 6
      1.2.2 Properties .............................................. 7
      1.2.3 Data ..................................................... 8
      1.2.4 Threads ............................................... 9
   1.3 File Drop Down Menu ......................................... 9
   1.4 View Drop Down Menu ....................................... 9

2 Tools ......................................................... 11
   2.1 Contrast Enhancement ........................................ 11
   2.2 Anisotropic Diffusion ........................................ 11
   2.3 Bilateral Filtering .......................................... 13
   2.4 GDTV Filtering ............................................. 14
   2.5 Segment Virus Map .......................................... 15
   2.6 Secondary Structure Elucidation ............................ 22
   2.7 Surface Reconstruction (Tight Cocone) ......................... 23
   2.8 High Level Set (HLS) Surface ............................. 23
   2.9 Surface Curation ............................................ 23
   2.10 Pocket/Tunnel Detection ................................... 23
   2.11 Skeletonization ........................................... 23
   2.12 LBIE Mesh Generation ..................................... 26
   2.13 LBIE Quality Improvement ................................ 28
   2.14 Multiple Material Segmentation using Higher-order Level Set (MMHLS) ... 28

A Volume Grid Rover ........................................ 31

B Supported Data Formats .................................... 33
   B.1 Voxel Space and Volume/Real Space .......................... 33
   B.2 RawIV .................................................... 33
B.2.1 Header ................................................................. 33
B.2.2 Data ................................................................. 34
B.3 RawV ................................................................. 35
B.4 Raw Geometry (raw, rawn, rawc, rawnc) ......................... 36
B.5 MRC ................................................................. 36

C License Agreement .................................................. 39
Chapter 1

Welcome to Volume Rover

Volume Rover (or VolRover shortly) is an interactive visualization software package that uses modern programmable graphics hardware to provide combined geometry and volume rendering displays. Volume Rover uses data sub-sampling to overcome hardware limitations and visualize datasets of arbitrary size. It also allows higher-resolution zoomed-in volume navigation.

1.1 Installation

We provide executable files to run Volume Rover in Linux, Windows, and Apple operating systems. You can download them at [http://cvcweb.ices.utexas.edu/cvcwp/](http://cvcweb.ices.utexas.edu/cvcwp/). You can also obtain the source code by contacting Dr. Bajaj at bajaj@ices.utexas.edu.

In order to run, Volume Rover needs a dedicated graphics card; NVIDIA GeForce 3, ATI Radeon 9700, or better.

1.1.1 Dependency with other libraries

Volume Rover 2.0 is using various external libraries for its efficient implementation and computation. If you want to build the source package on your own machine, you need to install all or some of those libraries to build core modules and enable tools which you need to use. Please check the names of libraries and their versions tested that we list in Table 1.1. You can also download some of the libraries in tested version from [here](http://cvcweb.ices.utexas.edu/cvcwp/).

However, the latest source of Volume Rover 2.0 is distributed along with source distributions of all the third-party libraries that are required and tested. The only dependencies that are required to be installed prior to building Volume Rover 2.0 are CMake 2.8.x or higher and Qt4. Although this can make our source package quite heavy, we believe the most tricky part for users is to build and link all the dependencies on their specific platform, thus better to minimize that endeavor. By packing all the dependencies together and providing building environment for them, building Volume Rover 2.0 became relatively easy, we believe. Here is a very few steps to build our source package.
<table>
<thead>
<tr>
<th>Name of the library</th>
<th>Tested version</th>
<th>Project page</th>
<th>Required by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qt</td>
<td>4.8.2</td>
<td><a href="http://qt.nokia.com/products/">http://qt.nokia.com/products/</a></td>
<td>Volume Rover core</td>
</tr>
<tr>
<td>CMake</td>
<td>2.8.x</td>
<td><a href="http://www.cmake.org">http://www.cmake.org</a></td>
<td>Volume Rover core</td>
</tr>
<tr>
<td>Boost</td>
<td>1.14.0</td>
<td><a href="http://www.boost.org">http://www.boost.org</a></td>
<td>Volume Rover core</td>
</tr>
<tr>
<td>GSL</td>
<td>1.15</td>
<td><a href="http://www.gnu.org/software/gsl">http://www.gnu.org/software/gsl</a></td>
<td>Tiling, VolumeGrid, Rover, VolUtils</td>
</tr>
<tr>
<td>HDF5</td>
<td>1.8.5</td>
<td><a href="http://www.hdfgroup.org/HDF5">http://www.hdfgroup.org/HDF5</a></td>
<td>Hierarchical volume handling</td>
</tr>
<tr>
<td>CGAL</td>
<td>4.0</td>
<td><a href="http://www.hdfgroup.org/HDF5">http://www.hdfgroup.org/HDF5</a></td>
<td>Tiling, Segmentation, Curation, Pocket-Tunnel, Skeletonization, Secondary/Secondary structure elucidation, Tightcocone, VolumeGrid, Rover</td>
</tr>
<tr>
<td>FFTW</td>
<td>4.0</td>
<td><a href="http://www.fftw.org">http://www.fftw.org</a></td>
<td>Reconstruction, VolUtils</td>
</tr>
<tr>
<td>Cg (nvidia)</td>
<td>4.0</td>
<td><a href="http://www.nvidia.com/content/devzone/index.html">http://www.nvidia.com/content/devzone/index.html</a></td>
<td>Shaded volume rendering</td>
</tr>
</tbody>
</table>

Table 1.1: External libraries having dependency to Volume Rover

1.1.2 Build

Volume Rover 2.0 uses `cmake` enabling a control of the software compilation process using simple platform and compiler independent configuration files for various compilation environment. Once you have required libraries installed, you should be able to build Volume Rover simply following instructions below:

With `ccmake`

```
$ cd $BUILD-DIR

$ ccmake $SOURCE-DIR
```

In ccmake GUI, set:

- `CMAKE_BUILD_TYPE` to `Release` or `Debug` (this should be consistent for third-party and VolumeRover2)
- `PRE_BUILD` to `ON`

Then configure (press ‘c’) and generate (press ‘g’) to generate makefiles for third-party libraries.

```
$ make
```

Once you have all the libraries built successfully, you need to configure VolumeRover2 using ccmake.

```
$ ccmake $SOURCE-DIR
```

In ccmake GUI, set:
- CMAKE_BUILD_TYPE as same as in the third-party build
- PRE_BUILD to OFF
- CMAKE_INSTALL_PREFIX to your desired installation path

If you configure (press ‘c’) after setting PRE_BUILD to OFF, then cmake will try to set up VolumeRover2 building environment including paths for third-party libraries you just built. If there is no error produced by cmake, then you should be good to go. But you may need to confirm the configuration by reconfiguring (press ‘c’) few more times. If the configuration is done, generate makefiles (press ‘g’) and execute ‘make’.

