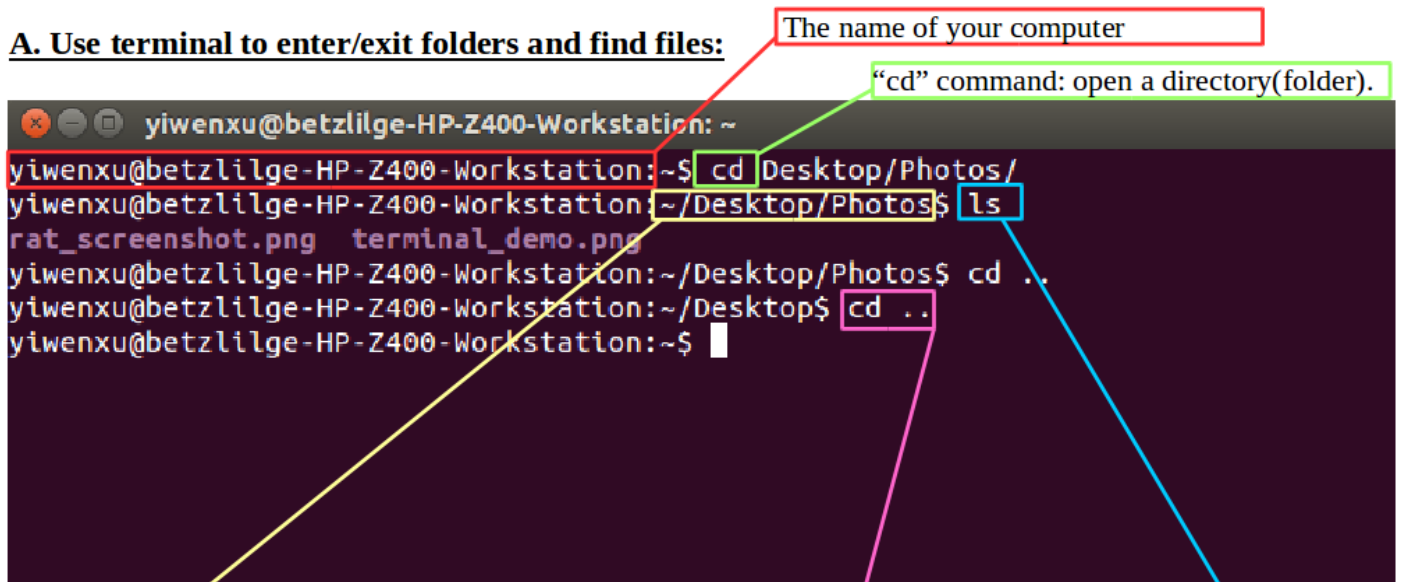


Appendix I – Linux Basics

A. Use terminal to enter/exit folders and find files:



The screenshot shows a terminal window with the following commands and output:

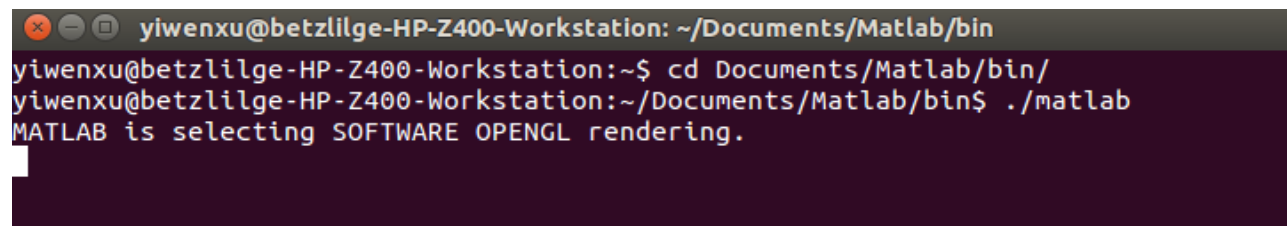
```
yiwenxu@betzlilge-HP-Z400-Workstation: ~  
yiwenxu@betzlilge-HP-Z400-Workstation:~$ cd Desktop/Photos/  
yiwenxu@betzlilge-HP-Z400-Workstation:~/Desktop/Photos$ ls  
rat_screenshot.png terminal_demo.png  
yiwenxu@betzlilge-HP-Z400-Workstation:~/Desktop/Photos$ cd ..  
yiwenxu@betzlilge-HP-Z400-Workstation:~/Desktop$ cd ..  
yiwenxu@betzlilge-HP-Z400-Workstation:~$
```

Annotations:

- The name of your computer
- "cd" command: open a directory(folder).
- Showing the current directory(folder). In this case, it means you are in the "Photos" folder on your desktop. "~" is the short form of your home directory, which is equivalent to "/home/username"
- "cd .." command: go back to the parent directory(folder).
- "ls" command: list all files, folders, programs,etc. in the current directory(folder).

B. Use terminal to execute programs:


Example: open a software named "matlab"



The screenshot shows a terminal window with the following commands and output:

```
yiwenxu@betzlilge-HP-Z400-Workstation: ~/Documents/Matlab/bin  
yiwenxu@betzlilge-HP-Z400-Workstation:~$ cd Documents/Matlab/bin/  
yiwenxu@betzlilge-HP-Z400-Workstation:~/Documents/Matlab/bin$ ./matlab  
MATLAB is selecting SOFTWARE OPENGL rendering.
```

step1: use "cd" command to enter the folder containing matlab.

step2: use "./"+"software name" to execute the program. In this case type "./matlab" in the terminal and hit 

Appendix II – Building From Source

A. Why building from source?

Binary packages: - packages that are already built from source by someone else
- easy to install
- may not have all options from upstream package

Source packages: - take more time to install
- heavy customization option

B. How to build software/library from source?

(The following is an outline for building using Cmake. Please see below the installation of MeshTool as a detailed example.)

step 1: Download source code or clone the source code from Github.

step 2: Go to the directory where you downloaded the source code file in terminal

step 3: Use command “tar zxvf” + “<file name.tar.gz>” to extract the file into a folder. The command may be different for different types of archive files.eg:

```
yiwenxu@betzllilge-HP-Z400-Workstation:~/Downloads$ tar zxvf nrg-dicombrowser-42e8faec05a3.tar.gz
```

step 4: Enter the folder in terminal and create a build-release folder inside it (for the purpose of out-of-source build).

step 5: Enter the build-release folder and run Cmake using command “ccmake ..”

step 6: Configure using Cmake and generate the software/library.

step 7: Make install the software/library.

