
VSim Installation and Release Notes

Release 10.1.0-r2780

Tech-X Corporation

Mar 12, 2020

CONTENTS

1 Overview	1
2 Installation Details	3
2.1 What is Installed with VSim?	3
2.2 VSim System Requirements	4
2.3 VSim Installation Instructions	5
2.4 VSim Documentation	10
2.5 VSim Release Notes	12
2.6 Software Licensing	30
3 Trademarks and licensing	45

OVERVIEW

This guide shows how to customize VSim [*VSi*], the arbitrary dimensional, electromagnetics and plasma simulation code, to add macros and analyzers.

VSim [*VSi*] is an arbitrary dimensional, electromagnetics and plasma simulation code consisting of two major components:

- VSimComposer, the graphical user interface.
- Vorpil [*NC04*], the VSim Computational Engine.

VSim also includes many more items such as Python, MPI, data analyzers, and a set of input simplifying macros.

INSTALLATION DETAILS

2.1 What is Installed with VSim?

Upon completing the installation process (described in *VSim Installation Instructions*), VSimComposer, the VSim Computation Engine, Python, and MPI will be installed on your computer. These are described in detail below.

2.1.1 VSimComposer

VSimComposer is a graphical user interface for

- Creating and editing VSim input files
- Executing VSim
- Analyzing VSim generated data
- Visualizing VSim generated data
- Viewing the documentation.

The VSimComposer editor and validator have built-in functions and graphical components that help you to create input files. Example input files, ranging in complexity from beginning to advanced, are included with VSimComposer. New VSim users can use these examples as templates. Advanced VSim users can use VSimComposer to validate the syntax of their own input files, whether their files have been created using VSimComposer or by using a text editor.

The VSimComposer Run pane invokes the VSim engine with user definable settings for number of steps, number of data dumps, and restart file, if any. It also allows selection of serial versus parallel VSim.

VSimComposer now allows selection of analysis programs, either supplied with VSim or user written.

The visualization in VSimComposer is provided by the graphical analysis tool **VisIt** (see <https://wci.llnl.gov/codes/visit/>). VisIt is embedded within VSimComposer. Data generated by VSim or by analysis programs automatically appears in the Visualization pane.

All documentation can be seen from within VSimComposer, fully cross-referenced.

2.1.2 VSim Computational Engine

The VSim computational engine runs both as a serial (vorpalsr) and parallel (vorpall) application for multi-processor / multi-core systems that support MPI. VSim now comes in the specialized VSim packages. The VSim computational engine is embedded within VSimComposer.

2.1.3 Python

Python is an open-source, interpreted scripting language managed by the Python Software Foundation. For more information about Python (See <http://www.python.org/>).

VSIm uses Python to process input files, allowing users to set up simulations with math functions, variable substitutions, and macros.

VSIm uses its own embedded version of the Python interpreter to pre-process input files and execute any Python code in an input file.

2.1.4 MPI

The Message Passing Interface (MPI) is an application programming interface (API) for communicating between processes that execute in parallel. There are many implementations. The Linux and Mac versions come with the OpenMPI (See <http://www.open-mpi.org/>) implementation of MPI. The Windows versions come with the Microsoft MPI implementation. The appropriate MPI implementation is embedded within VSImComposer.

More Information

More information about VSIm can be found at the VSIm Product Website (<https://www.txcorp.com/vsim>). Send questions about installing or running VSIm to Tech-X Customer Support at support@txcorp.com. For technical questions about the physics, users may wish to email the VSIm discussion list (<https://ice.txcorp.com/mailman/listinfo/vorpal-users>). Extensive assistance in the use of VSIm or simulation in general is available from Tech-X Professional Services. Please contact Tech-X directly for sales, consulting, and other questions at sales@txcorp.com.

2.2 VSIm System Requirements

VSIm runs on 64-bit Windows, Linux, and Mac and has installation procedures that users will be familiar with on their respective operating system. Some of the systems supported are:

- Windows 7, 8, 10, Server 2008 R2, Server 2012
- Linux distributions with glibc 2.11.3 or later (verified on Fedora/RedHat/CentOS, SUSE, Ubuntu)
- A compatibility version will support Linux distributions with glibc 2.5 or later (verified on CentOS and Ubuntu)
- Mac OS X Yosemite, El Capitan, Sierra, High Sierra, and Mojave
- Cray XC30

Note: The version of glibc can be found with the command: `ldd -version`

The VSIm serial engine (vorpalsr) is available for running on single processor workstations. The VSIm parallel engine (vorpals) is provided for multi-core systems that support the Message Passing Interface (MPI). There is support for running VSIm simulations on Linux clusters with common job schedulers as well as running with the “Windows Clustering” server technology.

VSIm is licensed per compute platform, which may be a workstation or a cluster. For a workstation, creating a license requires the MAC (hardware) address.