$ make

Then it should build a binary, VolumeRover2, for you. BUILD-DIR is a location to place files generated during the build and SOURCE-DIR is the location of the project’s root CMakeLists.txt file. BUILD-DIR can be the same as the SOURCE-DIR. Also you can set up all the building environments and options for Volume Rover using ccmake which provide a simple graphics user interface on OSX and Unix type OS. Once you’ve built the project, the binary will be in bin/ and any libraries created during the process will be in lib/.

**Without ccmake**

Also you should be able to turn on/off building options for tools depending on your needs or correct locations of libraries using command line interface. We recommend using ccmake to see all the available cmake options though. Basically you need to do the same things in the case of using ccmake by passing required arguments to cmake command line for proper configuration and makefile generation.

$ cd $BUILD-DIR

$ cmake $SOURCE-DIR -DCMAKE_BUILD_TYPE=Release -DPRE_BUILD=ON

$ make (build third-party)

$ cmake $SOURCE-DIR -DCMAKE_BUILD_TYPE=Release -DPRE_BUILD=OFF -DDESIRED_QT_VERSION=4

$ make (build VolumeRover2)

### 1.2 Tabs

There are four tabs: Viewers, Properties, Data, and Threads. The Viewers tab is the default tab used to view the loaded volumes and geometries. The properties tab is used to display all the stored option variables. The Data tab displays information about each of the loaded datasets. The threads tab displays each of the currently running threads.
1.2.1 Viewers

The main Volume Rover window is shown in Figure 1.1. There are two sliders below each render subwindow. The top one controls the render quality. Moving it all the way to the left selects the lowest render quality while moving it all the way to the right selects the highest render quality. The lowest render quality is sometimes useful for making a dataset more transparent. It is also useful for speeding up drawing when the view of the volume or the transfer function is being manipulated. The bottom slider controls the position of the near clipping plane. If you move it all the way to the left, none of the volume is clipped. Move it all the way to the right and all of the volume is clipped.

A subsampled version of the whole dataset is shown in the subwindow on the right, we call this window the Volume Explorer. The window on the left is called the Subvolume Viewer. The volume contained in the Subvolume Viewer is controlled by the Subvolume Control Box, which appears in the Volume Explorer. The Subvolume Control Box can be moved along an axis by clicking and dragging one of the axes, it can be stretched along an axis by dragging one of the axis end points. See Figure 1.2.

Transfer Function

At the bottom of the Volume Rover main window you will see the Transfer Function bar shown in figure 1.3. This bar allows the user to quickly and easily adjust the color and opacity of the volume. The left/right axis controls color and corresponds with the density values of the image. Left is low density and right corresponds to high density. Instead of using a simple gray-scale color scheme where black is low density and white maps to high density, Volume Rover allows the user to define colors for different density values by right clicking on the Transfer Function and adding a color node. Volume Rover then interpolates the colors between different color nodes. The up/down axis of the Transfer Function controls opacity, with up being opaque and down being translucent. An isosurface
Figure 1.2: The Volume Control Box is contained in the Volume Explorer window on the right of the main Volume Rover screen. This box is used to control what volume is displayed in the Subvolume Explorer.

Figure 1.3: The Volume Rover transfer function. Right clicking anywhere on the Transfer Function brings up the menu to add Alpha, Color, and Isocontour Nodes. These nodes may be moved by clicking and dragging them. They can then be deleted by right clicking on them and selecting Delete.

of a certain isovalue can be displayed by right clicking on the Transfer Function and adding a new Isocontour Node.

An Opacity Function is displayed on the Transfer Function bar. The Opacity Function maps density values to r,g,b,α values (r,g,b color with α opacity). The opacity function is linearly interpolated between Alpha Nodes. The user can move, add, and delete Alpha Nodes to control the opacity function. Producing a good transfer function can be a time consuming process. Volume Rover allows the user to save and load transfer functions by right clicking and selecting either save or load.

A contour spectrum can be computed and displayed by right clicking and selecting Display→Contour Spectrum. Similarly a contour tree can be displayed by right clicking and selecting Display→Contour Tree.

1.2.2 Properties

The properties tab lists all the various properties and their settings currently stored in Volume Rover, see figure 1.4. This is a sort of a catch-all for every option in Volume Rover, and serves as a poor-man’s GUI to allow the user to view and modify options that may not
Figure 1.4: The Properties tab in the Volume Rover main window. We’ve selected the property that controls the projection mode for the zoomed in volume. You can see that the mode is currently set to perspective.

Figure 1.5: The Data tab.

be accessible via a dedicated GUI. To change a value in this tab, simply double click on any value and type the new value. This is especially useful for developers to be able to drop in a new property and enter values without having to design a dedicated GUI. All properties can be manipulated by changing them directly within this tab, but many do have dedicated GUIs which can be used. An example is the thumbnail.isocontouring.method property. It can be changed in the properties tab or by clicking to View→Set Geometry View Options and changing the isocontouring method for the thumbnail drop down menu.

1.2.3 Data

The data tab lists all the stored datasets in Volume Rover. The Object Key is a user friendly label, such as a file name or a user defined name. The Type is the type of type of c++ object that the data exists as in memory. On the right half of the tab is a space that may be used to display information about each dataset. See figure 1.5.
1.2.4 Threads

The threads tab may be used when using parallel processing to view the different threads that are currently running.

1.3 File Drop Down Menu

- **Open File** Volume Rover can open the following file types.
  - RawIV is a custom CVC volume filetype, see section B.2
  - RawV is a next-generation custom CVC volume filetype, see section B.3
  - MRC is a standard volume filetype used in the sciences.
  - cvc-raw geometry is a custom CVC filetype used to store geometries.

- **Save Data** opens a dialog window to save volume and geometry files.

- **Close** brings up a dialog box to select different volumes, geometries, and other datasets currently loaded and to close them out of Volume Rover's memory.

- **Save Image** brings up a dialog box to save an image from either of the two viewing windows.

1.4 View Drop Down Menu

- **Select Current Volume** When multiple volumes are open this allows the user to switch between them.

- **Set Slice Rendering** Not implemented yet.

- **Set Background Color** sets the background color for the left (zoomed in) window.

- **Set Thumbnail Background Color** sets the background for the right (thumbnail) window.

- **Set Geometry View** gives options for how geometries are displayed on screen.
  - **Solid** sets geometry surfaces to be interpolated smoothly as a solid surface.
  - **Wireframe** sets the geometry to be displayed as a pure wireframe.
  - **Filled Wireframe** sets the geometry to be displayed as a wireframe with each polygon being filled in.
  - **Isocontouring** This pane gives options to turn on/off the isosurface display in either window and lets the user choose between two methods for generating the isosurface.
• **View Wire Cube** turns on/off the wire cube that displays the bounding boxes on screen.