C. What is Cmake?

CMake is an open-source, cross-platform family of tools designed to build, test and package software. CMake is used to control the software compilation process using simple platform and compiler independent configuration files, and generate native makefiles and workspaces that can be used in the compiler environment of your choice. The suite of CMake tools were created by Kitware in response to the need for a powerful, cross-platform build environment for open-source projects such as ITK and VTK.

We use Cmake to build software/library from source.

FullMonte Software Installation Instruction

Step 1: Install Itk-snap and Paraview

Step 2: Install MeshTool

Step 3: Install FullMonteSW

Note: A virtual machine running ubuntu 16.04 is available at http://www.eecg.utoronto.ca/~vaughn/full_monte/FullMonteSW.ova which facilitates running FullMonteSW since it has all the required packages installed.

Step 1: Install Itk-snap and Paraview

Itk-snap:

Itk-snap is a software application used to segment structures in 3D medical images.



Download link: <http://www.itksnap.org/pmwiki/pmwiki.php?n=Downloads.SNAP3>

Version: 3.6.0 or later

Tutorial: <http://www.itksnap.org/pmwiki/pmwiki.php?n=Train.Sep2014>

Paraview:


ParaView is an open-source, multi-platform data analysis and visualization application. ParaView users can quickly build visualizations to analyze their data using qualitative and quantitative techniques. The data exploration can be done interactively in 3D or programmatically using ParaView's batch processing capabilities.



Download

link:

Releases

Version of ParaView:	<input type="text" value="v5.4"/>
Type of Download:	<input type="text" value="ParaView Binary Installers"/>
Operating System:	<input type="text" value="Linux 64-bit"/>
File to Download:	<input type="text" value="ParaView-5.4.0-Qt5-OpenGL2-MPI-Linux-64bit.tar.gz"/>
<input type="button" value="Download"/> 	

<https://www.paraview.org/download/>

Version:
(5.4 or later)

Tutorial: <https://www.paraview.org/tutorials/>

Step 2: Install MeshTool

I. Download/build All Pre-Requisites

Download Cmake from: <https://cmake.org/download/>. Version 3.1 or later

Download Qt from : <https://download.qt.io/archive/qt/5.3/5.3.2/>.

Download VTK from: <http://www.vtk.org/download/>. Version 6.3.0 or later.

Download Boost from: <http://www.boost.org/>. Recent Version.

Download CGAL from: <https://github.com/CGAL/cgal/releases/tag/releases/CGAL-4.8.1>. Choose “Source Code (tar.gz)”, and build from source. See a detailed building instruction here: <http://doc.cgal.org/latest/Manual/installation.html>.

Download Eigen Library from: http://eigen.tuxfamily.org/index.php?title=Main_Page. Choose latest version and build from source using Cmake.

II. Clone source code of MeshTool and FullMonteSW from Github:

```
~$ git clone http://github.com/zachzzc/MeshTool
~$ cd MeshTool/
~/MeshTool$ mkdir build-release
~/MeshTool$ cd ..
~$ git clone http://github.com/jeffreycassidy/FullMonteSW
~$ cd MeshTool
~/MeshTool$ cd build-release/
~/MeshTool/build-release$ ln -s ~/FullMonteSW/ FullMonte
~/MeshTool/build-release$ cmake ..
```

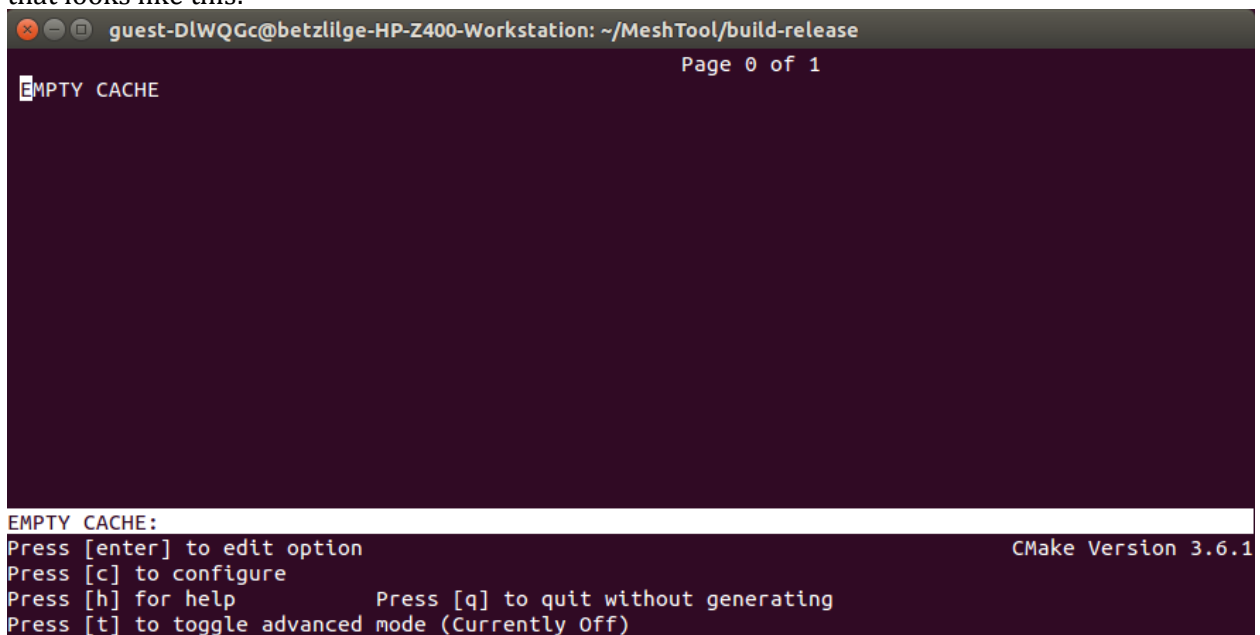
Interpretation of the commands above:

line 1: Clone the MeshTool software package from Github (need internet connection)

line 2: Enter the directory of the package that is cloned onto your computer
line 3: Create a new directory "build-release" inside MeshTool. This directory is where the software will be finally built.
line 4: Go back to home directory
line 5: Clone the FullMonte software package from Github (need internet connection)
line 6: Enter MeshTool directory
line 7: Enter build-release directory (you can combine these two steps into one by using "cd MeshTool/build-release/")
line 8: Make connection between MeshTool and FullMonteSW. To allow MeshTool run properly, FullMonteSW is needed.
line 9: Open CMake Gui

