PointCloudsVR User Manual

NASA / GSFC

This user manual contains build instructions and run instructions for the PointCloudsVR software. See the 2 relevant sections below. PointCloudsVR is built on top of several open source packages, the 2 main ones being OpenSceneGraph (http://www.openscenegraph.org) and OpenFrames (https://sourceforge.net/projects/openframes/). You should follow the build instructions steps below in order, and *only use the versions of software that are listed*. More recent versions, than those listed below, very likely may not work. These instructions have only been tested with Visual Studio 2017, but might work in Visual Studio 2019 (VS2019 not yet tested).

Build Instructions for OpenFrames (and Lib Dependencies) in Visual Studio 2017

- These instructions detail how to install the dependencies for OpenFrames and associated library dependencies. They assume that you have CMake installed (https://cmake.org) and know how to use the CMake GUI.
- First install Visual Studio 2017
    - Make sure to select C++ components from installer
    - For all lib's below, use the CMake GUI generator "Visual Studio 15 2017 Win64". It shows up the first time you click "Configure" when first running the CMake GUI on a new project.
    - For all lib's below, I created a "build" directory in the lib "root dir" and an "install" directory also in that same "root dir" for that lib.
    - For all lib's below, if they have a "3rd party dependencies" or "third party" CMake variable, set it point to the main OSGeo4W directory --> C:\OSGeo4W64. Note: you *cannot* install OSGeo4W64 somewhere else! It must be in C:\OSGeo4W64. Some other lib's assume that location.
- Next install OSGeo4W, which contains geospatial apps needed for point cloud rendering
    - Get the 64-bit OSGeo4W installer from http://trac.osgeo.org/osgeo4w
    - Use Advanced Install, and ensure that the OSGeo4W Root Directory is C:\OSGeo4W64
    - Select the following packages: gdal, gdal-dev, pdal; accept additional dependencies as indicated by OSGeo4W
        - NOTE: GDAL-dev is required by PDAL, but is not auto-added as a dependency in OSGeo4W, so must be selected manually. If this is fixed in the future, then explicitly selecting gdal-dev will be unnecessary.

- Install freetype lib, by copying it out of this .zip: https://download.osgvisual.org/3rdParty_VS2017_v141_x64_V11_small.7z. That means copy: "include\ft2build.h" into [OSGeo4W]\include, as well as the entire "include\freetype" directory. Also copy "lib\freetype271.lib" into [OSGeo4W]\lib.
- Set GDAL_DATA environment variable to path to [OSGeo4W]\share\gdal directory
- Add to PATH environment variable: [OSGeo4W]\bin
- Make sure that the [OSGeo4W]\bin path is the *first* item in your System Path

- Liblas: Using latest master from GitHub

  - install from GitHub (https://github.com/libLAS/libLAS). You can just download the .zip file, if desired.

  - Liblas requires Boost! Install the Boost binary "current release" for Windows from https://sourceforge.net/projects/boost/files/boost-binaries/ in "C:\local\boost_1_66_0". I used "boost_1_64_0", but probably the latest release as of this writing "boost_1_66_0" will work, but there are more steps than using boost_1_65_1, the last known version that works easily, so just use that version!. The actual binary Windows installer filename would be boost_1_66_0-msvc-14.1-64.exe (direct link: https://sourceforge.net/projects/boost/files/boost-binaries/1.66.0/boost_1_66_0-msvc-14.1-64.exe/download)

  - CMake will find most things it needs in Boost and OSGeo4W, but you will need to point the CMake JPEG_LIBRARY variable in the CMake GUI to [OSGeo4W]\lib\jpeg_i.lib

  - Set Boost_INCLUDE_DIR to, for example, C:/local/boost_1_65_1, if CMake cannot find it in a default location. Additionally, set Boost_LIBRARY_DIRS if it cannot find your libraries.

  - Generate VS project -> Switch to Release build, run ALL_BUILD project followed by INSTALL project

  - Note: if you see an error during building like: "Error MSB3073 The command "setlocal "C:\Program Files\CMake\bin\cmake.exe" -E copy liblas-osgeo4w-start.bat.tmpl osgeo4w/bin/liblas.bat.tmpl" don't worry about it as long as the liblas.dll and liblas_c.dll are successfully built. You are not using any OSGeo4W Liblas stuff -- you are building Liblas outside of OSGeo4W, so ignore that error.