• **Clip Geometry to Bounding Box** Sometimes a loaded geometry can extend beyond the bounding box of the displayed volume, this option controls the display of geometries outside the bounding box.
Chapter 2

Tools

Volume Rover includes a number of tools to process, enhance, and analyze volumetric data, and to produce computational models. All examples in this chapter use datasets downloadable from [http://cvcweb.ices.utexas.edu/software/data/VolRoverExamples.tgz](http://cvcweb.ices.utexas.edu/software/data/VolRoverExamples.tgz).

2.1 Contrast Enhancement

Our method of adaptive contrast enhancement [13] assigns a new value to each pixel according to an adaptive transfer function designed on the basis of the local statistics (local minimum/maximum as well as local average intensity). Contrast enhancement uses a resistor parameter, which ranges from 0 to 1. A smaller resistor value allows finer details to be enhanced. Figure 2.1 shows the result of contrast enhancement.

To run contrast enhancement, click **Tools→Contrast Enhancement** to bring up the control window in figure 2.2. make sure to select the desired volume in the drop down menu at the top. The resistor value must range between 0 and 1. The output volume may either be written out to a new file or stored as a new volume in memory. The Preview Output tab stores the output in memory. If stored in memory, the new volume will have to be displayed by clicking **View→Select Current Volume** and selecting the new volume. This volume can then be saved by selecting **File→Save**, selecting the Volume tab, and selecting the volume to be saved.

2.2 Anisotropic Diffusion

Anisotropic diffusion is used in image processing for its efficiency of smoothing noise while preserving sharp edges [11]. The only parameter for anisotropic diffusion is currently the number of iterations over the dataset. The default of 20 is good for many situations. See Figure 2.3.

To run Anisotropic Diffusion, click **Tools→Anisotropic Diffusion**, then select a volume from the drop down menu at the top of the pop-up window shown in Figure 2.4. The default 20 iterations is a good number, but this can be adjusted based on trial and error. The output
Figure 2.1: Contrast Enhancement on vh4, displaying a slice through the brain. We used the default resistor value of 0.95. The color table is set to a simple grey scale by setting the left-most color node to be black, the right-most color node to be white, and deleting all other color nodes in the middle.

Figure 2.2: Contrast Enhancement pop-up control window
data may either be written out to a new file or stored as a new volume in memory. The Preview Output tab stores the output in memory. If stored in memory, the new volume will have to be displayed by clicking View → Select Current Volume and selecting the new volume. This volume can then be saved by selecting File → Save, selecting the Volume tab, and selecting the volume to be saved.

### 2.3 Bilateral Filtering

Bilateral Filtering [6] is a simple, non-iterative, edge-preserving smoothing scheme. You can apply a bilateral filter to the subvolume’s data by selecting Bilateral Filter from the Tools menu. The three parameters needed to run Bilateral Filtering are Radiometric Sigma, Spacial Sigma, and Filter Radius. The Radiometric Sigma controls the discrimination
Figure 2.5: To demonstrate the smoothing of running bilateral filter, we used the p22 sample dataset available from our website. We used the default parameters of Radiometric Sigma = 200.0, Spatial Sigma = 1.5, Filter Radius = 2.0. Then for display we ran contrast enhancement, see section 2.1, using the default parameter of Resistor = 0.95. The custom color display can be used by right clicking on the transfer function, clicking Open, and choosing p22-9.5A-resized-128.vinay.

power between true features and noises with the assumption that larger pixel intensity value variations are mainly from true features and smaller pixel intensity value variations are contributed by noise. The Spacial Sigma controls the extent of the normal spacial low pass filtering in pixels (where a larger value causes severe smoothing). The Filter Radius controls the number of slices of the volume to filter at once.

Figure 2.5 gives a sample run. To run bilateral filtering, click Tools→Bilateral Filtering to bring up the control window in 2.6. Firstly, make sure the appropriate volume is selected in the drop down bar at the top of the window. The default parameters usually give good results, further details can be found in [6]. The output data may either be written out to a new file or stored as a new volume in memory. The Preview Output tab stores the output in memory. If stored in memory, the new volume will have to be displayed by clicking View→Select Current Volume and selecting the new volume. This volume can then be saved by selecting File→Save, selecting the Volume tab, and selecting the volume to be saved.

2.4 GDTV Filtering

GDTV (Generalized Digitized Total Variation) [17] is an efficient 3D volume filter method. GDTV keeps more features than bilateral filtering, while also smoothing better than Digitized
Total Variation (DTV). It has four parameters to control the tradeoff between feature conservation and smoothness.

- **Number of Iterations**
- **Exponent** is the power of a polynomial, usually near to 1 is a good choice.
- **Penalty Factor** controls the error tolerance.
- **Neighborhood Size** If Neighbourhood is 0, we just use 1 neighbourhood. Otherwise the values are for $\sqrt{3}$ neighbourhood.

To run GDTV click **Tools→GDTV** and the control window in figure 2.8 will pop up. Make sure the desired volume is selected in the drop down menu at the top of the window. The output volume may either be written out to a new file or stored as a new volume in memory. The Preview Output tab stores the output in memory. If stored in memory, the new volume will have to be displayed by clicking **View→Select Current Volume** and selecting the new volume. This volume can then be saved by selecting **File→Save**, selecting the Volume tab, and selecting the volume to be saved.

## 2.5 Segment Virus Map

Volume Rover offers tools to filter a virus map and segment the subunits and monomers [6]. Selecting **Tools→Segment Virus Map** will bring up the window in figure 2.8.

In running virus segmentation, we have the option to run the process either locally or on a remote machine. If remote segmentation is being used, the hostname, port, and remote
Figure 2.7: We produced this figure using the p22-9.5A-resized-128 example dataset. We ran GDTV filtering with iterations = 3, exponent = 0.8, penalty factor = 0.0001, and neighborhood size = 1.

Figure 2.8: GDTV pop-up control window
file name are required. If local segmentation is being used then the volume must be selected from the Volume Data List drop down bar.

The next option to choose is whether to segment the Capsid, Monomer, or Subunit. Capsid segmentation should be run first, then Subunit segmentation, and finally Monomer segmentation (SegSubunit relies on output from SegCapsid, and SegMonomer relies on output from SegSubunit).

**Capsid Segmentation**

- **Single Capsid - distinct** is the simplest segmentation. The capsid must be distinct from the rest of the data (see figure 2.9(A)). This option requires a threshold value (every density value less than this threshold is simply deleted), and a seed point lying in the capsid. The seed point can be found using the grid rover described in appendix A. We also have the option of running anisotropic diffusion described in section 2.2.

- **Single Capsid** is similar to the first option. It uses a more sophisticated segmentation technique, useful when the capsid is not distinct from the interior genomic structure. It requires an additional seed point of the genomic structure, preferably one that is very near the seed point given for the capsid itself, and in an area where it may be difficult for the computer to automatically detect the boundary.