III. Configure and build MeshTool:

After typing the last command "ccmake .." in your terminal, you should be able to see a new window that looks like this:



Press [c] to configure and you will see some error messages. Most of the time, these errors need to be fixed before generating the software.

```
guest-DlWQGc@betzllilge-HP-Z400-Workstation: ~/MeshTool/build-release

CMake Warning at CMakeLists.txt:29 (FIND_PACKAGE):
  By not providing "FindQt5Core.cmake" in CMAKE_MODULE_PATH this project has
  asked CMake to find a package configuration file provided by "Qt5Core", but
  CMake did not find one.

  Could not find a package configuration file provided by "Qt5Core" with any
  of the following names:

    Qt5CoreConfig.cmake
    qt5core-config.cmake

  Add the installation prefix of "Qt5Core" to CMAKE_PREFIX_PATH or set
  "Qt5Core_DIR" to a directory containing one of the above files.  If
  "Qt5Core" provides a separate development package or SDK, be sure it has
  been installed.

Errors occurred during the last pass

CMake Version 3.6.1

Press [e] to exit help
```

Press [e] to go back to the previous window and you will see some entries:

```
guest-DlWQGc@betzllilge-HP-Z400-Workstation: ~/MeshTool/build-release

Page 1 of 1

CMAKE_BUILD_TYPE          *
CMAKE_INSTALL_PREFIX      */usr/local
COMPILER_DISABLE_LOCAL_TYPEDEF *OFF
COMPILER_DISABLE_MISSING_BRACE *ON
Qt5Core_DIR               */tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5Core/
Qt5Gui_DIR                */tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5Gui/
Qt5Widgets_DIR            */tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5Widgets/
```

To fix the error messages in this example, we need to setup all Qt paths that Cmake failed to find manually. Use the arrow keys to choose the directory that you want to change (in this case they are "Qt5Core_DIR, Qt5Gui_DIR, Qt5Widgets_DIR"). Press [enter] to allow edition. And set the path according to the following (Press [enter] again once finish editing):

```
guest-DlWQGc@betzllilge-HP-Z400-Workstation: ~/MeshTool/build-release
Page 1 of 1
CMAKE_BUILD_TYPE          *
CMAKE_INSTALL_PREFIX      */usr/local
COMPILER_DISABLE_LOCAL_TYPEDEF *OFF
COMPILER_DISABLE_MISSING_BRACE *ON
Qt5Core_DIR               */tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5Core/
Qt5Gui_DIR                */tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5Gui/
Qt5Widgets_DIR            */tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5Widgets/
```

the CMake Package_DIR options give the location of PackageConfig.cmake (ie. if exists a file /a/b/c/Qt5Config.cmake then Qt5_DIR=/a/b/c)

The paths in this example. The path is often but not always of the form /a/b/c/lib/cmake

Then press [t] to toggle advanced mode. You will see more options as shown below:

```
guest-DlWQGc@betzllilge-HP-Z400-Workstation: ~/MeshTool/build-release
Page 1 of 4
CMAKE_AR                  /usr/bin/ar
CMAKE_BUILD_TYPE          *
CMAKE_COLOR_MAKEFILE      ON
CMAKE_CXX_COMPILER        /usr/bin/c++
CMAKE_CXX_FLAGS            *
CMAKE_CXX_FLAGS_DEBUG      -g
CMAKE_CXX_FLAGS_MINSIZEREL -Os -DNDEBUG
CMAKE_CXX_FLAGS_RELEASE    -O3 -DNDEBUG
CMAKE_CXX_FLAGS_RELWITHDEBINFO -O2 -g -DNDEBUG
CMAKE_C_COMPILER          /usr/bin/cc
CMAKE_C_FLAGS              *
CMAKE_C_FLAGS_DEBUG        -g
CMAKE_C_FLAGS_MINSIZEREL   -Os -DNDEBUG
CMAKE_C_FLAGS_RELEASE      -O3 -DNDEBUG
CMAKE_C_FLAGS_RELWITHDEBINFO -O2 -g -DNDEBUG
CMAKE_EXE_LINKER_FLAGS     *
CMAKE_EXE_LINKER_FLAGS_DEBUG *

CMAKE AR: Path to a program.
Press [enter] to edit option
Press [c] to configure
Press [h] for help
Press [t] to toggle advanced mode (Currently On)
CMake Version 3.6.1
```

Once in the advanced mode, set “CMAKE_CXX_FLAGS” to “-DBOOST_PARAMETER_MAX_ARITY=12”. And then press [c] to configure again.


```
guest-DLWQGc@betzlilge-HP-Z400-Workstation: ~/MeshTool/build-release
Page 1 of 4

CMAKE_AR                /usr/bin/ar
CMAKE_BUILD_TYPE
CMAKE_COLOR_MAKEFILE
CMAKE_CXX_COMPILER       /usr/bin/c++
CMAKE_CXX_FLAGS          -DBOOST_PARAMETER_MAX_ARITY=12
CMAKE_CXX_FLAGS_DEBUG    -g
CMAKE_CXX_FLAGS_MINSIZEREL -Os -DNDEBUG
CMAKE_CXX_FLAGS_RELEASE  -O3 -DNDEBUG
CMAKE_CXX_FLAGS_RELWITHDEBINFO -O2 -g -DNDEBUG
CMAKE_C_COMPILER         /usr/bin/cc
CMAKE_C_FLAGS            -g
CMAKE_C_FLAGS_DEBUG      -g
CMAKE_C_FLAGS_MINSIZEREL -Os -DNDEBUG
CMAKE_C_FLAGS_RELEASE    -O3 -DNDEBUG
CMAKE_C_FLAGS_RELWITHDEBINFO -O2 -g -DNDEBUG
CMAKE_EXE_LINKER_FLAGS
CMAKE_EXE_LINKER_FLAGS_DEBUG

CMAKE_CXX_FLAGS: Flags used by the compiler during all build types.
Press [enter] to edit option
Press [c] to configure
Press [h] for help
Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently On)
CMake Version 3.6.1
```