  - After building and installing via CMake GUI, copy liblas.dll and liblas_c.dll into [OSGeo4W]\bin

  - Note, if using Boost 66 or later, the "Build Install" step in Visual Studio 2017 may fail, but as long as the .dll's above are successfully built, you are fine. Just copy over the .dll's and proceed.

    - OSG will call the liblas.dll from the .las osgPlugin at runtime.
    - Separate notes for Boost_1_66_0 and beyond (at least as of 2/13/18): You would need to separately set all of the following Boost libraries in CMake: program_options, thread, system, iostreams, filesystem, date_time. The directory path for each would look like, e.g., C:/local/boost_1_66_0/lib64-msvc-14.1/libboost_thread-vc141-mt-x64-1_66.lib. But instead, make your life easy and use boost_1_65_1! If you use 66 or later, you will still see a final error in Visual Studio 2017 that 1 target failed to be built, but it doesn't matter. As long as liblas.dll and liblas_c.dll are built, and the liblas_info.exe runs ok, you are good to go.
- OpenSceneGraph: Using tag 3.5.6 from GitHub
    - Use this Git clone command to get 3.5.6:
    - `git clone --branch OpenSceneGraph-3.5.6 https://github.com/openscenegraph/OpenSceneGraph.git`
    - Use the main OpenSceneGraph "root dir" as the source code directory in the CMake GUI, e.g.:
    - `C:\Users\Public\Documents\AR-VR\PointCloudsVR_all\OSG-3.5.6` That dir. contains the "src" directory itself.
    - Enable WIN32_USE_MP (need to check the "Advanced" box in CMake GUI to find this!)
    - Set CMAKE_INSTALL_PREFIX as needed (I created an "install" directory in my OSG root directory)
    - It's good to build the OSG examples, tho' they are optional, they can be useful.
    - Make sure Curl, Freetype, GDAL, JPEG, PNG, TIFF, and ZLIB all point to OSGeo4W
    - CMake GUI will find the JPEG include dir., but you must point the CMake JPEG_LIBRARY variable in the CMake GUI to [OSGeo4W]\lib\jpeg_i.lib
    - For Liblas, point the CMake LIBLAS_LIBRARY variable to [liblas install dir]\lib\liblas.lib (e.g. D:/VR/OpenFrames/libLAS/installDir/lib/liblas.lib). And point LIBLAS_INCLUDE to [liblas install dir]\include, and set LIBLASC_LIBRARY to [liblas install dir]\lib\liblas_c.lib (e.g. D:/VR/OpenFrames/libLAS/installDir/lib/liblas_c.lib)
    - Now OSG will know to build the .las osgPlugin, which we need!
    - After checking Advanced box in CMake GUI, and after hitting "Configure" you should see that Boost_INCLUDE_DIR should already be set to, for example, C:/local/boost_1_64_0. It's *ok* if Boost_DIR has a value of Boost_DIR-NOTFOUND.
    - Generate VS project -> In VS 2017, switch to "Release build" in drop-down menu, run ALL_BUILD project followed by INSTALL project
    - If you encounter a linker error (i.e. can't find this Boost lib: libboost_thread-vc141-mt-1_64.lib) while building the las OSG plugin, you may need to add this directory (e.g. C:\local\boost_1_64_0\lib64-msvc-14.1) for linking by right clicking on

    "plugins las" in the Solution Explorer and then, add it to Properties: Configuration Properties->Linker->General->Additional Library Directories
  - Add to PATH environment variable: [OSG-install]\bin
  - Test at command prompt: osgversion (should be 3.5.6)

- osgEarth: Use the 2.9 version:

  - `git clone https://github.com/gwaldron/osgearth.git`
  - `git reset --hard osgearth-2.9`
  - Enable WIN32_USE_MP
  - Set CMAKE_INSTALL_PREFIX as needed (I created an install directory in the osgEarth root directory)
  - Disable BUILD_OSGEARTH_EXAMPLES if you don't want the examples (it's nice to have them)
  - Ignore "Could NOT find Protobuf", "Could NOT find ROCKSDB" CMake Configure messages
  - Generate VS project -> Switch to Release build, run ALL_BUILD project followed by INSTALL project
    - NOTE: WHAT ABOUT GEOS? The GEOS version with OSGeo4W is missing most include files
  - Add to PATH environment variable: [osgEarth-install]\bin
  - Test at command prompt : osgearth_version (should be "osgEarth Library 2.9.0")
  - Test at command prompt from [osgEarth/tests] directory: osgearth_viewer readymap.earth
    - If there's an error related to a "missing ordinal", then ensure that the [OSGeo4W]\bin path is the *first* item in your System Path.