- **Double Capsid - Initial Segmentation** is for viruses that have a double layer capsid. This process must be run after first running one of the Single Capsid segmentation routines. It requires a lower threshold value just like the single capsid routines. It also requires an estimate radius for the inner and outer capsid layers (in voxel coordinates). The radii can be determined using grid rover. Set the slice to the center of the virus and then determine the approximate radii by holding the mouse over the inner and outer capsid shells and observing the coordinates. The center of the virus is (necessarily) the center of the volume, so dividing the dimension of the dataset by half gives the center coordinate of the virus.

- **Double Capsid - Refined Segmentation** takes the same input as the initial segmentation, but it also needs to know whether there are 3-fold, 5-fold, or 6-fold subunits.

**Subunit Segmentation**

SegSubunit has 6 different arguments. Subunit segmentation is performed as follows:

1. **H and K numbers** These numbers come from the formula \( T = H^2 + H \cdot K + K^2 \), \( H \leq K \) where \( T \) is the quasi-symmetry number and \( H \) and \( K \) refer to the distance between two pentamers using the hexamers as coordinates defining a grid system (see figure 2.10). The T-number is often given in the papers describing the virus, but the H and K numbers may also be visually detected. For more information, visit [http://en.wikipedia.org/wiki/Capsid](http://en.wikipedia.org/wiki/Capsid).
Figure 2.9: (A) is a single and distinct capsid, (B) is a single but not distinct capsid, and (C) is a double capsid.

Note: While it is generally understood that $H \leq K$, sometimes setting $H > K$ will improve Volume Rover’s results.

2. **3-fold** If there are triangular subunits to segment at the 3-fold symmetry axes then $3\text{-fold} = 1$, if no then $3\text{-fold} = 0$.

3. **5-fold** If there are pentamers at the 5-fold axes? If yes, $5\text{-fold} = 1$. Otherwise, $5\text{-fold} = 0$.

4. **6-fold** If there are pentamer subunits located at the 6-fold symmetry axes then $6\text{-fold} = 1$, if no then $6\text{-fold} = 0$.

Figure 2.10: Using the hexamers as a grid on the surface of the virus, and using one of the pentamers as an origin, the H and K numbers refer to the coordinates of a neighboring pentamer in this grid system. Image taken from [http://en.wikipedia.org/wiki/Capsid](http://en.wikipedia.org/wiki/Capsid).

The output is as follows:

- `filename_index.rawiv` (a new volume where the densities are assigned a value based on the segmentation index)
- `filename_seg.rawv` (coloring map)
- `filename_3f_subavg.rawiv` or `filename_6f_subavg.rawiv` (averaged subunit)
SegMonomer

SegMonomer has 1 argument, the fold number of a segmented subunit. For example, if the subunit is a trimer, the fold number is 3. If the subunit is a penton, the fold number is 5.

Example using the P22 dataset

With our release of Volume Rover we have included a sample data set of the P22 virus (EMD 1101, resolution 9.5A). Here is a step by step guide to producing good results with this dataset.

1. First run contrast enhancement described in 2.1 using the default Resistor = 0.95 and choose 'File Output' instead of 'Preview Output'. Make sure to give it a recognizable filename, like p22-capsid. Then load the newly created volume (Volume Rover will place the new file in the same directory from which Volume Rover is executed, not in the directory where the original dataset is stored).

2. Open the virus segmentation dialog by clicking Tools→Segment Virus Map. Enter the values, Threshold=120, genomic seed point = (73, 49, 83), capsid seed point = (96, 106, 63), and uncheck the 'Run Diffuse' option (see figure 2.11).

3. Open the new capsid file (it will have the same filename except with ‘capsid’ appended to the end). Click on Tools→Segment Virus Map again, and this time select the Subunit tab. Make sure the capsid volume is selected in the Volume Data List drop down menu and enter the parameters H-num = 2, K-num = 1, 5-fold=1, 6-fold=1, Initial radius = 5 and click OK. The results should look like figure 2.12.

4. Open the new subunit volumes, they will have the same filename as the capsid but with _5f_subunit and _6f_subavg appended to the end of the names. Open the virus segmentation window again and select the Monomer tab. Select the 5-fold subunit from the Volume Data List drop down menu and enter 5 into the Number of Symmetry Folding text field. Press OK to begin the monomer segmentation, then do the same for the 6-fold subunit. The results should look like figure 2.13.
Figure 2.11: P22 capsid segmentation. The parameters used are displayed at top. Bottom left is the resulting capsid volume (opacity of the transfer function has been increased to improve visualization), and bottom right is the slice view in Grid Rover. Note that grid rover will only display the volume that was first loaded, so you will need to restart Volume Rover if you wish to view a 2D slice of the capsid.
Figure 2.12: P22 subunit segmentation. At top are the parameters used, and below are the resulting volumes.

Figure 2.13: P22 monomer segmentation.
Figure 2.14: a) P22 capsid segmented. b) P22 subunits segmented.

2.6 Secondary Structure Elucidation

Volume Rover can determine a protein’s secondary structure by clicking Tools → Secondary Structure Elucidation. There are two methods implemented, a volume method and a point cloud method. The volume method takes the volume itself as input, and the point cloud method takes points from a user-defined isosurface as input.

Volume Based Method

The Volume Based Method searches for cylinder and sheet formations within the volume using the following parameters to guide the process. All units are in voxels, see section B.1. The default values provided when opening up the segmentation window are usually good values.

**Helix Width** is used initially as a minimal width for helices.

**Min Helix Width Ratio** is used in later stages to add a buffer radius around the Helix Width given above. The buffer minimum is given by the formula: \( \text{helixBufferMin} = \text{helixWidth} \times (\text{minHelixWidthRatio}/2) \).

**Max Helix Width Ratio** is similar to the Min Helix Width Ratio, with the formula: \( \text{helixBufferMax} = \text{helixWidth} \times (\text{maxHelixWidthRatio}/2) \).

**Min Helix Length** Gives a minimum length for each helix.

**Sheet Width** is used initially as a minimal width for sheets.

**Min Sheet Width Ratio** is used in later stages to add a buffer radius around the Sheet Width given above. The buffer minimum is given by the formula: \( \text{bufferMin} = \text{sheetWidth} \times \text{minSheetWidth} \).

**Max Sheet Width Ratio** is similar to the Min Sheet Width Ratio except it uses the formula: \( \text{bufferMax} = \text{sheetWidth} \times \text{maxSheetWidth} \times \text{sheetExtent} \).
Sheet Extended is used to determine the min/max sheet width buffer.

**Point Cloud Based Method**

If there is a previously generated set of points, load this file directly. If not then generate a set of points from a loaded volume. To generate a set of points, simply define an isosurface (see figure 1.3). Volume Rover generates a mesh to visualize the isosurface and the vertices of this mesh will be used as the point cloud. Open the Secondary Structure Elucidation window and select the Point Cloud-based tab. Be sure to change the Filename drop down menu to the desired pointset (if generating the points using an isosurface, this will be thumbnail_isocontour) and click OK.

