

The LiDAR Viewer Manual

Chris Bowles & Oliver Kreylos

March 24, 2010

Contents

| | | |
|----------|---|-----------|
| 1 | Preface | 1 |
| 1.1 | About this document | 1 |
| 1.2 | Free/open source software | 1 |
| 1.3 | Bug reporting | 1 |
| 2 | What are VRUI and LiDAR Viewer | 2 |
| 2.1 | Introduction to VRUI | 2 |
| 2.2 | Introduction to LiDAR Viewer | 2 |
| 3 | Installing VRUI | 3 |
| 3.1 | System Requirements | 3 |
| 3.2 | Download VRUI | 3 |
| 3.3 | Installing VRUI | 4 |
| 4 | Installing LiDAR Viewer | 5 |
| 4.1 | Download LiDAR Viewer | 5 |
| 4.2 | Installing LiDAR Viewer | 5 |
| 5 | Processing Data for use in LiDAR Viewer | 6 |
| 5.1 | Commands for Pre-Processing LiDAR Data | 6 |
| 5.2 | Pre-Processing example | 7 |
| 5.3 | Processing data with Lidar Illuminator | 8 |
| 5.4 | LidarIlluminator Processing Example | 9 |
| 6 | Using LiDAR Viewer | 10 |
| 6.1 | Setting up Controls | 10 |
| 6.2 | Turning on Illumination & Render Settings | 11 |
| 6.3 | Selecting Points | 11 |
| 6.4 | Primitives & Strike/Dip | 13 |
| 6.4.1 | Intersection of Primitives | 14 |
| 7 | Other Applications in LiDAR Viewer | 16 |
| 7.1 | LiDAR Subtractor | 16 |
| 7.2 | Point Set Similarity | 16 |
| 8 | Miscellaneous Commands | 18 |
| 8.1 | Batch editing large text files | 18 |
| 8.2 | Changing the background color in VRUI | 18 |
| 8.3 | Helpful Commands | 19 |

List of Figures

| | | |
|---|---|----|
| 1 | Terminal print out after performing the "head" command in terminal | 8 |
| 2 | Results of LidarPreprocessor in the terminal | 9 |
| 3 | Examples of both menus that will pop up when setting up navigation controls | 10 |
| 4 | The Render Settings Dialog Box | 11 |
| 5 | Example of menu that will pop up when attempting to set up selecting points | 12 |
| 6 | The Interactive Settings Dialog Menu | 13 |
| 7 | Example of LiDAR Viewer Primitives Menu | 14 |
| 8 | Results of indicating the strike and dip on a point cloud within LiDAR Viewer . . . | 15 |
| 9 | Example of menu that will pop up when assigning the dragger tool | 15 |

1 Preface

1.1 About this document

This document describes how to install the Virtual Reality User Interface (VRUI) and LiDARViewer-2.5. It also provides a tutorial on navigating a dataset within LiDARViewer-2.5. Through the use of pre-packaged datasets, this document also provides information about preparing data for the LiDARViewer-2.5 and then performing certain tasks within the program.

All the programs and pre-packaged datasets are available for download at: [xxxxx.html](#). At present, no download package is available. We will be adding on shortly. If you would like some data immediately please email cjbowles@ucdavis.edu.

While this users manual focuses on using LiDARViewer-2.5 in desktop environments, the software also works in semi-immersive or fully immersive virtual reality (VR) environments, such as Geowalls or CAVEs. Please contact the developers via email at visualizer-software@keckcaves.org for information on how to install and use Vrui-1.0-068 and LiDARViewer-2.5 in such environments. The software offers the same functionality in all supported environments, but its user interface is environment-dependent.

With questions/comments please email Chris at cjbowles@ucdavis.edu

1.2 Free/open source software

Vrui-1.0-068 and LiDARViewer-2.5 are free software. They can be redistributed and/or modified under the terms of the GNU General Public License as published by the Free Software Foundation.

1.3 Bug reporting

For questions or comments about Vrui-1.0-068 and LiDARViewer-2.5, or to report any bugs, please send an email to: visualizer-software@keckcaves.org and specify which program you are reporting about in the subject heading.

2 What are VRUI and LiDAR Viewer

2.1 Introduction to VRUI

Vrui-1.0-068 is a C++ software development toolkit for highly interactive virtual reality applications, with a focus on portability between vastly different virtual reality environments, from laptop or desktop computers to CAVEs and other fully immersive systems. More information about Vrui can be found at <http://idav.ucdavis.edu/~okreylos/ResDev/Vrui> or <http://www.keckcaves.org>.

Vrui-1.0-068's development was supported by the University of California, Davis, by the UC Davis W.M. Keck Center for Active Visualization in the Earth Sciences (KeckCAVES, www.keckcaves.org), and the W.M Keck Foundation.

2.2 Introduction to LiDAR Viewer

Light Detection and Ranging (LiDAR) data provides very high resolution imagery to the remote sensing community. In its simplest form, LiDAR creates a point cloud which is a series of returns (points) that each contain an x , y , z , (location) and usually i (intensity). Generally, the remote sensing community sub-samples and reduces these data to create digital elevation models (DEMs). LiDARViewer-2.5 provides an opportunity for the user to view point cloud datasets without sub-sampling or reducing the data. The program will load in a point cloud and display each individual point from the survey. LiDARViewer-2.5 allows the user to select points and extract them to a separate file, extract primitives (plane, sphere, cylinder) from selected points, determine distance from a plane, and navigate in real-time through large datasets (>2.7 billion points). It is a powerful tool that can provide unique perspectives to LiDAR datasets that are difficult to attain through DEMs.