- NetCDF: for reading netcdf files

  - Download and install the latest version of "Pre-Built netCDF-C libraries for Visual Studio" from here: https://www.unidata.ucar.edu/software/netcdf/docs/winbin.html
    - The main folder for this installation comes with 'bin', 'include', and 'lib' directories, which we will refer to next.
  - In Visual Studio, project settings:
    - VC++ Directories->Libraries directories: add the path to netCDF's 'lib' folder.
    - VC++ Directories->Include directories: add the path to netCDF's 'include' folder.
    - VC++ Directories->Executable directories: add the path to netCDF's 'bin' folder.
    - Linker -> Input-> Additional Dependencies: add 'netcdf'lib'(and potentially other .lib files found in the lib directory, if needed)
    - In your code put: '#include <netcdf.h >'

    - To open and visualize netcdf files you can use Panoply available for installation from here:https://www.giss.nasa.gov/tools/panoply/download/

- OpenVR: Using latest master from GitHub
    - Note: no need to build anything! The binaries are already in the repo.! Just download it.
    - install from GitHub (https://github.com/ValveSoftware/openvr). You can just download the .zip file, if desired.
    - Add [OpenVR-install]\bin\win64 to your System Path

- Qt
    - Install Qt 5.10.1 (probably has to be done by an Admin) from: https://www.qt.io/download(pick Open Source version)
    - Add QT_DIR env. variable to Windows -- e.g. D:\Qt\5.10.1\msvc2017_64

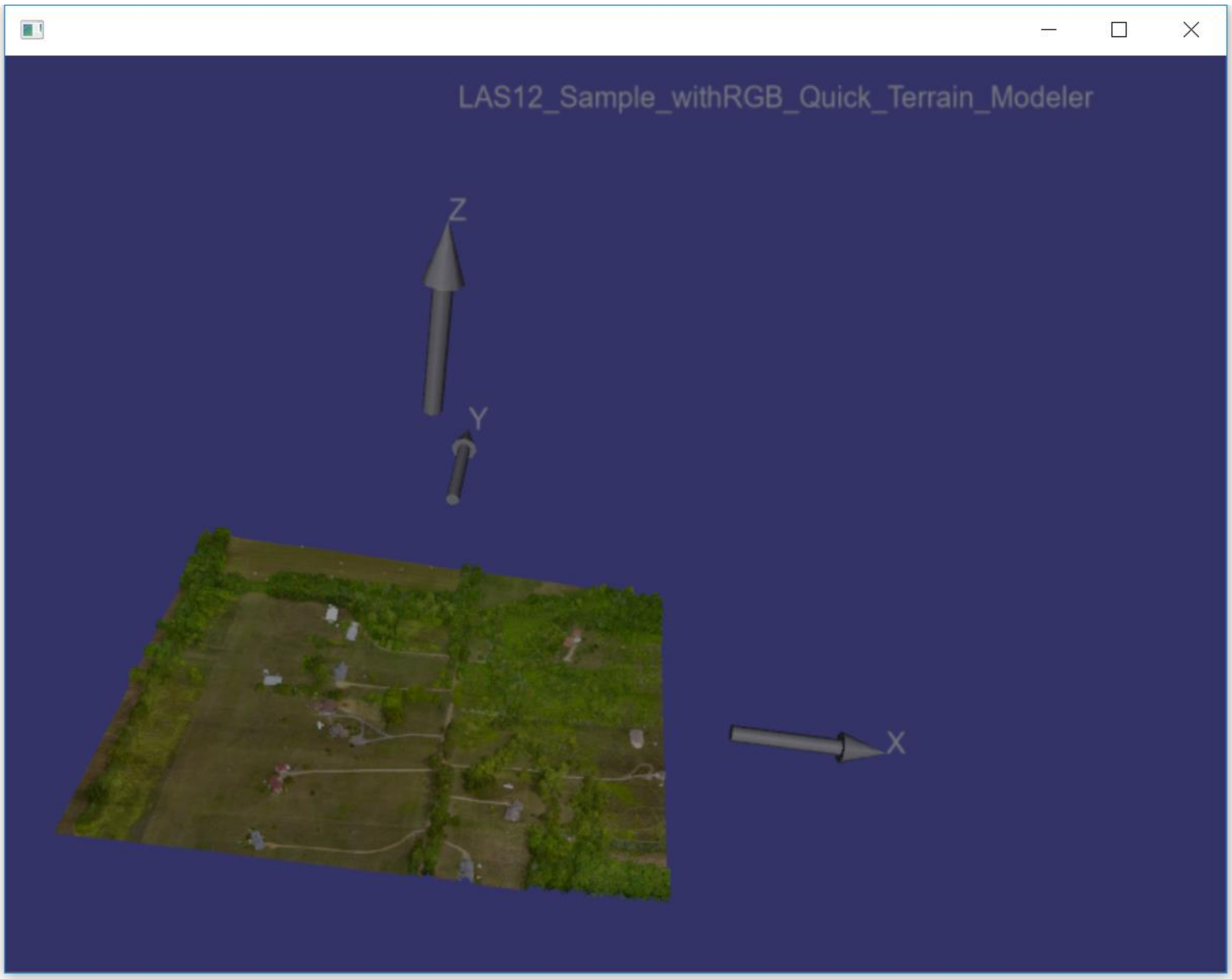
- OpenFrames:
    - Follow OpenFrames build instructions at Open Frames Wiki for Windows: https://sourceforge.net/p/openframes/wiki/Building%20OpenSceneGraph%20in%20Windows/
    - Set OF_QT_SUPPORT CMake variable to enable Qt support in OpenFrames
    - Set OF_VR_TYPE to "OpenVR Stub" if you don't have an OpenVR-compatible HMD, otherwise select "OpenVR" if you have a Vive or Rift or other HMD. If you do select OpenVR, after you hit "Configure" in the CMake GUI, you will need to set the location of OPENVR_SDK_ROOT_DIR