After running some computations, the window at left of figure 2.16 will pop up. This shows a histogram of all the possible alpha-helices and beta-sheets. Check the render box, select a region of the histogram by clicking and dragging, and click the update button. The structures selected in the histogram will be drawn in the viewer windows. The number box above each histogram controls the maximum number of features that will be found.

### 2.7 Surface Reconstruction (Tight Cocone)

Tight Cocone takes a point set and creates a geometric surface wrapping the surface of the point set.

### 2.8 High Level Set (HLS) Surface

HLS Surface can be used to generate a smooth isosurface from a volume with a rough isosurface. First it generates a new volume from the isosurface using a Signed Distance Function. Then using HLS, it smooths the volume. The resulting volume takes an isovalue of 0 to generate a smooth isosurface that matches the original isosurface.

### 2.9 Surface Curation

Surface Curation patches holes and and corrects non-manifold errors, and reduces noise in geometric data.

### 2.10 Pocket/Tunnel Detection

### 2.11 Skeletonization

A simple skeletonization method is being implemented in which the skeletons of each 2D contour contribute 1 or more nodes to a skeletonization of the 3D object. This skeleton will
Figure 2.15: We performed secondary structure elucidation on the sample GroEL subunit dataset using the volume-based approach. Using the default parameters shown at top we found the 16 alpha helices shown bottom right. For comparison with the pbd file, we generated the image at left using PyMOL and pdb data.
Figure 2.16: Secondary Structure Elucidation using the point cloud method. The isovalue used was 109. (You can open the LBIE Mesh Generation window and check the current isovalue under the Outer iso-value text box. A more convenient method should be available in a future version.) We then selected the histogram ranges shown at left to produce the image at right.
be the basis for a labeled, embedded graph representing the various 3D objects that will be used for metric, combinatorial and topological queries on the dataset.

2.12 LBIE Mesh Generation

LBIE Mesh Generation produces volumetric (tetrahedral or hexahedral) and surface (triangular or quad) 3D finite element meshes. For tet/tri meshes a top-down octree subdivision is coupled with a dual contouring method to extract adaptive meshes with correct topology. Edge contraction and smoothing methods are used to improve the mesh quality [20].

For hex/quad meshes a bottom-up octree-based algorithm is applied to select a starting octree level. Then the dual contouring method is used to extract a preliminary uniform quad/hex mesh, which is decomposed into finer quads/hexes adaptively. The positions of all boundary vertices are recalculated to approximate the boundary surface more accurately. Mesh adaptivity is controlled by a feature sensitive error function. Finally, a relaxation based technique is deployed to improve mesh quality [18].

Input Data

At the top of the LBIE Meshing window is a drop down bar listing all the volumes currently loaded in Volume Rover. Select the volume you wish to mesh with this drop down bar.

Error Tolerance

The error tolerance allows for some flexibility in how closely the surface of the mesh matches the isosurface. Setting the Error tolerance closer to zero will increase the number of elements in the mesh. In the case of only one isosurface, the value in the Inner Error Tolerance is not used.
Figure 2.18: We ran LBIE meshing on 1MAH.rawiv, available in our example dataset package. Before meshing, we ran GDTV filtering to improve the smoothness of the generated isosurfaces (see section 2.4), the GDTV parameters were: Iterations = 3, Exponent = 0.8, Penalty Factor = 0.0001, Neighborhood = 1.0. We then ran LBIE with Error Tolerance = 1.2501, Inner Error = 0.0001, Outer isovalue = Inner Isovalue = 8.79322, Mesh Type = Quad, Improvement Iterations = 3, Normal Type = B-Spline Convolution. A histogram of the quad mesh’s jacobian values is displayed at left, with an image of the resulting mesh displayed at right.

Isovalues

LBIE mesh generation uses an isosurface to determine the boundary of a mesh. The outer isovalue determines this isosurface, when meshing a double boundary an inner isovalue is used to produce an inner isosurface.

The isovalue is the primary input used by the mesh generation algorithm, besides the volume itself. If the isovalue generates a jagged isosurface then the mesh quality will deteriorate. Performing smoothing on the volume will improve the smoothness of the isosurface and thereby improve the mesh quality. This can be performed in Volume Rover by using either Bilateral or GDTV filtering, see sections 2.3 or 2.4 respectively.

Mesh Type

- Single: A triangular surface mesh.
- Tetra: A tetrahedral volume mesh (requires one isosurface).
- quad: A quad surface mesh.
- Hexa: A hexahedral volume mesh (requires one isosurface).
- Triangle - Double Surface: A double layered triangular surface mesh of two isosurfaces.
- Tetra - Double Boundary: A tetrahedral volume mesh of the volume between two isosurfaces.
Improvement Iterations

Quality improvement can be run after producing the mesh. But to save time for the user, we offer this option to automatically run quality improvement before producing any output. This variable controls how many iterations of improvement will be automatically run. A description of the algorithm is given in section 2.13.

Normal Type

Surface vertices have a normal vector used to approximate the isosurface. This vector can be computed using one of these three methods:

- B-Spline Convolution
- Central Difference
- B-Spline Interpolation

2.13 LBIE Quality Improvement

LBIE uses geometric flow [19] to reduce noise and improve aspect ratio. Geometric flow relocates surface and interior vertices to improve both surface noise and aspect ratio. Surface noise is improved by relocating surface vertices along the normal direction. To improve aspect ratio, surface vertices are moved along the tangent plane and interior vertices are relocated freely. This method preserves features and is especially suited to biomolecular meshes because surface diffusion flow preserves spheres accurately if the initial surface is close to a sphere.

2.14 Multiple Material Segmentation using Higher-order Level Set (MMHLS)

Multiple material segmentation produces smooth manifold meshes having certain gaps between them using geometric flow [19] and higher-order level set scheme for each domain defined in classification map as an input, or solves intersection problem of series of closely packed meshes in similar way (please refer [16] for the detail). In Volume Rover, two functionalities, `generateRawV` and `generateMesh`, were provided with UI to generate level set fields for each domain and final meshes from them respectively (please see Figure 2.21 for a description of the input parameters in the UI). The `generateRawV` can take either an classification map in `rawiv` format or series of meshes as an input and will store a list of level set volumes as a result. Once you have the level set volumes generated, you can run `generateMesh` to reconstruct meshes from them by loading manifest file stored with the volumes. After setting set of parameters, it will automatically process whole volumes you set.
Figure 2.19: A classification map for a brain data in rawiv file format loaded on Volume Rover.