3 Installing VRUI

The following section will detail how to install Vrui-1.0-068. The commands used during installation will vary greatly with the directory structure that is utilized. A certain directory structure is recommended (especially if using Mac OS X). In this manual, the following formatting will signify: **\$** The following is code to type in a terminal. Do *not* type the "\$" sign
The following is a description of what the code does

3.1 System Requirements

1. Operating System: UNIX, Linux, or Macintosh OS X (Not available for Windows OS)
2. You will need X11, OpenGL, and a C/C++ compiler
3. The Xcode package is not a standard install for the Mac OS and therefore needs to be installed separately. It is available on the installation disks as an optional install. It can also be download from <http://developer.apple.com/tools/download/>. This code will only work with Snow Leopard (OS 10.6). For Mac OS Leopard (OS 10.5) [click here](#)¹.

3.2 Download VRUI

1. Download the following file to the desktop (We will move the file in a later step):
[Click here - http://stout.idav.ucdavis.edu/Vrui-1.0-069.tar.gz](http://stout.idav.ucdavis.edu/Vrui-1.0-069.tar.gz)
2. Open a terminal window²
3. The following will set up the default directory structure. This is recommended for first-time users. Type the following in terminal:

```
$ mkdir ~/src
```

creates a directory called "src" in your home directory
 - To see what your home directory is:

```
$ echo $HOME
```

prints the home directory (probably /Users/YourUserName)
4. Move the Vrui-1.0-068.tar.gz file to this the ~/src/ directory. This is easily done by dragging the file to the ~/src/ folder. However if you want to remain true to the terminal here are the commands needed to move the file:

```
$ mv ~/Desktop/Vrui-1.0-068.tar.gz ~/src/
```

Helpful Tip - when typing the name or directory in terminal, you can press tab before completing the name and if the computer has enough information it will complete the filename or directory. For example:

```
$ cd ~/De
```

At this point, press tab. Desktop should be filled in. This comes in handy when dealing with long and complicated filenames. If not enough information is given, the computer will not fill in the details. For example:

```
$ cd ~/D
```

At this point, press tab. Nothing happens. However, double click tab and the terminal will provide the three options available (Probably Downloads, Documents, Desktop).

¹<https://connect.apple.com/cgi-bin/WebObjects/MemberSite.woa/wa/getSoftware?bundleID=20414>

²On Mac: Applications → Utilities → Terminal

3.3 Installing VRUI

Do the following:

1. `$ cd ~/src`
moves you to the src directory
2. `$ tar zxvf Vrui-1.0-068.tar.gz`
unpacks the VRUI tarball
3. `$ cd Vrui-1.0-068`
moves you to the VRUI directory
4. `$ make && make install`
Compiles the software

Note: The default option is to install the Vrui-1.0 executables, libraries, and configuration files to the `$ HOME/Vrui-1.0` directory. If you want VRUI installed to a different directory, edit the make file before you make and install VRUI. See the README document in the Vrui-1.0-068 directory for more information. This manual assumes you use the default option, if you do not, you will need to adjust the code found in the rest of the document.

The build will take several minutes. If there are error messages during the build, read the README file in the `~/src/Vrui-1.0-068` directory. The support for PNG and JPEG is an important section if errors concerning JPEG or PNG missing files occur. To test that VRUI was installed correctly and your computer is capable of running the program:

5. `$ cd ~/src/Vrui-1.0-068/ExamplePrograms/`
switches to ExamplePrograms directory
6. `$ make`
compiles the programs
7. `$./bin/ShowEarthModel`
runs the Earth Model program

A window should pop up of a 3D earth rotating. *Congratulations*, VRUI is installed correctly.

4 Installing LiDAR Viewer

4.1 Download LiDAR Viewer

1. Download the following files to the desktop (We will move the file in a later step):
 - [Click here - http://stout.idav.ucdavis.edu/LidarViewer-2.5.tar.gz](http://stout.idav.ucdavis.edu/LidarViewer-2.5.tar.gz)
 - [Click here - http://keckcaves.org/software/lidar/manual/UCDWaterTower_LV.tar.gz](http://keckcaves.org/software/lidar/manual/UCDWaterTower_LV.tar.gz)
2. Open a terminal window³
3. Move the LiDARViewer-2.5.tar.gz file to this the ~/src/ directory. This is easily done by dragging the file to the ~/src/ folder. However if you want to remain true to the terminal here are the commands needed to move the file:

```
$ mv ~/Desktop/LiDARViewer-2.5.tar.gz ~/src/  
$ mkdir ~/src/Data  
$ mv ~/Desktop/UCDWaterTower_LV.tar.gz ~/src/Data
```

4.2 Installing LiDAR Viewer

Do the following:

1. `$ cd ~/src`
moves you to the src directory
2. `$ tar zxvf LiDARViewer-2.5.tar.gz`
unpacks the VRUI tarball
3. `$ cd LiDARViewer-2.5`
moves you to the LiDAR Viewer directory
4. `$ make`
Compiles the software

The build will take several minutes. If there are error messages during the build, read the README file in the ~/src/LiDARViewer-2.5 directory. The support for PNG and JPEG is an important section if errors concerning JPEG or PNG missing files occur. If the organizational structure of your computer is different, you will also need to read this document.

If you downloaded the data package for the LiDAR Viewer you may use this data to test to see if LiDAR Viewer is installed correctly.