- PointCloudsVR (initial main application: vlpointcloud.cpp):
    - Clone the PointCloudsVR repo.
    - vlpointcloud.cpp is the initial single point cloud app which uses liblas. It's based on ofviewer.
    - Create a "build" directory and an "install" directory in the project root dir. (e.g. D:...\PointCloudsVR\build, and D:...\PointCloudsVR\install)
    - The VS 2017 .sln file will be in PointCloudsVR\build.
    - You need to set 2 environment variables:
        - OPENFRAMES_DIR set to OF install location, e.g., D:\VR\OpenFrames\openframes-code\install
        - OPENVR_SDK_ROOT_DIR set to OpenVR "root" dir., e.g., D:\VR\OpenVR
    - Then run CMake GUI in the usual way, i.e. use the CMake GUI generator "Visual Studio 15 2017 Win64". It shows up the first time you click "Configure" when first running the CMake GUI on a new project. Set CMAKE_INSTALL_PREFIX as needed -- e.g. to D:...\PointCloudsVR\install
    - Generate VS project -> Switch to Release build

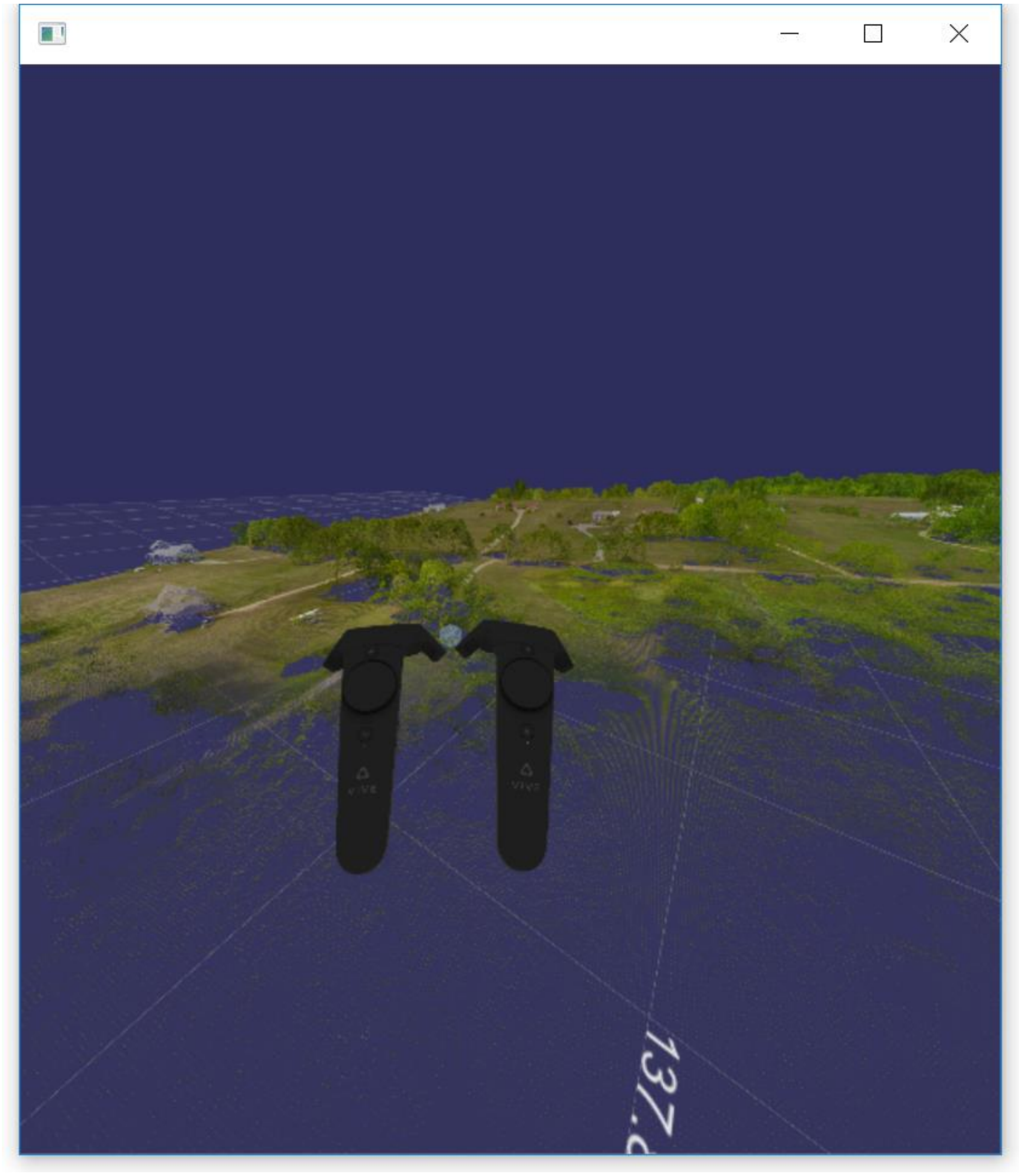
- For the “vlpointcloud” target in VS 2017 Solution Explorer:
    - right-click vlpointcloud —> Properties —> C/C++ —> General —> Additional Include Directories —> add the dir. for [OpenFrames Install Dir]\include
    - Also, Properties —> Linker —> Additional Library Directories —> add the dir. for [OpenFrames Install Dir]\lib
    - You probably need to perform these same 2 steps for OpenVR as well using, e.g., for the Include dir.:
        - `C:\Users\Public\Documents\AR-VR\PointCloudsVR_all\OpenVR\headers`
        - And for the Lib dir.:
        - `C:\Users\Public\Documents\AR-VR\PointCloudsVR_all\OpenVR\lib\win64`