A cluster is defined to be “A dynamic collection of compute nodes sharing a common filesystem and a single common queuing system.” For a cluster, all nodes be able to determine their own hardware (or MAC) address and hostname. For a cluster, we also require that there is a method available from each compute node that can determine the list

of compute nodes from the queuing system given to us by the customer. If the customer wants the capability to dynamically change the size of the cluster after licensing, then we require that the compute nodes be able to remote-shell-connect to each other. If a job is running on a collection of nodes, any one of which is licensed, and all of which can see the shared filesystem and are in the nodes list from the queuing system, then we consider the job to be running on the licensed cluster. Otherwise, if one or more job nodes can reach one the licensed nodes and can validate it, and all of the nodes can see the shared filesystem and are in the nodes list from the queuing system, then we also consider the job to be running on the licensed cluster.

VSim is highly scalable, and has been developed to solve the most challenging computational electromagnetics problems of our time. Some example calculations, particularly for plasma acceleration, require a supercomputer to run adequately in 3D, though VSim desktop users may explore how the files work by running them in 2D. Some microwave device examples, such as the Smith Purcell Radiation example, require 12GB RAM or more to run. However, it is possible to set up an input file at lower resolution if you have limited resources.

The VSim 10 Windows installer is around 384MB, and requires around 1.2GB of disk space to install. The Linux installer is 757MB due to the inclusion of additional system packages, and unpacks into around 2.0GB. Please ensure you have enough additional space to run your simulations.

The visualization in VSim uses OpenGL and requires optimal graphics drivers. The standard Linux distributions may not come with drivers written by the graphics-card manufacturer, which are necessary for full hardware acceleration. You should download and install the latest driver for your graphics card from your graphics card vendor's website. In the case of an NVidia graphics card, you can get the latest driver by going to NVidia's website, selecting the Download Drivers link, and then selecting the Linux Display Drivers link.

The application has a minimum size that does not work well below the resolution of 1280x800 pixels or XVGA. Smaller resolutions will start to lose parts of the user interface of the application.

VSim has GPU computing capability based on the NVidia CUDA Toolkit, currently at version 10.1. This is supported only on 64-bit Linux and 64-bit Windows. One does not need a CUDA-enabled graphics card to use VSim unless the GPU capability is invoked, but if a simulation is configured to run on the GPU, then the graphics card must have CUDA capability and a driver that is compatible with version 10.1 of the toolkit.

Note: To use the GPU capability of VSim, you will need a CUDA-enabled GPU card with a driver that is compatible with the CUDA Toolkit version 10.1. Downloading the developer toolkit is not needed, just the driver, but if you do download the CUDA Toolkit (see <https://developer.nvidia.com/cuda-downloads>, then it will guarantee your driver is compatible.

For large parallel simulations running across nodes in high performance cluster, there are some requirements that can worked out easily with system administrators and depend on the details of the cluster configurations. In general, we don't recommend running VSim on AFS file systems.

Note: The Andrew File System (AFS) is not recommended to run VSim in parallel. The distributed AFS system is optimized for location-transparency over a wide area network rather than the low-latency conditions necessary for high performance in cluster storage. File systems such as Lustre or the General Parallel File System (GPFS) perform better when running VSim.

For installation instructions see: [VSim Installation Instructions](#)

2.3 VSim Installation Instructions

These are instructions on installing the VSim product. Please see the [VSim System Requirements](#) first to make sure your system meets the requirements.

2.3.1 Windows VSIm Software Installation

The VSIm distribution package for 64bit Windows is a self-extracting executable installer. Invoke the installer by double clicking on it. The default installation path is:

```
C:\Program Files\Tech-X (Win64)\VSIm-10.X
```

where X is the minor version of your software. To open the VSIm software, go to the Start Menu, click on the Tech-X folder, click on VSIm 10.X, then click on VSImComposer. See Fig. 2.1.

2.3.2 VSIm Composer Properties for Windows 10 High-Resolution Displays

For composer to display correctly on a Windows 10 machine with a high-resolution display, proceed as follows:

1. Go to C:\Program Files\Tech-X\VSIm-10.1.0
2. Right-click on VSImComposer and select Properties.
3. Select the Compatibility tab.
4. Click on “Change high DPI settings”.
5. Under “High DPI scaling override” select the box for “Override high DPI scaling behavior”.
6. Expand the options for “Scaling performed by:” and change the setting to “System (Enhanced)”.
7. Click OK.
8. Back in the VSIm Composer Properties window, click Apply, then click OK.
9. Re-open VSImComposer.

2.3.3 Windows VSIm Software Installation on a Network Share

These instructions outline how to install VSIm to a shared network location. This method will allow users from multiple machines to access and run this installation of VSIm on their local machines.