Figure 2.20: The multi-domain meshes produced using our algorithm from the classification map shown in Figure 2.19. Smooth surface meshes were generated with small tolerance which enables building tightly packed domain decomposition without intersections between domains.

and produce colored meshes for each domain in rawc file format. Here is a simple example of using our MMHLS tool:

1. Generate higher-order level set for each domain using generateRawV option under Tools/MMHLS menu.
   
   **Input case I** A classification map as shown in Figure 2.19.

   **Input case II** Series of meshes having intersection(s) between domains.

2. Reconstruct meshes from the level set volumes stored in the previous stage using generateMesh option under Tools/MMHLS menu. The triangulated surface meshes will be saved as rawc format for colored surfaces as shown in Figure 2.20. Each domain in the result have a small tolerance for gaps.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume name</td>
<td>Selection</td>
<td>Select a classification map loaded into Volume Rover</td>
</tr>
<tr>
<td>Dimension</td>
<td>Integer</td>
<td>Dimension of the reconstructed volume of the higher-order level set. Larger number will be required for a bigger volume to obtain high quality mesh.</td>
</tr>
<tr>
<td>EdgeLength</td>
<td>Float</td>
<td>Control the number of points sampled for level set reconstruction. Small value will generate more points.</td>
</tr>
<tr>
<td>Out Prefix</td>
<td>String</td>
<td>This string will be attached to original volume file name and the counter will be added for the final file name of the each level set volume.</td>
</tr>
</tbody>
</table>

(a) *generateRawV* dialog for an input of classification map

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume name</td>
<td>Selection</td>
<td>Select a volume loaded into Volume Rover. The bounding box of this volume will be used as the one for level set volume and it must enclose all the meshes.</td>
</tr>
<tr>
<td>Dimension</td>
<td>Integer</td>
<td>Dimension of the reconstructed volume of the higher-order level set. Larger number will be required for a bigger volume to obtain high quality mesh.</td>
</tr>
<tr>
<td>EdgeLength</td>
<td>Float</td>
<td>Control the number of points sampled for level set reconstruction. Small value will generate more points.</td>
</tr>
<tr>
<td>Mesh Prefix</td>
<td>String</td>
<td>A prefix of the mesh file name. Volume Rover will try to read foo1.raw, foo2.raw when the prefix was set as foo and MeshStart and MeshEnd were set as 1 and 2 respectively.</td>
</tr>
</tbody>
</table>

(b) *generateRawV* dialog for an input of series of meshes

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manifest File</td>
<td>File</td>
<td>Having information of the level set volumes generated.</td>
</tr>
<tr>
<td>Isovalue Ratio</td>
<td>Float</td>
<td>Ratio of the iso value used for generating meshes from the level set volume. If ratio is 1, then the value in the manifest file will be used.</td>
</tr>
<tr>
<td>Tolerance</td>
<td>Float</td>
<td>This value in [0,1] defines the threshold of gap between domains. Too small value may produce an intersection. For tightly packed domains without intersection, [0, 0.2] could be used.</td>
</tr>
<tr>
<td>Volume threshold</td>
<td>Float</td>
<td>The extracted meshes may include multiple components. If a volume of each component is less than the threshold, the component will be removed.</td>
</tr>
<tr>
<td>MeshStart</td>
<td>Integer</td>
<td>Start mesh index</td>
</tr>
<tr>
<td>MeshEnd</td>
<td>Integer</td>
<td>End mesh index</td>
</tr>
<tr>
<td>Out Prefix</td>
<td>String</td>
<td>This string will be attached to original volume file name and the counter will be added for the final file name of the meshes for each domain</td>
</tr>
</tbody>
</table>

(c) *generateMesh* dialog for mesh reconstruction from the level set volumes

Figure 2.21: A description of the input parameters
Appendix A

Volume Grid Rover

Grid Rover is a tool to view 2D slices of the currently loaded volume. Its current implementation is a relic from a previous version of Volume Rover. We include it in this distribution because it is necessary to use for the virus segmentation in section 2.5. Many of the features have been disabled. The important feature of this tool is the ability to move the mouse over a point and to read the 3D coordinate of that point. You can open grid over by clicking View→Show Volume Grid Rover.

Grid rover uses, by default, the color map currently loaded in Volume Rover, but it is usually better to switch to a grey scale density map by clicking the check box in the bottom right panel. The drag bar below the image scrolls through all the slices. Above the image are three tabs allowing the user to select which dimension to slice through in 3D space. To the right of the image is a voxel coordinates tab. As the mouse moves over the image the current coordinates are displayed here. The image index refers to the actual dataset as a 3D array of values, while the object coordinates refer to the real space in which the object lies. Note that as of version 2.0.4744, the segmentation takes all values in image index space and NOT in object coordinate space.

Alternatively, you can also select the Voxel Info tab to display information about the

Figure A.1: Volume Grid Rover window.
voxel currently underneath the mouse pointer. The RGBA values refer to the color assigned to that point by the transfer function, and the corresponding color name is given. The Value refers to the true value of the dataset while the Mapped Value refers to the value used by Volume Rover, which scales the values it uses to lie between 0 and 255.
Appendix B

Supported Data Formats

B.1 Voxel Space and Volume/Real Space

Volume datasets consist of an array of density values, which represent a discretization of density in volumetric real space. When navigating this volume, there are two natural coordinate systems we may use: voxel coordinates, and volume or real coordinates. The voxel coordinates refer to the indexing of the array of density values and are more closely related to the dataset than to the actual volume. The volume or real coordinates are simply coordinates in \( \mathbb{R}^3 \).

B.2 RawIV

The rawiv data format is used to represent 3D volumetric data of scalar fields defined on a regular grid. A rawiv file is created by adding the header to the raw format. Everything is in big-endian. Big endian is the byte order on Sun, SGI, IBM architectures. Intel’s byte order is little endian. The suffix on the name of a rawiv file is .rawiv.

B.2.1 Header

Order of information is as follows, concatenated contiguously.

\((\text{minX})(\text{minY})(\text{minZ})(\text{maxX})(\text{maxY})(\text{maxZ})(\text{numVerts})(\text{numCells})(\text{dimX})\)
\((\text{dimY})(\text{dimZ})(\text{originX})(\text{originY})(\text{originZ})(\text{spanX})(\text{spanY})(\text{spanZ})\)

- \((\text{minX}, \text{minY}, \text{minZ})\) are the co-ordinates of the 1st voxel.
- \((\text{maxX}, \text{maxY}, \text{maxZ})\) are the co-ordinates of the last voxel.

The mins, and maxs are floats. These define the bounding box of the data in coordinate space.

- \text{numVerts} is the number of vertices in the grid.
  \text{numVerts} = \text{dimX} \times \text{dimY} \times \text{dimZ}
  \text{numVerts} is an unsigned int.
• numCells is the number of cells in the grid.
  \[\text{numCells} = (\text{dimX} - 1) \times (\text{dimY} - 1) \times (\text{dimZ} - 1)\]
  numCells is an unsigned int.

• \text{dimX} = number of vertices in x direction
  \text{dimY} = number of vertices in y direction
  \text{dimZ} = number of vertices in z direction
  The dims are unsigned ints.