    - Note: you might have to do the above 2 steps for OSG's install/include and install/lib, if OSG files aren't found
- run ALL_BUILD project followed by INSTALL project
- The executable will end up in CMAKE_INSTALL_PREFIX/bin (e.g. D:...\PointCloudsVR\install\bin)
- example run (non-VR run shown below, but not currently supported ;) -- instead, launch SteamVR, and add "--vr" to the command line to give the 2nd screenshot below):

```
D:\VR\OpenFrames\PointCloudsVR\install\bin>vlpointcloud.exe
..\..\data\las\LAS12_Sample_withRGB_Quick_Terrain_Modeler.las
ViewerBase::configureAffinity() numProcessors=8
databasePagers = 1
_forceVertexArrayObject = 0
_forceVertexBufferObject = 0
OpenFrames renderer: GeForce GTX 980/PCIe/SSE2, version: 4.5.0 NVIDIA 369.09 with 4x MSAA
```

You should see:



Running with the --vr option, you should see (after zooming in a bit):



PointCloudsVR.exe Run Instructions (example given for Gaia data)

Note -- The run instructions shown are for displaying Gaia Mission star data, but .las LIDAR point cloud data files can be shown by simply passing the path to an .las file, after "—data", as explained below.

1. You need a Steam account. It's required to run SteamVR, which many VR App's use these days. If you need to create an account, you go here:

   https://store.steampowered.com/join/

2. Start SteamVR (can be found within the Steam App.), but it would be good to create a SteamVR shortcut on your desktop.

3. Open a Windows command console (or "shell"). You can type "cmd" into the Windows 10 search bar and select "Command Prompt".