To start, ensure that you have the shared network location that you want to install to. It is easy to find instructions on how to set up network file sharing on Windows, but the quick steps for Windows 10 are:

1. Open Windows File Explorer.
2. Navigate to the folder you wish to share.
3. Right-click the item and choose Properties.
4. In the Properties dialog click the Sharing tab.
5. Click the Share button.
6. In the Network access dialog, choose the people on your network to share the folder with and click the Share button.
7. In the next frame of the Network access dialog, note the names in the Individual Items list. There will be the name you assigned it in step (6) and the UNC (Unified Naming Convention) Path of the shared directory, which will have the form `\MACHINEpathtosharedirectory`.
8. Click the Done button in the Network access dialog.
9. At this point you can change the UNC Path to a single string by clicking the Advanced Sharing... button and then clicking the Share checkbox and typing a string.

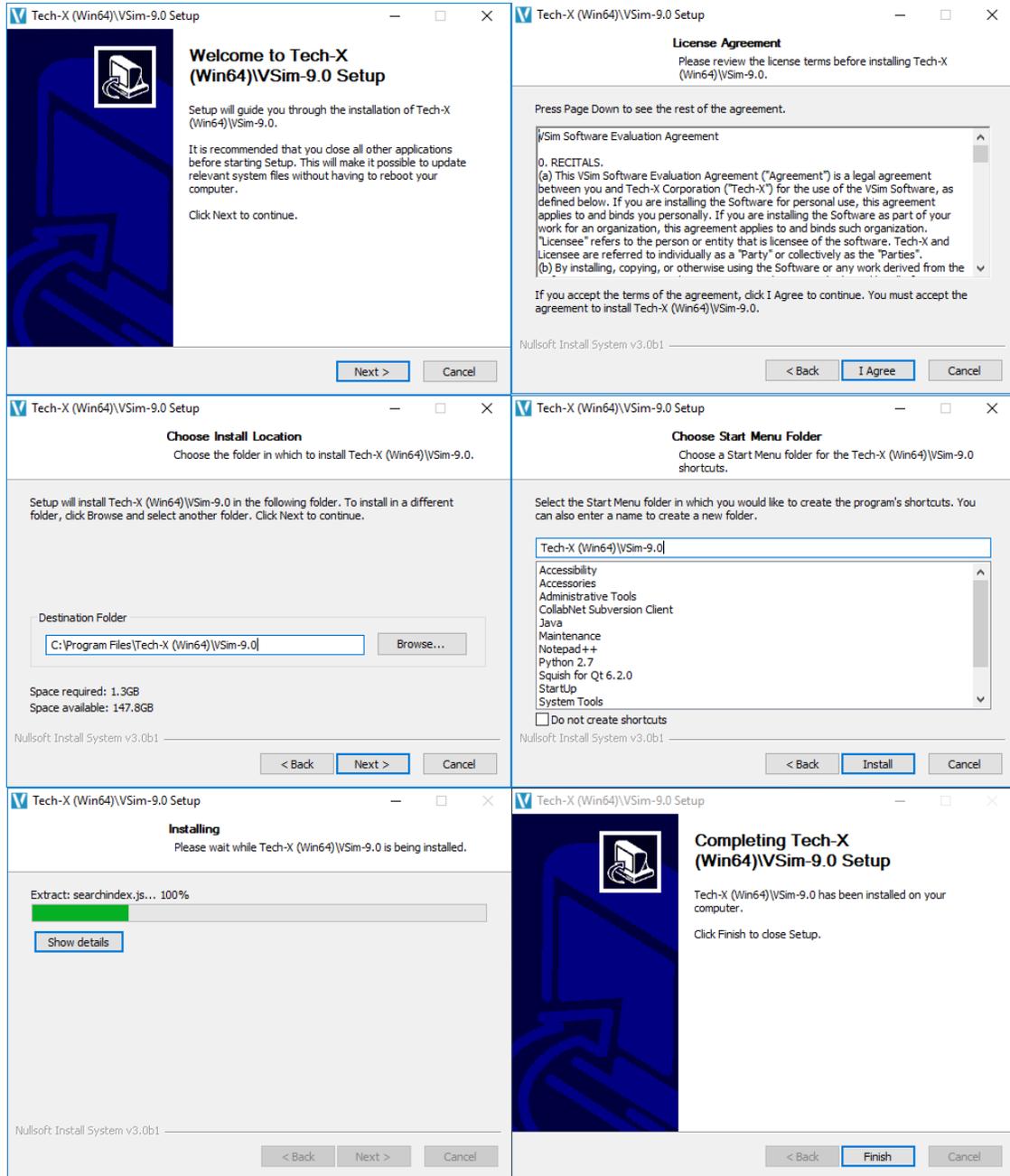


Fig. 2.1: Dialogs for Windows installation

10. Click Close in the Properties dialog.

Now that this folder is accessible, install to the folder by following the Windows Installation instructions above. When you get to the “Choose Install Location” step, type in the shared location as your installation location or click the Browse button and navigate to it. Note that you first go to the Network machine in the left pane (not This PC) and then you navigate down in directories according to the UNC Shared Path. One can install VSIm from the machine where the folder is shared from, or from another machine that has access to the shared folder.