1. `$ cd ~/src/Data/`
switches to the Data directory
2. `$ tar zxvf UCDWaterTower_LV.tar.gz`
unpacks the Data tarball
3. `$ cd ~/src/LiDARViewer-2.5/bin/`
move to the directory with the LiDAR Viewer executables
4. `./LidarViewer ~/src/Data/UCDWaterTower.LiDAR`
Calls the LiDAR Viewer Program and Loads the Data

A window should pop up showing a point cloud. *Congratulations*, LiDAR Viewer is installed correctly.

³On Mac: Applications → Utilities → Terminal

5 Processing Data for use in LiDAR Viewer

The LidarPreprocessor is what is used to take a LiDAR point cloud file with x , y , z , and, ideally, intensity (i) and convert it into a folder with index and point files that are needed to run the LiDAR Viewer.

The LidarPreprocessor is very flexible with column order, spacing type, and type of file. The types of files with point cloud data supported include:

- .txt** - Text File
- .bin** - Binary files of 3D points with intensity values
- .asc** - ASCII File
- .las** - Standardized binary format for storing LiDAR points⁴
- .oct** - File format from LidarViewer version 1.x

5.1 Commands for Pre-Processing LiDAR Data

You can call the LidarPreprocessor in any directory, but for simplicity in this manual, the LidarPreprocessor will be run from its source directory. `$ cd ~/src/LiDARViewer-2.5/bin/`
In its simplest form the command to run LidarPreprocessor is:

Note: Split into 2 lines due to the length of the command

```
$ ./LidarPreprocessor -np <number of points> -header <number of header lines>  
-o <output file> -c <R> <G> <B> <input file parameter> <input files>
```

-np <number of points> - specifies the maximum number of LiDAR points to store in each octree node. Should be a power of two; 4096 is the recommended number to use.

-header <number of header lines> - specify the number of header lines your text file has.

-o <output file> - specifies the name of the generated LiDAR file. The preprocessor will create a directory of that name and create an index and points file inside that directory. An extension of `.LiDAR` for the folder is recommended to indicate that the folder is specific to the LidarViewer.

-c <red mask> <green mask> <blue mask> - specifies an RGB color mask to apply to all read LiDAR points. Any color read from an input file is component-wise multiplied with this color mask. The initial value, 1.0 1.0 1.0, leaves color values alone; 1.0 0.7 0.7, for example, will give every point a slight red tint. This can also be used to scale from an input file's intensity range to LiDAR Viewer's range of 0-255.

<input file parameter> - The following parameters change the way LidarPreprocessor reads input files, and any parameters affect all following input files, until another parameter is specified again.

-bin - specifies that all following input files are binary files of 3D points with intensity values.

-binrgb - specifies that all following input files are binary files of 3D points with RGB color values.

⁴see <http://www.spatialresources.com/id78.html> for more information

- las** - specifies that all following input files are LAS interchange files.
- oct** - specifies that all following files are LiDAR files in LiDAR Viewer version 1.x format.
- idl** - you don't want to know what this is.
- ascii** **<x>** **<y>** **<z>** **<intensity column>** - specifies that all following files are generic ASCII input files with space- or comma-separated columns. The four additional parameters specify the zero-based column indices of x, y, z, and intensity values, respectively.
- csv** **<x>** **<y>** **<z>** **<intensity column>** - almost the same as -ascii, but using a strict CSV (comma-separated value) file format.
- asciirgb** **<x>** **<y>** **<z>** **<red column>** **<green column>** **<blue column>** - specifies that all following files are generic ASCII input files with space- or comma-separated columns. The six additional parameters specify the zero-based column indices of x, y, z, red, green, and blue values, respectively.
- csvrgb** **<x>** **<y>** **<z>** **<red column>** **<green column>** **<blue column>** - almost the same as -asciirgb, but using a strict CSV (comma-separated value) file format.
- xyzi** - specifies that all following input files are ASCII files with four space-separated columns: X Y Z intensity. This is a shorthand for -ascii 0 1 2 3.
- xyzrgb** - specifies that all following input files are ASCII files with six space-separated columns: X Y Z red green blue. This is a shorthand for -asciirgb 0 1 2 3 4 5.
- auto** - tells LidarPreprocessor to attempt to auto-detect the formats of all following files by their extension, where possible. This works for -bin (.bin), -binrgb (.binrgb), -las (.las), -xyzi (.xyzi), -xyzrgb (.xyzrgb), and -oct (.oct).

<input files> - specifies the location of the files being input. This can be a list of numerous files. For example: If you have numerous .las files in one directory, you can use a wildcard (*.las) and LidarPreprocessor will stitch all the files together.

Important Notes: The column values begin with 0. For example, an ASCII file with the following header:

```
x, y, z, gpstime, intensity, classification, flightline
```

Will be called in the PreProcessor by:

```
-ASCII 0 1 2 4
```

After running the PreProcessor, a new directory will appear that contains two files: 1. Points and 2. Index.

5.2 Pre-Processing example

This section will provide detailed steps to download an example text file and process it for use in the LiDAR Viewer. If you have a point cloud data-set that you would like to process, you can substitute the UCDWaterTower example with the name of your point cloud file.