If you then see this message or nothing, press [e] and you are almost there. If there is any other error message, please fix it before you proceed to the next step.

```
guest-DLWQGc@betzlilge-HP-Z400-Workstation: ~/MeshTool/build-release

C++ flags: -DBOOST_PARAMETER_MAX_ARITY=12 -Wall -Wno-missing-braces
-frounding-math

CMake produced the following output
Press [e] to exit help
CMake Version 3.6.1
```

You should be able to see a new option [g] by now(If you don't, try to configure one more time).
Press
[g] to generate the software.

```
guest-DlWQGc@betzlilge-HP-Z400-Workstation: ~/MeshTool/build-release
Page 1 of 1
CGAL_CONCURRENT ON
CGAL_DIR /usr/local/lib/CGAL
CMAKE_BUILD_TYPE
CMAKE_INSTALL_PREFIX /usr/local
COMPILER_DISABLE_LOCAL_TYPEDEF OFF
COMPILER_DISABLE_MISSING_BRACE ON
EIGEN3_INCLUDE_DIR /sw/include/eigen3
FULLMONTE_ROOT
Qt5Core_DIR /tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5
Qt5Gui_DIR /tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5
Qt5OpenGL_DIR /tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5
Qt5Widgets_DIR /tmp/guest-DlWQGc/Qt/5.3/gcc_64/lib/cmake/Qt5
VTK_DIR /usr/local/lib/cmake/vtk-6.1

CGAL_CONCURRENT: Enable concurrency in CGAL
Press [enter] to edit option CMake Version 3.6.1
Press [c] to configure Press [g] to generate and exit
Press [h] for help Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently Off)
```

Now the window should have disappeared and you are back to the terminal.

```
guest-DlWQGc@betzlilge-HP-Z400-Workstation: ~/MeshTool/build-release
guest-DlWQGc@betzlilge-HP-Z400-Workstation:~$ cd MeshTool/build-release/
guest-DlWQGc@betzlilge-HP-Z400-Workstation:~/MeshTool/build-release$ MeshGUI
bash: /home/lucywei/Downloads/MeshTool-master/MeshGUI: Permission denied
guest-DlWQGc@betzlilge-HP-Z400-Workstation:~/MeshTool/build-release$ cmake ..

guest-DlWQGc@betzlilge-HP-Z400-Workstation:~/MeshTool/build-release$
```

Type “make MeshGUI” in the terminal and the software will then be generated automatically. Once you see [100%], the software is built successfully.

```
guest-DlWQGc@betzlilge-HP-Z400-Workstation: ~/MeshTool/build-release
guest-DlWQGc@betzlilge-HP-Z400-Workstation:~/MeshTool/build-release$ cmake ..

guest-DlWQGc@betzlilge-HP-Z400-Workstation:~/MeshTool/build-release$ make MeshGUI
Scanning dependencies of target MeshGUI_automoc
[ 4%] Automatic moc and uic for target MeshGUI
Generating moc source moc_CompareMeshDialog.cpp
Generating moc source moc_filedialog.cpp
Generating moc source moc_mainwindow.cpp
Generating moc source moc_offsettable.cpp
Generating moc source moc_vtkwindow.cpp
Generating moc compilation MeshGUI_automoc.cpp
Generating ui header ui_CompareMeshDialog.h
Generating ui header ui_filedialog.h
Generating ui header ui_mainwindow.h
Generating ui header ui_vtkwindow.h
[ 4%] Built target MeshGUI_automoc
Scanning dependencies of target CGAL2VTK_automoc
[ 8%] Automatic moc and uic for target CGAL2VTK
Generating moc compilation CGAL2VTK_automoc.cpp
[ 8%] Built target CGAL2VTK_automoc
Scanning dependencies of target CGAL2VTK
[12%] Building CXX object CMakeFiles/CGAL2VTK.dir/Core/CGALToVTK.cpp.o
[16%] Building CXX object CMakeFiles/CGAL2VTK.dir/CGAL2VTK_automoc.cpp.o
```

Step 3: Install FullMonteSW

Option 1: Run FullMonteSW using virtual machine

A virtual machine running Ubuntu 16.04 is set up with FullMonteSW installed and built. To access this virtual machine, Oracle VM VirtualBox is required. It can be downloaded at:

<https://www.virtualbox.org/wiki/Downloads>

Locate the FullMonteSW.vmdk image file for the virtual machine. Open and clone the virtual machine use VirtualBox.

Note that the image file is quite large (~20Gb) and the virtual machine should be given as much disk space as possible. Under the current setting (2 cores, 4096 Mb base memory, 32 Gb dynamic storage), some of the graphics feature cannot be fully displayed.

Option 2: Install from source

Note: FullMonteSW has been tested both on Ubuntu 14.04 and Ubuntu 16.04.

I. System Pre-Requisites (in addition to those required for MeshTool)

- Modern compiler with C++11 support: G++ version 4.9 or higher, or Clang.
If you have a recent Linux distribution, a newer version of G++ should have already been installed. To check the G++ version on your machine, in terminal, run: `g++ --version`.

Instruction for installing G++ 4.9: <https://askubuntu.com/questions/428198/getting-installing-gcc-g-4-9-on-ubuntu>.

- Tcl 8.5 or 8.6 (for Tcl wrapping)
May need development headers (`tcl8.5-dev` and `tk-dev` packages)
Download Tk package: `sudo apt-get install tk-dev`
Download Tcl package: `sudo apt-get install tcl8.6-dev`
- SWIG 3.0 (for Tcl wrapping)
Download SWIG: `sudo apt-get install swig3.0`

II. Configure and Build FullMonte

Make sure you read the notes below and the file `README.md` in the FullMonteSW package first before configuring or building the software.