• \text{originX} \\
• \text{originY} \\
• \text{originZ}
  The origins are floats.
  The existence of the origin co-ordinates is somewhat of a mystery. Some developers claim the origin co-ordinates are exactly the same as the co-ordinates of the first voxel.

• The spans are the spacing between one vertex and the next along the given description.
  \[\text{spanX} = (\text{maxX} - \text{minX})/(\text{dimX} - 1)\]
  \[\text{spanY} = (\text{maxY} - \text{minY})/(\text{dimY} - 1)\]
  \[\text{spanZ} = (\text{maxZ} - \text{minZ})/(\text{dimZ} - 1)\]
  The spans are all floats.

• The size of a rawiv header is 68 bytes.

• There are a number of fields in the header that are redundant. For example, numVerts = \text{dimX} \times \text{dimY} \times \text{dimZ}. If while reading the rawiv format, you find that numVerts \neq \text{dimX} \times \text{dimY} \times \text{dimZ}, then the appropriate action is to determine that the rawiv file is corrupted.

• A byte is 8 bits. A float is 4 bytes. An unsigned int is 4 bytes. An unsigned short is 2 bytes. A character is a single byte.

B.2.2 Data

The data portion in raw format immediately follows the header. The raw portion of the rawiv file is in binary big-Endian format used to represent 3D volumetric data of scalar fields defined on a regular grid. It is simply a sequence of values. These values can be floats, unsigned shorts, or unsigned chars. The data is listed with the x co-ordinate varying fastest, and z varying slowest. So, in C++ syntax a reader would contain the following code snippet:

```cpp
for (int z=0; z < \text{dimZ}; z++)
  for (int y=0; y < \text{dimY}; y++)
    for (int x=0; x < \text{dimX}; x++)
```
B.3 RawV

The rawv data format is very similar to rawiv. A rawv file is a binary file consisting of a variable length header followed by one or more volumetric scalar fields. The rawv file format was created with multi-variable time varying data in mind. All data including the header is big endian.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Size</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>unsigned int</td>
<td>Magic = 0xBAADBEEF</td>
</tr>
<tr>
<td>4</td>
<td>4×3</td>
<td>unsigned int</td>
<td>XYZ Dimensions</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>unsigned int</td>
<td># of Time steps</td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>unsigned int</td>
<td># of Variables</td>
</tr>
<tr>
<td>24</td>
<td>4×4</td>
<td>float</td>
<td>min X,Y,Z,T</td>
</tr>
<tr>
<td>40</td>
<td>4×4</td>
<td>float</td>
<td>max X,Y,Z,T</td>
</tr>
<tr>
<td>56</td>
<td>1</td>
<td>unsigned char</td>
<td>Variable Type 1</td>
</tr>
<tr>
<td>57</td>
<td>1×64</td>
<td>char</td>
<td>Variable Name 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Volume Data</td>
</tr>
</tbody>
</table>

- Fixed size 121 byte header for single variable datasets
- Variable names are NULL terminated
- Variable types are:
  - 1 for unsigned char (1 byte)
  - 2 for unsigned short (2 bytes)
  - 3 for unsigned int/long (4 bytes)
  - 4 for float (4 bytes)
  - 5 for double (8 bytes)
- RawV is pronounced ”Raw Five”
- Data is stored with the X co-ordinate varying fastest, followed by Y, followed by Z.
- T varies slower than Z
- All of a variable’s time steps are stored contiguously
B.4 Raw Geometry (raw, rawn, rawc, rawnc)

Raw geometry files are simple ASCII files used to represent triangle meshes. They come in four flavors: raw, rawn, rawc, and rawnc. Raw files are just triangles, rawn files are triangles with vertex normals for smooth shading, rawc files are triangles with vertex colors and rawnc files are triangles with vertex normals and vertex colors. See table B.4 for a simple raw file example. The basic file structure is as follows.

```
<numvertices><numtriangles>
<vertex 0>  
::
<vertex n>
<triangle 0>  
::
<triangle m>
EOF
```

The vertices section differs depending on the file type. They are as follows

- **Raw** (type: float)
  
  ```
  <vertX><vertY><vertZ>
  ```

- **Rawn** (type: float)
  
  ```
  <vertX><vertY><vertZ><normX><normY><normZ>
  ```

- **Rawc** (type: float)
  
  ```
  <vertX><vertY><vertZ><colorR><colorG><colorB>
  ```
  colors are in range [0,1]

- **Rawnc** (type: float)
  
  ```
  <vertX><vertY><vertZ><normX><normY><normZ><colorR><colorG><colorB>
  ```
  colors are in range [0,1]

Each line in the triangles section defines a triangle by references to lines in the above vertices section. Indices may start at 0 or 1. VolRover will load a file marginally faster if the indices start at 0. The values in this section are all of type **int**.

```
<vertexIndex><vertexIndex><vertexIndex>
3  1  
0.0 0.0 0.0
1.0 0.0 0.0
0.0 1.0 0.0
0  1  2
```

B.5 MRC

The MRC header has length 1024 bytes.
Table B.1: MRC header format

<table>
<thead>
<tr>
<th>SIZE</th>
<th>DATA</th>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>int</td>
<td>NX</td>
<td>Number of columns (fastest changing in map)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>NY</td>
<td>Number of rows</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>NZ</td>
<td>Number of sections (slowest changing in map)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>MODE</td>
<td>Types of pixel in image</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = Image unsigned bytes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Images signed short integer (16 bits)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Image float</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = Complex short×2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = Complex float×2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>NXSTART</td>
<td>Number of first COLUMN in map (Default = 0)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>NYSTART</td>
<td>Number of first ROW in map (Default = 0)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>NZSTART</td>
<td>Number of first SECTION in map (Default =0)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>MX</td>
<td>Number of intervals along X</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>MY</td>
<td>Number of intervals along Y</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>MZ</td>
<td>Number of intervals along z</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>XLEN</td>
<td>Cell dimensions (Angstroms)</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>YLEN</td>
<td>Cell dimensions (Angstroms)</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>ZLEN</td>
<td>Cell dimensions (Angstroms)</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>ALPHA</td>
<td>Cell Angles (Degrees)</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>BETA</td>
<td>Cell Angles (Degrees)</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>GAMMA</td>
<td>Cell Angles (Degrees)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>MAPC</td>
<td>Which axis corresponds to Columns (1,2,3 for X,Y,Z)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>MAPR</td>
<td>Which axis corresponds to Rows (1,2,3 for X,Y,Z)</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>MAPS</td>
<td>Which axis corresponds to Sections (1,2,3 for X,Y,Z)</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>AMIN</td>
<td>Minimum density value</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>AMAX</td>
<td>Maximum density value</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>AMEAN</td>
<td>Mean density value</td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>ISPG</td>
<td>Space group number (0 for images)</td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>NSYMBT</td>
<td>Number of bytes used for storing symmetry operators</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>NEXT</td>
<td>Number of bytes in extended header</td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>CREATID</td>
<td>Creator ID</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>EXTRA</td>
<td>Not used. All set to zero by default</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIZE</th>
<th>DATA</th>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>short</td>
<td>NINT</td>
<td>Number of integers per section</td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>NREAL</td>
<td>Number of reals per section</td>
</tr>
<tr>
<td>28</td>
<td>EXTRA2</td>
<td></td>
<td>Not used. All set to zero by default</td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>IDTYPE</td>
<td>0=mono, 1=tilt, 2=tilts, 3=lina, 4=lins</td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>LENS</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>ND1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>ND2</td>
<td></td>
</tr>
</tbody>
</table>
MRC header format table...continued from previous page.