1. Download this file to the desktop (This *is different* from the file downloaded in the Installing LiDAR Viewer Section):

<http://keckcaves.org/software/lidar/manual/UCDWaterTower.xyzrgb.gz>

2. Type the following:

```
$ mv ~/Desktop/UCDWaterTower.xyzrgb.gz ~/src/Data
```

```
$ gzip -d ~/src/Data/UCDWaterTower.xyzrgb.gz
```

```

geodynamo218:~ bowles$ mv ~/Desktop/UCDWaterTower.xyzrgb.gz ~/src/Data
geodynamo218:~ bowles$ gzip -d ~/src/Data/UCDWaterTower.xyzrgb.gz
geodynamo218:~ bowles$ head ~/src/Data/UCDWaterTower.xyzrgb
-25.353685 208.660507 0.332980 85 85 85
-25.356344 208.667877 0.358294 104 104 104
-25.353342 208.670074 0.449003 92 92 92
-25.350813 208.666946 0.518088 90 90 90
-25.345797 208.671783 0.499568 78 78 78
-25.350731 208.687912 0.404720 90 90 90
-25.341187 208.676666 0.422902 128 128 128
-25.344831 208.697754 0.509609 158 158 158
-25.343817 208.699219 0.436825 156 156 156
-25.340183 208.713089 0.378101 156 156 156

```

Figure 1: Terminal print out after performing the "head" command in terminal

3. Move file and unzip it in the Data folder we created earlier

Due to the large size of point cloud files, it is sometimes difficult for a computer to load the entire file in TextEdit. Therefore, if you need to see what format a particular file is in we can use the "head" command in terminal:

```
$ head ~/src/Data/UCDWaterTower.xyzrgb
```

prints out the top 10 lines of a text file - See **Figure 1**

As you can see the data is space delimited with values: x y z r g b. Therefore, we can use several file input parameters -asciirgb or -xyzrgb or -auto. For simplicity, we will use -auto.

4. \$ cd ~/src/LiDARViewer-2.5/bin/

5. Note: Following line is split into 2 lines due to the length of the command

```
$ ./LidarPreprocessor -np 4096 -o ~/src/Data/UCDWaterTower.LiDAR
-auto ~/src/Data/UCDWaterTower.xyzrgb5
```

this will create a .LiDAR folder in the src/Data/ directory

The print out on the terminal will look like **Figure 2**

6. Now that you have processed the data, you can type the following to start LiDARViewer-2.5:

```
$ ./LidarViewer ~/src/Data/UCDWaterTower.LiDAR
```

This will load the LidarViewer in an X11 window

5.3 Processing data with Lidar Illuminator

The LidarIlluminator will create a normal vector for each point and store those in a file called Normals in the .LiDAR file directory. This allows a user in LidarViewer to toggle on/off an artificial surface on the point cloud. This effectively creates a hillshade in real-time.

In order for the LidarIlluminator to work, you must have a .LiDAR file created from the LidarPreprocessor. Here is the command in the simplest form to run the LidarIlluminator:

```
$ cd ~/src/LiDARViewer-2.5/bin/
```

Note: Following line is split into 2 lines due to the length of the command

```
$ ./LidarIlluminator -cache <cache size> -radius <approximation radius>
```

⁵\$./LidarPreprocessor -np 4096 -o ~/src/Data/UCDWaterTower.LiDAR -auto ~/src/Data/UCDWaterTower.xyzrgb

```

geodynamo218:bin bowles$ ./LidarPreprocessor -np 4096 -o ~/src/Data/UCDWaterTower.LiDAR/
-auto ~/src/Data/UCDWaterTower.xyzrgb
Processing XYZI input file /Users/bowles/src/Data/UCDWaterTower.xyzrgb... Bounding box: [
-28.695913, -12.351953] x [207.474625, 223.299316] x [-0.676296, 39.446491]
Intensity range: 55.000000, 255.000000
done.

Storing 2847920 points as temporary octree... done
Creating octree for 2847920 points
Creating octree... 100.000000% done
Octree contains 4337 nodes with up to 4096 points per node in 8 resolution levels
Processing octree level 0
Processing octree level 1
Processing octree level 2
Processing octree level 3
Processing octree level 4
Processing octree level 5
Processing octree level 6
Processing octree level 7
Octree file sizes are 104116 bytes and 81078948 bytes
Writing final octree files... done
Time to load input data: 1.649024s, time to create octree: 10.845123s, time to write final
octree files: 1.150317s

```

Figure 2: Results of LidarPreprocessor in the terminal

<input file>⁶

-cache <cache size> - Optional command. Set value to half the computer's physical RAM size.

-radius <approximation radius> - Approximately twice the average point spacing in the data in the native units of the data set. **More Information:** When calculating the normal vector for each point, LidarIlluminator finds all nearby points not farther away from the point than <approximation radius>, measured in Euclidean distance. It will then fit a least-squares approximating plane to that set of points using principal component analysis, and store that plane's normal vector for illumination. The approximation radius has to be selected appropriately for a given data set. Getting the best result is a matter of some experimentation. The value uses the measurement units native to the LiDAR file.

-input LiDAR file - The location of the .LiDAR file directory created from the LidarPreprocessor

5.4 LidarIlluminator Processing Example

This example will use the data that was processed in Section 5.2.

```
$ cd ~/src/LiDARViewer-2.5/bin/
```

```
$ ./LidarIlluminator -radius 0.25 ~/src/Data/UCDWaterTower.LiDAR
```

This will create a new file in the .LiDAR/ directory called Normals.