The configuring and building process of FullMonte is similar to that of MeshTool. An out-of-source build is recommended, meaning that the object files are created in a folder separate from the source files. A typical developer's use case would have separate Debug and Release builds.

Additional notes:

- It may be necessary to specify paths to packages such as Boost (`-DBOOST_ROOT`). CMake documentation will help with standard packages (VTK, Tcl/Tk, Swig) and compiler settings.
- To enable TCL and VTK interfaces, the `WRAP_VTK` and `WRAP_TCL` values must be set to ON (default is OFF). Users must also modify the paths for external softwares/libraries to local paths on their machine, especially when the install location is non-standard.
- Ensure that build type is set to release for full performance. Architecture should be set appropriately to "AVX" or "AVX2" (Use 'cpuid' command to help choosing AVX vs AVX2)
The processor must support at least AVX.

Table 1. Important variables

Variable	Value	Function
BOOST_ROOT	path	Location of Boost library and header files
WRAP_VTK	ON/OFF	Generate VTK interface code and scripting support
WRAP_TCL	ON/OFF	Generate Tcl interface code
CMAKE_BUILD_TYPE	Release/Debug	Build type (use Release for best performance)
VTK_DIR	path	Folder containing VTKConfig.cmake

An example build script, `'build_fullmonte.sh'`, is provided with the FullMonte software package. By changing the variable paths to local path, the above mentioned steps will be completed automatically.

For FullMonte configuration and build on Mac OS, consult README.md available in the FullMonteSW package.

For further information on cmake, consult documentation on <https://cmake.org>.

Step 4: Run Simulation

There are several example scripts provided with the FullMonte software package. They can be found under `'/FullMonte_build_dir/Examples'`.

To run these examples,

- (1) Go to `'/FullMonte_build_dir/bin'` directory.
- (2) Type `'./tclmonte.sh ../Examples/name_of_example'` at command line.
- (3) Resultant files are placed either in `/bin` or the respective `/Example` directory.

Tutorial on Running Colin27 Examples and Regression Tests

Colin27 is a human brain mesh. Its corresponding mesh and tissue property files are placed under /FullMonte_source_dir/data/MMC/Colin27. In the FullMonte_build_dir, there are several examples available:

- (1) Run FullMonte simulation given a source and output result in terms of fluence/energy.
- (2) Compare FullMonte simulation results with MMC*.
- (3) Perform regression tests to verify correctness of the latest FullMonte result with saved golden copies.

I. Run FullMonte simulation with a point source and output energy data.

Step 1: Open /FullMonte_build_dir/Examples/colin27_energy.tcl and modify the file.

```
# Create and place source (units are mm, referenced to the input geometry)
# This placement was derived by loading the mesh in Paraview and manipulating a point source
Point P
P position "90.77090454101562 170.47908742804466 124.069055680822278"
```

Set the type and location of the source. In this case, a point source is specified at location: (x, y, z) = (90.77090454101562, 170.47908742804466, 124.069055680822278). The unit is millimeter (mm).

```
##### Instantiate and configure simulation kernel
TetraVolumeKernel k

# Kernel properties
k source P
# the source to launch from

k geometry $mesh
# mesh

k materials $opt
# materials

# Monte Carlo kernel properties (standard, unlikely to need change)
k roulettePrWin 0.1
# probability of roulette win

k rouletteWMin 1e-5
# minimum weight "wmin" before roulette takes effect

k maxSteps 10000
k maxHits 10000

# Number of packets to simulate. More -> higher quality, longer run time. Try 10^6 to start.
k packetCount 1000000

# Thread count should be number of cores on the machine, or 2x number of cores with SMT (aka. "Hyperthreading")
k threadCount 8
```

Specify the standard kernel properties and most are not expected to change. The example shown here uses a kernel that outputs data summed over the volume of each tetrahedron. Some key parameters that could be modified include 'source', which is the source type used, and 'packetCount', which is the number of photon packets to launch.

```
VW SetFileName "../Examples/colin27/Data/FullMonte_energy_point_${i}.vtk"
VW SetFileTypeToBinary
VW Update
```

Set the name of the output file. The naming convention is:

Simulator_outputType_sourceType_trialNumber.vtk.

An example would be: "FullMonte_energy_point_0.vtk".

Step 2: Change directory to /FullMonte_build_dir/bin and run:

```
./tclmonte.sh ../Examples/colin27_energy.tcl 1
```

This will run the simulator only once. To run multiple trials, change the last command line option to any desired number.

```
siyunli@betzgrp-pcjeff2:~/FullMonte_RC2/FullMonteSW/Build/bin$ ./tclmonte.sh ../Examples/colin27/colin27_energy.tcl 1
Building faces with Np=70227 Nt=423376
  Done (850209 faces, 6916 on the surface)
423376 tetras, 850209 faces
vtkFullMonteTetraMeshWrapper::mesh(const char* mptr) received type TetraMesh (SWIG string _90dd7a0100000000_p_TetraMesh)
Trial 1 of 1
Building kernel tetras from TetraMesh
Converting 423376 tetras
INFO: Checking faces on 423376 tetras
PointTetraLocator::locate() - Searching for tetra to enclose 90.7709 170.479 124.069
  Tetra 184369 encloses with heights 0.239891 0.78428 0.470195 0.496656 INFO: Point source at 90.7709,170.479,124.069
  in tetra 184369
Sources set up
Done with 423376 tetras
Materials ok
Progress 0.43%Terminated due to unusual number of steps at tetra 368895 dimensionless step remaining=0.0892904 w=1.1258
3e-05
Progress 13.66%Terminated due to unusual number of steps at tetra 356538 dimensionless step remaining=0.00386639 w=1.405
69e-05
Progress 21.72%Terminated due to unusual number of steps at tetra 380795 dimensionless step remaining=0.0148805 w=1.4212
e-05
Progress 32.67%Terminated due to unusual number of steps at tetra 276247 dimensionless step remaining=0.0441978 w=2.4245
9e-05
Progress 56.30%Terminated due to unusual number of steps at tetra 189442 dimensionless step remaining=0.00432738 w=2.851
3e-05
Progress 60.33%Terminated due to unusual number of steps at tetra 390273 dimensionless step remaining=0.00860496 w=1.203
71e-05
Progress 67.23%Terminated due to unusual number of steps at tetra 111855 dimensionless step remaining=0.117468 w=2.51859
e-05
Progress 93.77%Terminated due to unusual number of steps at tetra 388794 dimensionless step remaining=0.0174341 w=1.0171
1e-05
Progress 94.35%Terminated due to unusual number of steps at tetra 323798 dimensionless step remaining=0.0373476 w=2.1767
6e-05
Progress 100.00%
Kernel is done
vtkFullMonteArrayAdaptor::source(const char*) - SWIG pointer of type OutputData
vtkFullMonteArrayAdaptor::update()
Trial 1 of 1 complete
siyunli@betzgrp-pcjeff2:~/FullMonte_RC2/FullMonteSW/Build/bin$
```