4. In that Windows command console window you just opened, type (and then hit enter):

```
> cd C:[install_location]\PointCloudsVR\install\bin
```

You will need the data files for this example. The Gaia data files (which include radial velocities) can be obtained from here:

http://cdn.gea.esac.esa.int/Gaia/gdr2/gaia_source_with_rv/csv/

The data path shown in the run example below, i.e. "..\..\..\ParticleData\GaiaDR2" should be to a directory that contains the .csv files from the link above.

Now to run, after *first* pressing the power buttons on the HTC Vive controllers, and verifiying that the Vive HMD, Tracking Boxes, and Controllers are all "Green" in the SteamVR status window, type (and enter):

```
> PointCloudsVR.exe --scene gaia --data ..\..\..\ParticleData\GaiaDR2
```

Notes about the above:

- `--scene scenename` determines what kind of demo is being run (can have value of `gaia`, `mars`, `juniper`, or can be absent)
- `--data` is the directory is where all the Gaia DR2 .csv data files are. There are 8 files (about 7 GB total) holding around 7 million stars with radial velocities.

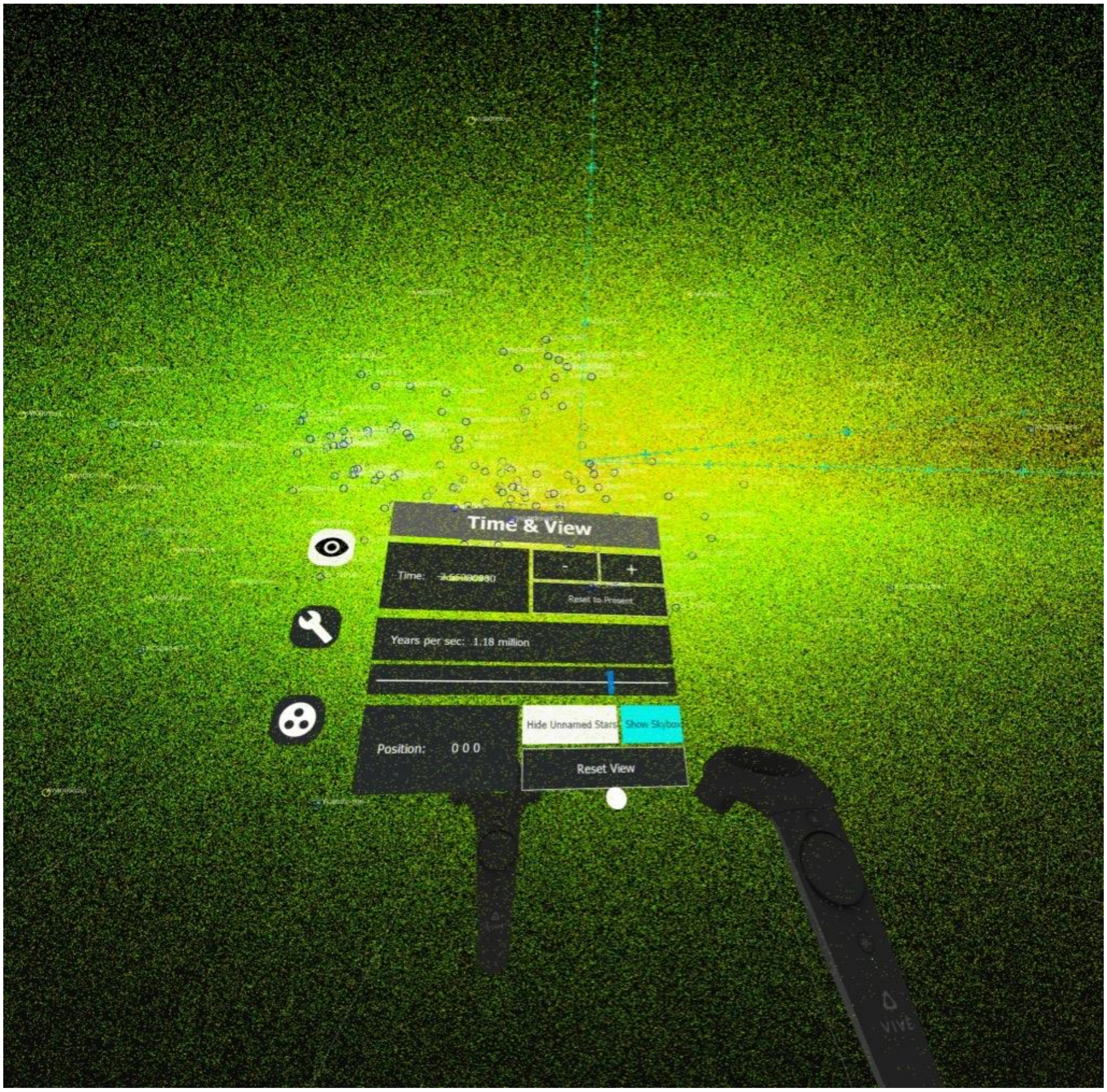
After running, you will see:

```
OpenVR HMD driver name: lighthouse
OpenVR HMD device serial number: LHR-0FA882B1
OpenVR eye texture width = 2138, height = 2376
OpenFrames::OpenVRDevice: Setting up render data for device generic_hmd
OpenFrames::OpenVRDevice: Setting up transform for device generic_hmd0
OpenFrames::OpenVRDevice: Setting up render data for device lh_basestation_vive
OpenFrames::OpenVRDevice: Setting up transform for device lh_basestation_vive1
OpenFrames::OpenVRDevice: Setting up transform for device lh_basestation_vive2
OpenFrames::OpenVRDevice: Setting up render data for device vr_controller_vive_1_5
OpenFrames::OpenVRDevice: Setting up transform for device vr_controller_vive_1_53
OpenFrames::OpenVRDevice: Setting up transform for device vr_controller_vive_1_54
Qt WebEngine seems to be initialized from a plugin.
Please set Qt::AA_ShareOpenGLContexts using QCoreApplication::setAttribute
    before constructing QGuiApplication.
 y = 437 x = 874
 y = 437 x = 874
Number of known Gaia star files = 5
Known Gaia star file name is: ..\..\data\particles\KnownGaiaStars\DR2\Beta_Pictoris.csv
Known Gaia star file name is: ..\..\data\particles\KnownGaiaStars\DR2\Cotten_Song.csv
Known Gaia star file name is: ..\..\data\particles\KnownGaiaStars\DR2\Disk_Detective.csv
Known Gaia star file name is: ..\..\data\particles\KnownGaiaStars\DR2\Sco-Cen.csv
Known Gaia star file name is: ..\..\data\particles\KnownGaiaStars\DR2\Tucana-Horologium.csv
Number of Selection Spheres = 2

.....  [ lots more files and stars ].....

Total Number of Stars (with filtering, if used): 4.09644e+06
```