⁶\$./LidarIlluminator -cache <cache size> -radius <approximation radius> <input LiDAR file>

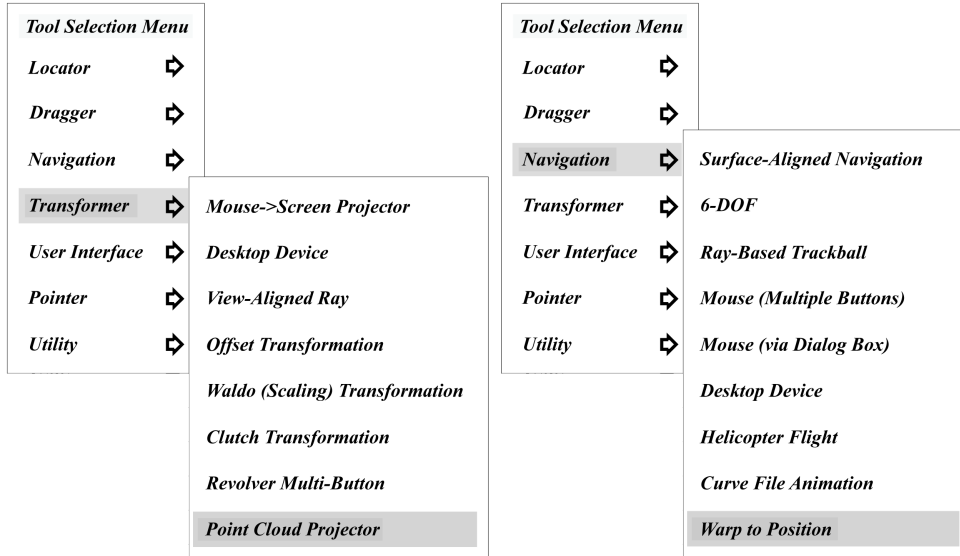


Figure 3: Examples of both menus that will pop up when setting up navigation controls

6 Using LiDAR Viewer

6.1 Setting up Controls

There are numerous ways to control navigation and visualization of the point cloud in LiDAR Viewer, but we recommended using the following control set-up for use on a desktop. In the following example, we will use the data package downloaded and processed in the earlier sections.

```
$ cd ~/src/LiDARViewer-2.5/bin/
$ ./LidarViewer ~/src/Data/UCDWaterTower.LiDAR
Opens the LiDAR Viewer in an X11 window
```

To set up the controls:

- With the cursor inside the X11 window, press and hold the “1” button. A menu pops up. Go to *Transformer* → *Point Cloud Projector* and then let go of button “1”. Press the “1” button again and the same menu pops up. Go to *Navigation* → *Warp to Position* and then go of button “1”. See Figure 3.

The controls to navigate are now setup. The best way to navigate around the data using this setup is to:

1. Move cursor to location of point cloud and press the “1” button. The view will snap to the point cloud where the cursor is located.
2. Use the mouse wheel to zoom in and out. Alternatively, you can press and hold “z” and then click the left mouse button to zoom with finer control.
3. Click and hold the left button with no buttons pressed to rotate the view around the white crosshairs. To move the crosshairs, point to the location where the center should be (on the point cloud) and press “1”.
4. Press and hold “z” but do *not* click the mouse to pan the crosshairs around.
5. Press and hold “shift” and “z” and click the mouse to “zoom” the crosshairs.

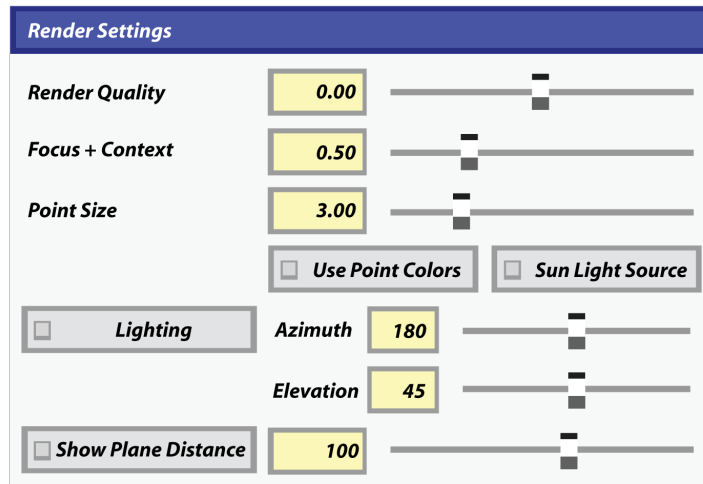


Figure 4: The Render Settings Dialog Box

6.2 Turning on Illumination & Render Settings

LiDAR Viewer Illumination is only available if the point cloud has been processed using LidarIlluminator. See Section 5.3. You can check this by going into the .LiDAR directory of the dataset you would like to view. If there are three files (Index, Points, Normals) then you have processed the data for Illumination. If only Index and Points are in the folder then you need to run LidarIlluminator.

With LiDAR Viewer open, right-click and hold the mouse button to bring up an alternative menu (example of menu in Figure ??). Continue to hold the mouse button down and navigate to *Dialogs* → *Show Render Dialog* and let go of the mouse button. A *Render Settings* dialog box pops up (see Figure 4). The menu controls the following:

- *Render Quality* - Adjusts Default value is 0.00
- *Focus + Context* - Adjusts Default value is 0.50
- *Point Size* - Adjusts the size of each point in the dataset. Default value is 3.
- *Lighting* - Press button to turn on Lighting. This creates an artificial hillshade on the points.
- *Use Point Colors* - Box is checked by default - Turn off/on with Lighting on to show point colors.
- *Sun Light Source* - Default is off with lighting tied to the view. Check this box to turn sun on. Adjust the azimuth by dragging the slide back and forth. Adjust the elevation in a similar fashion.
- *Show Plane Distance* - By checking this box, the points will be colored depending on their distance from either a derived plane from selecting points or a default plane tied to the current view. The slider adjusts the coloring of points by toggling the distance from the plane.

6.3 Selecting Points

Again there are numerous ways to set up controls in LiDAR Viewer, but here we suggest only one method. Before setting this up, first set up the navigation controls following the instructions in

Section 6.1.