The simulation should run and display messages similar to the figure above. Note that there would be some messages like “Terminated due to unusual number of steps...”. These messages should not be a concern as they simply mean that the roulette is initiated and the packet is terminated. The output file is saved as “FullMonte_energy_point_0.vtk” under the directory: /FullMonte_build_dir/Examples/Data/colin27.

II. View simulation results.

Step 0: Complete Example I to generate the ‘FullMonte_energy_point_0.vtk’ file for visualization.

If ParaView is installed (Highly Recommended):

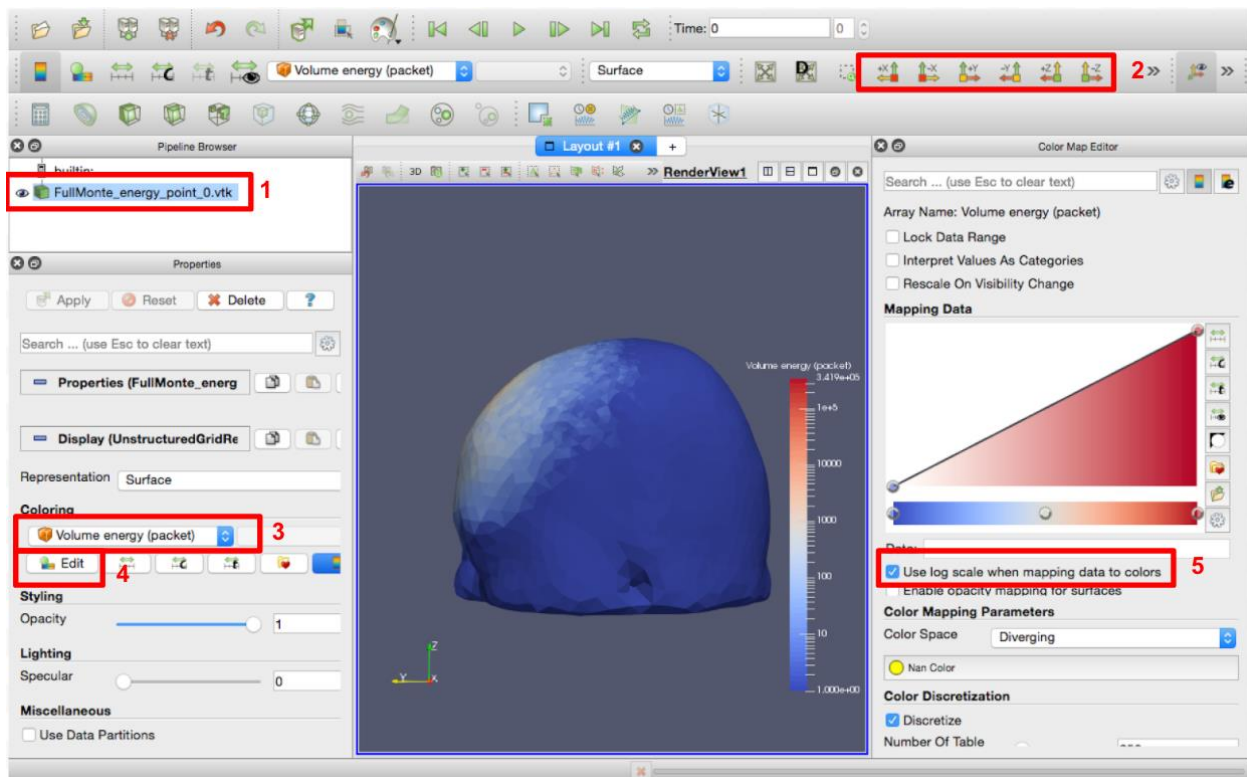
Step 1: In ParaView, go to ‘File’ -> ‘Open’ to load the file ‘FullMonte_energy_point_0.vtk’. The opened file should be in ‘Pipeline Browser’ and make sure that the eye symbol on the left is ON.

Step 2: Click the first button of the panel shown to set view direction to positive x-axis. One may experiment with other view directions using the other buttons.

Step 3: Change the 'Coloring' to 'Volume energy (packet)' to display the simulation result.

Step 4: By default, ParaView uses linear scale to map data to colours. In this example, the range of the energy data is quite large so a logarithmic scale is more appropriate to use. Click on the 'Edit' button to open up the 'Color Map Editor'.