- The PointCloudsVR Window should now be showing on the Desktop and you can put on the HMD, grab the 2 Controllers, and enter VR 😄.
- An example screenshot, showing stars and some star groups:



- Other command line arguments you can put after "PointCloudsVR.exe" on the command line:

```
> PointCloudsVR.exe --help

General options:
    --help                          Show this text.
    --scene <gaia|mars|juniper>     Choose which scene to run. If option omitted, will
simply display all data files at origin.
    --data  <datafile or directory>  Choose which data to display (can appear multiple
times, but must appear at least once).
```

```
    --novr                           If present, means do not run in vr mode.
    --winRes <num pixels>            Change width, height for square mirror window from 
default 600x600.
                                        Higher resolution means worse performance, but 
better screenshots.

Gaia options:
    --minPc       <parsecs>          Filter out stars closer than --minPc or farther than --
maxPc.
    --maxPc       <parsecs>
    --minParallax <parallax>         Filter out stars with parallax less than --minParallax 
or greater than --maxParallax
    --maxParallax <parallax>
    --minMag      <abs mag>          Filter out stars with absolute magnitude less than --
minMag or greater than --maxMag.
    --maxMag      <abs mag>
    --minTeff     <effective temp>   Filter out stars with effective temperature less than -
-minTeff or greater than --maxTeff.
    --maxTeff     <effective temp>
    --magColor                       If present, color stars by magnitude as opposed to by 
teff(which is the default).
```

- Use the small menu button above the thumb pad to display the menu.
- Most menu items and buttons are relatively self-explanatory.
- Use the trigger button to show the “laser” to select items from the menu. You must “fully click down” on the trigger button to select menu items.
- The thumb pad button, if clicked, will exit "drawing mode" if you are drawing lines or spheres. If that thumb pad button is clicked while the stars are animating forward or backward in time, then the button acts as a "Pause / Play" button, allowing you to Pause / Play at any moment while watching star motion.
- Selection Spheres (full of stars) are stored here:

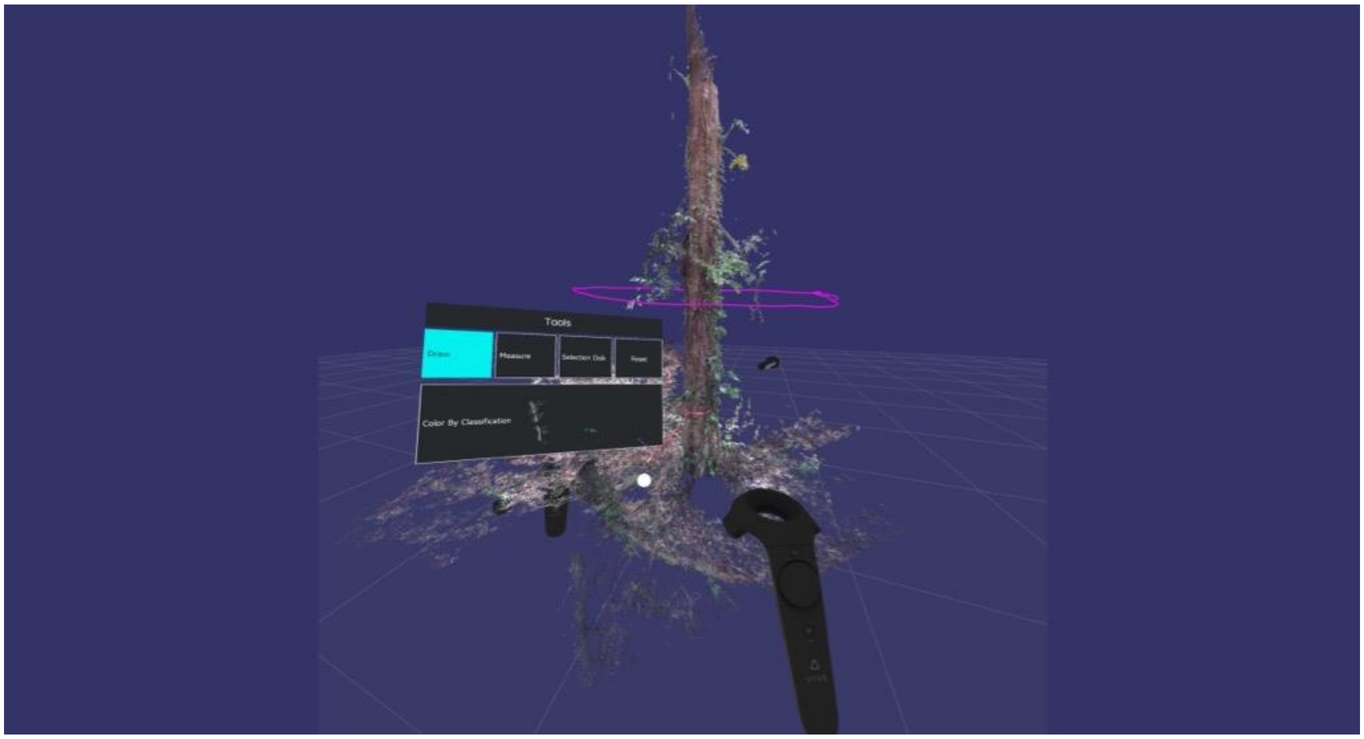
  C:\Users\Public\Documents\AR-VR\PointCloudsVR_all\PointCloudsVR\data\particles\SelectionSpheres\DR2
- An example one, SelectionSphere1.csv, contents look like:

```
    #Origin: 20.1497 -18.4894 1.6623
    #Radius: 6.1937
    SourceID,    X,    Y,    Z,    U,    V,    W
    5833702727878419328,24.2618,-18.533,-1.94515,-0.0393184,-0.012115,-0.013142
    5835620104382030080,21.8225,-15.023,-1.73877,0.00641683,-0.00772623,-0.025994
    5848156426657892480,18.7338,-20.3127,-3.44066,-0.0649497,-0.015452,-0.0119482
    5848544412499643264,16.4936,-17.0837,-2.88798,-0.0398736,-4.9536e-05,-0.0247242
    ...
```

- These selection sphere .csv files, if present in the above directory, are read in at runtime and will be under the "Spheres" menu. They can be shown / hidden via their checkboxes, and will display the number of stars they contain in their labels. They are red, whereas any selection sphere you are drawing will be blue. Both are 50% transparent.

- To take a screenshot, hold the grip button and then press the menu button system button (above the touchpad). The screenshots will be saved into data/images.

An example .las LIDAR file screenshot:


Tools
Draw
Measure
Selection Disk
Reset
Color By Classification