To set up the controls for selecting points:

- With the cursor inside the X11 window, press and hold the “2” button. A menu pops up. Go to *Transformer* → *Point Cloud Projector* and then let go of button “2”. Press the “2” button again and the same menu pops up. Go to *Locator* → *6-DOF Locator* and then go of button “2”. See Figure 5.

A yellow sphere pops up and should be locked to the point cloud. Test this by navigating around the point cloud. The yellow sphere is a “paintbrush” and all points inside the sphere will be selected. To select points, move the sphere onto the area of interest and press and hold button “2”. The points within the sphere should turn green.

Right-click and hold the mouse button to bring up the LiDAR Viewer menu. Continue to hold the mouse button down and navigate to *Dialogs* → *Show Interaction Dialog* and let go of the mouse button. An *Interaction Settings* dialog box pops up (see Figure 6). The menu controls the following:

- *Override Tools* - Ignore for now.....
- *Brush Size* - The slider bar adjusts the size of the sphere.
- *Add/Subtrat Buttons* - When “add” button is selected, points are added to selection. When “subtract” button is selected, points are removed from the selection.
- *Neighborhood Size* - This adjusts.....

When the selecting points, you can also right-click and hold the mouse button and navigate to *Selector Modes* → *Add or Subtract* to change the selection mode quickly.

To save the selection, right-click and hold the mouse button and navigate to *Selection* → *Save Selection*. This will output a file called `SelectedPoints.xyzrgb` in the folder where LidarViewer was run from. In the above examples, this would be `~/src/LiDARViewer-2.5/bin/`. To clear the selection follow the steps mentioned in the previous sentence, but select *Clear Selection* instead of *Save Selection*.

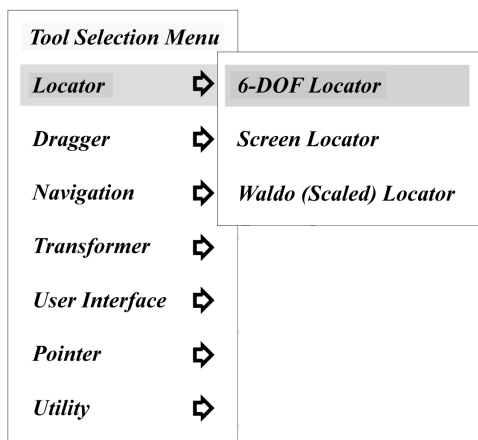


Figure 5: Example of menu that will pop up when attempting to set up selecting points

6.4 Primitives & Strike/Dip

After selecting points you can extract several primitives (Plane, Sphere, Cylinder) and Strike and Dip measurements. The process is very simple. To extract a primitive or S/D measurement, click and hold the right-mouse button and navigate to *Primitives* → *Extract Plane / Sphere / Cylinder* or *Indicate Strike + Dip*. See Figure 7

Extract Plane - Creates a best fit plane to the selected points and prints out the following in terminal:

```
Plane fitting 192999 points
Centroid: (-0.725789, 0.535179, -7.32168)
Normal vector: (0.0318101, -0.0114751, -0.999428)
RMS approximation residual: 0.664162
```

Extract Sphere - Fits a sphere to the selected points and prints out the following in terminal:

```
Sphere fitting 192999 points
Center point: (-0.352979, 0.0308593, -7.80707)
Radius: 6.80815
RMS approximation residual: 0.442413
```

Extract Cylinder - Fits a cylinder to the selected points and prints out the following in terminal:

```
Cylinder fitting 192999 points
Center point: (-0.367986, 0.0359054, -7.27789)
Axis direction: (0.000561941, 0.0671889, 0.99774)
Radius: 6.75066, height: 4.42028
RMS approximation residual: 0.425434
```

Indicate Strike + Dip - Fits a plane to the selected points and prints out the plane fitting equation as described above in terminal. More importantly though, within the X11 window, the dip direction and the dip are provided on screen. The dip direction is in green and by definition is 90° from the strike direction in the positive or negative direction depending on the right-hand rule. The dip is in blue. The results from extracting the strike and dip of a dataset from the Raplee anticline can be seen in Figure 8.

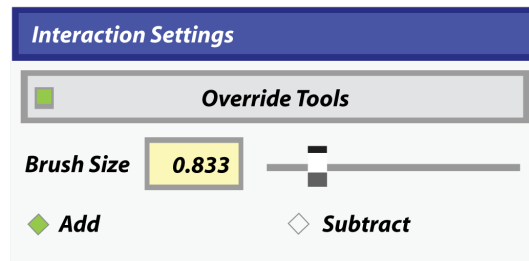


Figure 6: The Interactive Settings Dialog Menu

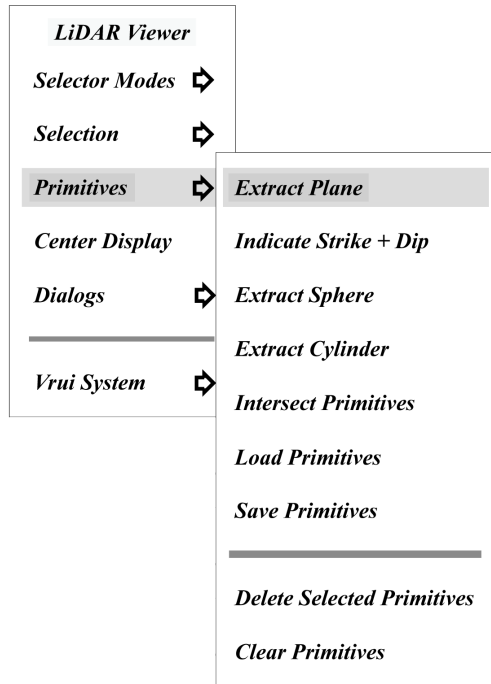


Figure 7: Example of LiDAR Viewer Primitives Menu

6.4.1 Intersection of Primitives

In certain situations you may need to intersect primitives. For example, one can track the deformation of a house after an earthquake by intersecting two planes created from the walls of the house and using the line generated as a deformation marker. After several surveys of the same house, if deformation of the house is occurring, the line will reflect this. In order to intersect primitives you must assign a new tool:

1. With the cursor inside the X11 window, press and hold the “3” button. A menu pops up. Go to *Transformer* → *Point Cloud Projector* and then let go of button “3”. Press the “3” button again and the same menu pops up. Go to *Dragger* → *6-DOF Dragger* and then go of button “3”. See Figure 9.
2. You will need two have created two primitives, preferably planes. To do this see Section 6.4.
3. Move your cursor over one of the primitives and press the “3” button. If done correctly the primitive will turn blue. Repeat for the other primitive.
4. After selecting the primitives, click and hold the right-mouse button and navigate to *Primitives* → *Intersect Primitives*.

If done correctly, a line will appear on screen and the following will be printed out on the terminal

Example - print out after intersecting two planes:

```
Line intersecting two planes
Center point: (-222, -87.9578, -15.1207)
Axis direction: (0.874492, -0.479426, -0.0735885)
Length: 30.1352
```

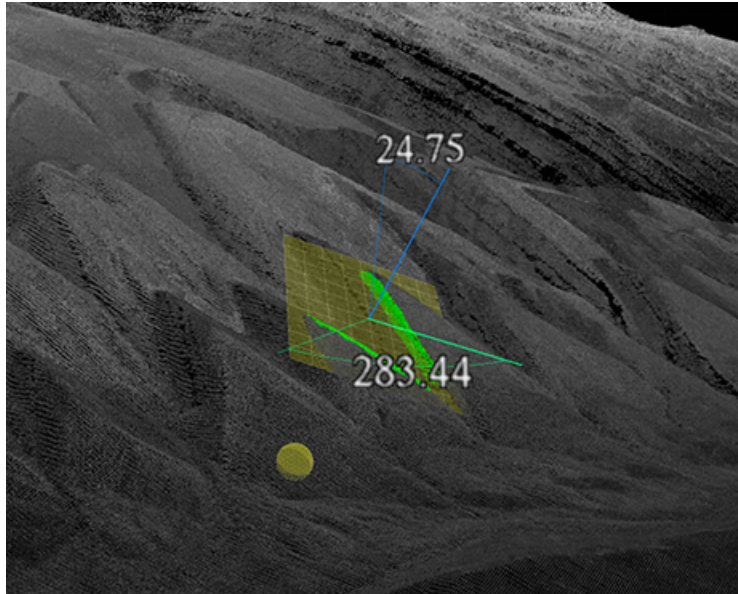



Figure 8: Results of indicating the strike and dip on a point cloud within LiDAR Viewer

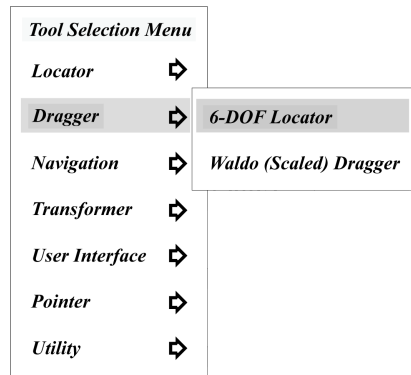


Figure 9: Example of menu that will pop up when assigning the dragger tool

You can try intersecting different types of primitives and different numbers of primitives. If you intersect three planes, LiDAR Viewer will print out the point that intersects the three planes in terminal. If LiDAR Viewer can not intersect the primitives it will print out a warning such as “Did not intersect primitives due to exception mismatching selected primitives”

7 Other Applications in LiDAR Viewer

7.1 LiDAR Subtractor

This tool allows the user to subtract points from a .LiDAR file after a user has selected points from that respective .LiDAR file. This is useful to clip data, manually remove vegetation, or extraneous points from a LiDAR point cloud. The LidarSubtractor will create another .LiDAR file ready for the LidarViewer minus the selected points.

In order for this to work, you will need:

1. The original data file (.LiDAR file)
2. A selection of points from that file (.xyzrgb; see Section 6.3)

The code for the LidarSubtractor is similar to LidarPreprocessor, but has some important differences:

```
$ cd ~/src/LiDARViewer-2.5/bin/
```

Note: Split into 2 lines due to the length of the command

```
$ ./LidarSubtractor -np <number of points> -o <output file> <base .LiDAR file>  
    <selected point set> -eps <epsilon>
```

-np <number of points> specifies the maximum number of LiDAR points to store in each octree node. Show be a power of two; 4096 is the recommended number to use.

-o <output file> - specifies the name of the generated LiDAR file that has the points removed from it.

<base .LiDAR file> - The location of the original .LiDAR file from which the points will be removed.

<selected point set> - The location of the Selected Points file (.xyzrgb) that was created by LidarViewer (see Section 6.3).

-eps <epsilon> - a fudge value to identify matching points. Due to floating-point issues, you need a non-zero epsilon to reliably match points between the base file and the selected set. For airborne LiDAR, a value of 0.01 (1 cm) seems to do the job. For tripod Lidar, you'll want to use a millimeter or even less. Again this value will change with units of original data.