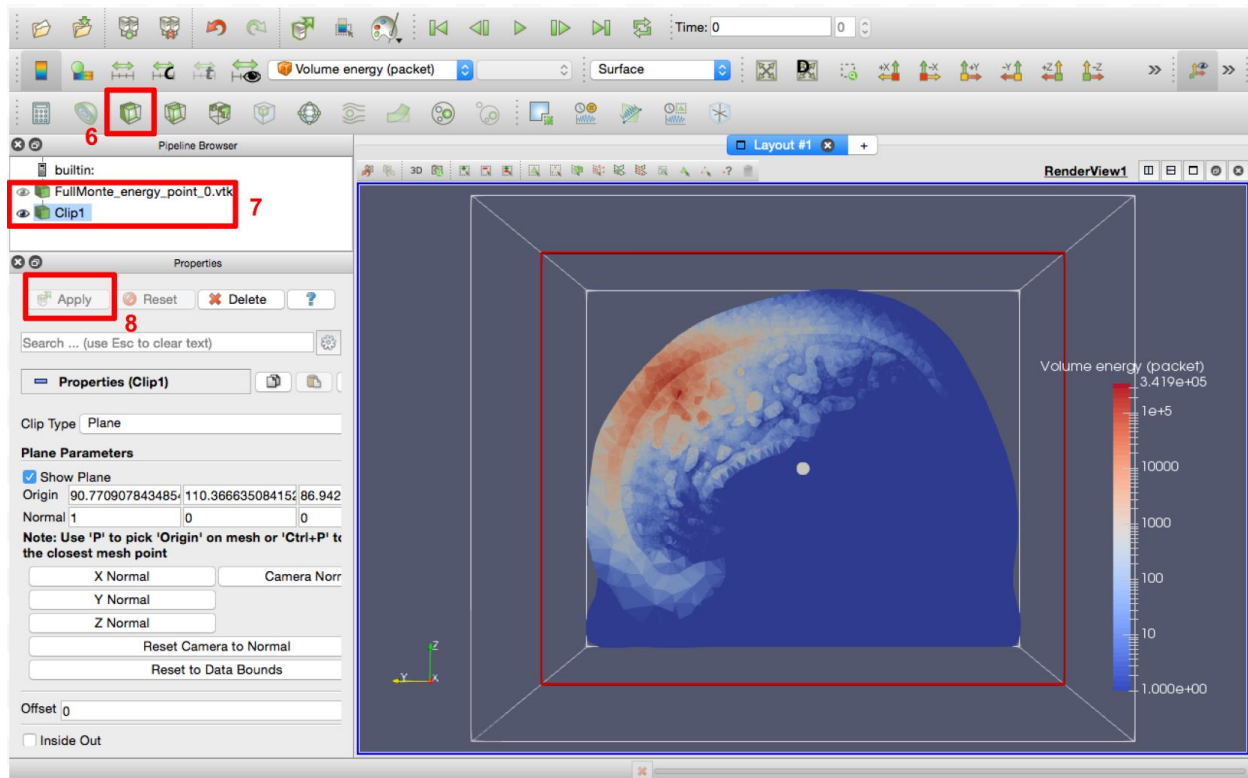
Step 5: In the 'Color Map Editor', check the 'Use log scale when mapping data to colors' option.



Step 6: To view the inside of the mesh model, one can click on the 'Clip' tool.

Step 7: A clip should be created in the 'Pipeline Browser'. Make sure to turn off the eye symbol beside 'FullMonte_energy_point_0.vtk' and turn on the one beside 'Clip 1'.

Step 8: In the 'Properties' window, the 'Apply' button should be green. Click on 'Apply' to apply the changes to the rendered view.



Some basic interactive tools include:

- To rotate the rendered view, click and drag with the left key of the mouse.
- To zoom in/out the rendered view, scroll up/down using the scroll wheel on the mouse.
- To move the mesh model, press and drag using the scroll wheel on the mouse.

ParaView offers a wide variety of filters that can be used to view the data. For further information, please consult the documentation for the proper version of ParaView at:

<http://www.vtk.org/download/>

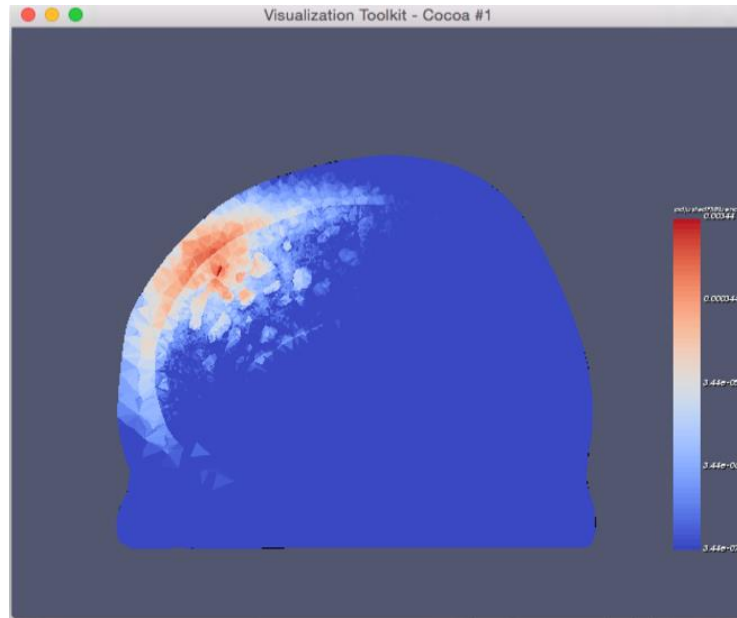
If using the provided C++ class in the FullMonteSW package, to generate an OpenGL window:

Step 1: Change directory to /FullMonte_build_dir/Examples/colin27/FullMonte_visualize/. Make sure that FullMonte_visualize.in has dataType = energy and sourceType = point.

Step 2: In terminal, run the pre-compiled executable:

```
./FullMonte_visualize FullMonte_visualize.in
```

Step 3: The interactive window as shown below should automatically pop up. Use the mouse to rotate/zoom/pan the model.



Step 4: A visualization figure is saved in .png format. File name: FullMonte_energy_point.png
To open the plot file from terminal, type: `xdg-open fileName`. This has the same effect as double-clicking in a file window.

III. Compare FullMonte with MMC, both using point sources and outputting energy.

Step 0: Complete Example I to generate the 'FullMonte_energy_point_0.vtk' file for visualization.