<table>
<thead>
<tr>
<th>SIZE</th>
<th>DATA</th>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>short</td>
<td>VD1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>short</td>
<td>VD2</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>float</td>
<td>TILTANGLES</td>
<td>Used to rotated model to match new rotated image</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>XORIGIN</td>
<td>Origin of image</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>YORIGIN</td>
<td>Origin of image</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>ZORIGIN</td>
<td>Origin of image</td>
</tr>
<tr>
<td>4</td>
<td>char</td>
<td>CMAP</td>
<td>Contains ‘MAP’</td>
</tr>
<tr>
<td>4</td>
<td>char</td>
<td>STAMP</td>
<td>Machine stamp</td>
</tr>
<tr>
<td>4</td>
<td>float</td>
<td>RMS</td>
<td>Deviation of map from mean density</td>
</tr>
<tr>
<td>4</td>
<td>int</td>
<td>NLABL</td>
<td>Number of labels being used</td>
</tr>
<tr>
<td>800</td>
<td>char</td>
<td></td>
<td>10 labels of 80 character</td>
</tr>
</tbody>
</table>
Appendix C

License Agreement

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

1. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder (THE COMPUTATIONAL VISUALIZATION CENTER at THE UNIVERSITY OF TEXAS AT AUSTIN) or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

2. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

3. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 2 above, provided that you also meet all of these conditions:

- The modified work must itself be a software library.
- You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful. (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 3d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)
These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from
the Library, and can be reasonably considered independent and separate works in themselves, then this License, and
its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the
same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the
terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part
regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather,
the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the
Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

4. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy
of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary
GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary
GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any
other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License
applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

5. You may copy and distribute the Library (or a portion or derivative of it, under Section 3) in object code or executable
form under the terms of Sections 2 and 3 above provided that you accompany it with the complete corresponding
machine-readable source code, which must be distributed under the terms of Sections 2 and 3 above on a medium
customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access
to copy the source code from the same place satisfies the requirement to distribute the source code, even though third
parties are not compelled to copy the source along with the object code.

6. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being
compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work
of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the
Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is
therefore covered by this License. Section 7 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for
the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially
significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be
true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small
inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is
legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under
Section 7.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of
Section 7. Any executables containing that work also fall under Section 7, whether or not they are linked directly with
the Library itself.

7. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library
to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided
that the terms permit modification of the work for the customer’s own use and reverse engineering for debugging such
modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its
use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright
notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to
the copy of this License. Also, you must do one of these things:

- Accompany the work with the complete corresponding machine-readable source code for the Library including
  whatever changes were used in the work (which must be distributed under Sections 2 and 3 above); and, if
  the work is an executable linked with the Library, with the complete machine-readable "work that uses the
  Library", as object code and/or source code, so that the user can modify the Library and then relink to produce
a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

- Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (2) uses at run time a copy of the library already present on the user’s computer system, rather than copying library functions into the executable, and (3) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

- Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 7a, above, for a charge no more than the cost of performing this distribution.

- If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

- Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the ”work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

8. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

- Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

9. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

11. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients’ exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

12. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution
system which is implemented by public license practices. Many people have made generous contributions to the wide
range of software distributed through that system in reliance on consistent application of that system; it is up to the
author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot
impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

13. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted
interfaces, the original copyright holder who places the Library under this License may add an explicit geographical
distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus
excluded. In such case, this License incorporates the limitation as if written in the body of this License.

14. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time
to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new
problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which
applies to it and "any later version", you have the option of following the terms and conditions either of that version
or of any later version published by the Free Software Foundation. If the Library does not specify a license version
number, you may choose any version ever published by the Free Software Foundation.

15. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible
with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation,
write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two
goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of
software generally.

CREDITS

WE REQUEST THAT YOU AGREE TO ACKNOWLEDGE THE USE OF THE
SOFTWARE THAT RESULTS IN ANY PUBLISHED WORK, INCLUDING SCI-
ENTIFIC PAPERS, FILMS AND VIDEOTAPES BY CITING THE REFERENCES
IN CODE FILE AND DOCUMENTATION BOUNDED WITH THE SOFTWARE.

C. Bajaj, Z. Yu, M. Auer
Volumetric Feature Extraction and Visualization of Tomographic Molecular Imaging
(pdf)

This software has been developed at the Computational Visualization Center at The
University of Texas at Austin under

Dr Chandrajit Bajaj
Computational Applied Mathematics Chair in Visualization
Professor of Computer Sciences
Director of Computational Visualization Center
The Institute of Computational Engineering and Sciences
The University of Texas at Austin
201 East 24th Street, ACES 2.324A
1 University Station, C0200
Austin, TX 78712-0027
email: bajaj@ices.utexas.edu
URL: http://www.cs.utexas.edu/users/bajaj/

NO WARRANTY
16. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

17. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

18. WE REQUEST THAT YOU PROVIDE A MENU OR A SCREEN THAT IS TO BE SHOWN WITH OUR CENTER NAME (CVC) AND THE UNIVERSITY OF TEXAS AT AUSTIN TO SHOW THAT WE ARE THE PROVIDER OF THE LIBRARY OR SOURCE CODE. AND WE ALSO REQUEST THAT YOU PROVIDE OUR WORLD WIDE WEB ADDRESS (HTTP://WWW.ICES.UTEXAS.EDU/CVC) IN YOUR SOFTWARE.

19. IF YOU DESIRE TO USE THIS CODE FOR A PROFIT VENTURE, OR IF YOU DO NOT WISH TO ACCEPT THIS LGPL, BUT DESIRE USAGE OF THIS CODE, PLEASE CONTACT CHANDRAJIT BAJAJ AT THE ADDRESS ABOVE FOR A DIFFERENT LICENSE.

END OF TERMS AND CONDITIONS

43
Bibliography


44


[21] Yongjie Zhang, Chandrajit Bajaj, and Guoliang Xu. Surface smoothing and quality improvement of quadrilateral/hexahedral meshes with geometric flow. In Byron W.