7.2 Point Set Similarity

The PointSetSimilarity tools compares two selected point sets (i.e. .xyzrgb files) and provides a similarity value in percent. For example, during a user-study where different users self-classify points (i.e. vegetation vs. ground) and you need a way to quantify the differences. For this tool to work you will need 2 selected point sets that you created using LiDAR Viewer (see Section 6.3).]

```
$ cd ~/src/LiDARViewer-2.5/bin/
```

```
$ ./PointSetSimilarity <point set 1> <point set 2> <epsilon>
```

<point set 1> - 1st .xyzrgb file

<point set 2> - 2nd .xyzrgb file

<epsilon> - has the same purpose and value as in LidarSubtractor. The only result from running the program is a similarity value in percent. If you compare two identical point sets, you'll get 100%. If the two point sets have not a single point in common, you'll get 0%. The exact formula is:

$$similarity = 2 \times \frac{|A \cap B|}{(|A| + |B|)} \times 100$$

8 Miscellaneous Commands

8.1 Batch editing large text files

The following section will provide one method of editing large text files such as point clouds using ‘awk’, a program available with Linux and Mac OS in the terminal. This can be useful, for example, if you want to add commas to a space delimited file or need to add intensity values because the file you are loading into LidarPreProcessor only contains x , y , and z values. In the following, I will detail how to perform the two previously mentioned examples. After understanding how they work, you should be able to edit a point cloud file for your specific purposes.

In the first example, we will add commas to a space delimited file.

1. Determine the number of variables contained within the text file. For example a text file with, “ $x y z$ ” has 3 variables. Use the “head” command if the text file is large and cannot be loaded by a text editor.

2. Open terminal and type:

```
$ awk '{printf "%s, %s, %s\n", $1, $2, $3}' InputFile.txt > OutputFile.asc
```

3. The only thing that needs to be edited are the numbers of %s and the number of \$#'s. In the above example, there are 3 variables. If there were 5 variables. It would look like this:

```
$ awk '{printf "%s, %s, %s, %s, %s\n", $1, $2, $3, $4, $5}' In > Out
```

In the second example, we will add a uniform intensity value (“1” in this example) to a file that contains x , y , z , and then intensity values of “0”. This can occur if one were to load a point cloud into ArcGIS to clip a section out. ArcGIS will strip the intensity values off and replace them with 0 when exporting to a .txt file. Creating a .LiDAR file with points with intensity values of “0” will make the points black and therefore not visible in LiDAR Viewer with a black background.

In the following example, I will assume you have a file that contains $x y z R G B$ values, but all the $R G B$ values are set to “0”

1. Open terminal and type:

```
$ awk '{printf "%s %s %s 10\n", $1, $2, $3}' InputFile.txt > OutputFile.asc
```

2. This will load the first three columns of the file (\$1, \$2, \$3) and then add a fourth column and insert the value “10” for every point.

8.2 Changing the background color in VRUI

Although this is more related to VRUI than the LiDAR Viewer, it is included in this section because it can be helpful when viewing point clouds. This is an advanced-user feature so use at your discretion.

1. Open terminal and type:

```
$ cd ~/Vrui-1.0/etc/
```

```
open -e Vrui.cfg
```

This opens a VRUI configuration file in a text editor. Note, this file is not in the /src/ directory.

2. In the textfile, find “section Desktop” (should be near the top) then locate the line:

```
backgroundColor (0.0, 0.0, 0.0, 1.0)
```

In detail this mean: backgroundColor(R , G , B , Transparency Value)

3. Change the R, G, B values to taste and save the configuration file. The easiest way to remember the previous background color and be able to quickly change back to it is to comment out the existing line (by adding “#” if front of the line), and add a new line. For example to set the background color to light gray, change:

```
backgroundColor (0.0, 0.0, 0.0, 1.0)
```

4. to

```
# backgroundColor (0.0, 0.0, 0.0, 1.0)
```

```
backgroundColor (0.75, 0.75, 0.75, 1.0)
```

See http://www.web-source.net/216.color_chart.htm for color values.

8.3 Helpful Commands

The terminal is a powerful tool, but requires knowledge of the commands needed to function successfully. The following is a list of commands and suggestions to make your terminal use more efficient: All words after ‘#’ indicate what command does and should not be included in terminal.

```
$ pwd # prints the directory you are presently in
$ ls # prints a list of files and folders in the present directory
$ ls -d */ # prints a list of folders in the present directory
$ cd /directory/ # changes directory to specified directory
$ mkdir directory # creates a new directory (folder)
$ open file # uses default program to open file
$ open -e file # uses Text Editor to open file
$ mv file directory # moves specified file to specified directory
$ cp file directory # copies specified file to specified directory
$ wc -l file # provides the number of lines a txt/ascii file contains
$ rm file #deletes the file CAUTION this is permanent
$ rm -rf folder # deletes a folder and all its contents
```

- Press tab to complete a file name or directory after enough information is entered.
- Press tab twice (quickly) to display a list of files within the directory, or, if part of a file name or directory is typed in it will display the possibilities that will complete the file name or folder.
- The internet is an excellent source of information on this subject if you require more knowledge. A simple search for “Terminal Commands in Mac OS” returns numerous pages with helpful hints.

9 Acknowledgements

The LiDAR data used in the example datasets were provided by Dr. Gerald Bawden from the U.S. Geological Survey (USGS). They were collected using a Tripod mounted LiDAR system.

VRUIs development was supported by the University of California, Davis, by the UC Davis W.M. Keck Center for Active Visualization in the Earth Sciences (KeckCAVES, <http://www.keckcaves.org>), and the W.M. Keck Foundation.

VRUI and all the LidarViewer components are all programmed by Oliver Kreylos, computer scientist at University of California, Davis. Parts of this document are provided by the READMEs and communication with Oliver Kreylos.