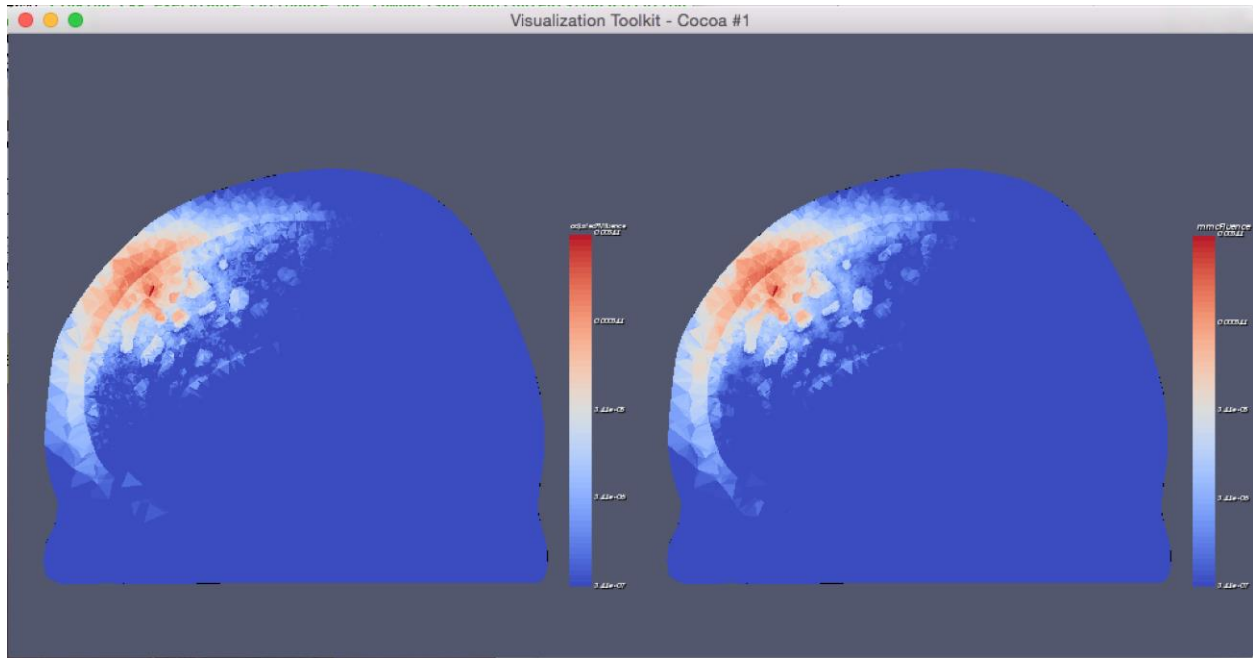
Step 1: Change directory to `/FullMonte_build_dir/Examples/colin27/FullMonte_MMC_point/`. Make sure that `FullMonte_MMC_point.in` contains the following information:

```
1           //N
1e6         //packets
1e5         //topNvalues
1e-8        //threshold
energy      //dataType
point       //sourceType
colin27     //mesh
```

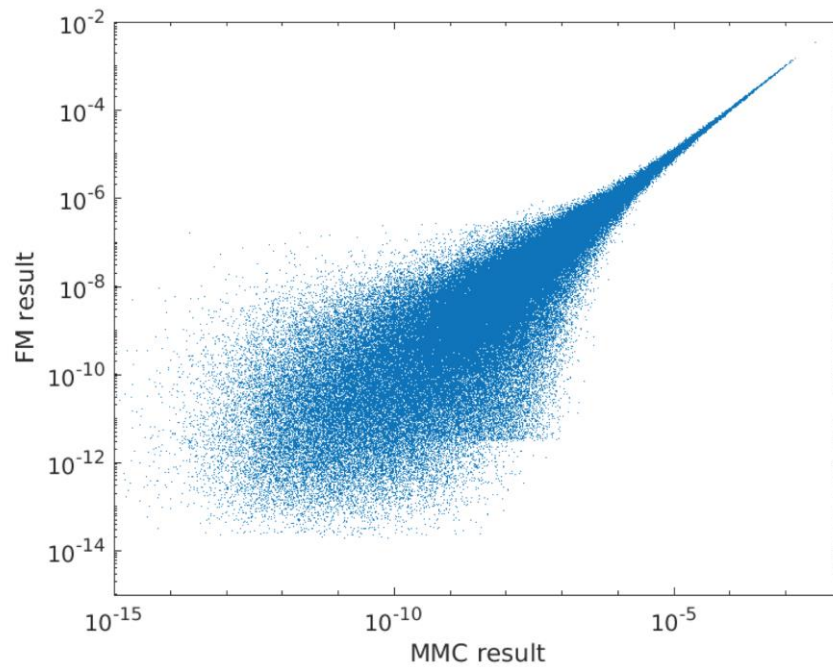
Step 2: In terminal, run the pre-compiled executable:
`./FullMonte_MMC_point FullMonte_MMC_point.in`

Note that to generate the data comparison plots, [MatLab runtime v92](#) must be installed and the executable must be run in [Linux-based](#) environment. Otherwise, the plot will not be generated. Detailed information is in README.md.

Step 3: An interactive window should pop up. A clip of the rendered mesh model with simulation results is displayed. FullMonte energy data is normalized so that the total energy launched is equal to one, and displayed on the left. MMC is displayed on the right.



Step 4: In the /Results, /Results/colin27 subdirectories, the following files should be saved:
(1) A 2D scatter plot (if MatLab runtime is installed). File name: comparison_plot_0.png



The horizontal axis represents the energy absorbed at each tetrahedron using MMC and the vertical axis is that of FullMonte. The units are in J/mm².

- (2) A visualization figure in .png format. It should look similar to the window in Step 3. File name: FullMonte_MMC_point.png.
- (3) A data summary text file listing results from both simulators in order of tetrahedron id. The order in the text file is: tetraID, MMC, FullMonte.
File name: point_energy_data_summary_0.txt.
- (4) A text file with statistical test results, including chi-squared statistic, p-value, absolute error between the two simulators using all data points, partial energy used for statistical analysis, and total energy recorded to be absorbed.
File name: chi2test_results.txt
- (5) A .vtk file containing the mesh information, the region data, and the energy data from both simulators. The field names are: adjustedFMenergy, and mmcEnergy.
File name: point_energy_data_summary_0.vtk

Note: To open the plot and the png file from terminal, type: `xdg-open fileName`. This has the same effect as double-click in a file window. The text files can be opened with any text editors of your choice. The .vtk file can be used for visualization as shown in Example II.