When running VSIm from a network location, one can map the UNC Shared Path to a network drive on any machine that has access to the path. To do this from Windows Explorer on the machine where VSIm will be run, navigate to the second to last folder in the UNC Shared Path, right-click on the icon for the last folder, and select the Map Network Drive menu item. At this time you will be prompted to select a drive letter and can change a couple settings. After making your choices press Finish.

For a concrete example, say we have the follow values:

```
- Installation Computer Name = JANESCOMPUTER
- Shared Folder = C:\Users\jane\Documents\myshare
  (shared on JACKSCOMPUTER)
- Folder UNC Shared Path = \\JANESCOMPUTER\jane (note here Jane
  has chosen the advanced option of a single name "jane" to represent
  the full path of \Users\jane\Documents\myshare)
- VSIm installation Folder = C:\Users\jane\Documents\myshare\VSIm
- Mapped Network Drive on JACKSCOMPUTER = S:\
```

So, for this example, the drive S:\ will show up on JACKSCOMPUTER and is mapped to \\JANESCOMPUTER\jane which is a share name for C:\Users\jane\Documents\myshare on JANESCOMPUTER. So, Jack should see S:\VSIm as the installation location for VSIm and he can conveniently run VSImComposer by double-clicking

```
S:\VSIm\Contents\bin\VSImComposer.exe
```

Finally, to ensure all users have access to the license, the license should be copied to the Contents\engine\bin subdirectory next to the Vorpil engine. So, in our example, this would be on JANESCOMPUTER at the location:

```
C:\Users\jane\Documents\myshare\VSIm\Contents\engine\bin\license.txt
```

inside the original installation of VSIm. This is best done by using Windows Explorer rather than adding the license using the VSImComposer settings.

2.3.4 Windows Cluster VSIm Software Installation

There are a few extra steps when installing on a Windows Cluster and we detail them here. VSIm will work on a Windows cluster that uses Microsoft HPC Pack (Version 2012 R2 or later is required).

First, install VSIm on the cluster headnode using the instructions above in the *Network Share Section*. The cluster license file will need to be copied into the correct location as described above and it will need to refer to the correct shared directory – normally a good choice for this directory would be a subdirectory of the UNC Shared Path alongside the VSIm installation.

For operation on a Windows Cluster it is crucial to use the MPI that comes with Microsoft HPC Pack. So, we must set aside the MPI that is distributed with VSIm. Using the example from the *Network Share Section* above, one would open a Command Prompt and execute the following commands:

```
S:\>cd VSIm\Contents\engine\bin
S:\VSIm\Contents\engine\bin>move mpiexec.exe mpiexecOFF.exe
```

(continues on next page)

(continued from previous page)

```
1 file(s) moved.
S:\VSIm\Contents\engine\bin>move msmpi.dll msmpiOFF.dll
1 file(s) moved.
```

This will ensure that the mpiexec.exe in the PATH variable will be the one on the system. This can be verified by the following:

```
S:\VSIm\Contents\engine\bin>where mpiexec.exe
C:\Program Files\Microsoft MPI\Bin\mpiexec.exe
```

This completes the installation and now it can be tested using the Running Vorpak on a Windows HPC Cluster section of the VSIm User Guide.

Now (also following the example in the *Network Share Section* above), Jack can use VSImComposer to create a simulation directory (say mysim) in the shared folder, for example:

```
S:\jack\simulations\mysim
```

This directory and, therefore, the simulation input will then be available to all the nodes on the cluster when a job is submitted.

2.3.5 Linux VSIm Software Installation

The VSIm distribution package for Linux is a gzipped tarball. Unpack the gzipped tarball into the directory in which you wish to install VSIm. A typical location would be

```
/usr/local/VSIm-10.0
```

The unzip and untar command is

```
$ cd /usr/local
$ tar xf VSIm-10.0.0-Linux64.tar.gz
```

Or, if your Linux machine does not have OpenGL rendering support then you may want to install the “offscreen” version, in which case the file would be “VSIm-10.0.0-Linux64-offscreen.tar.gz. After untarring, the user interface is started with the command:

```
$ cd VSIm-10.0
$ ./VSImComposer.sh
```

If you plan to run the simulation engine or any other executable from the command-line then you will need to source the startup script:

```
$ source /usr/local/VSIm-10.0/VSImComposer.sh
<execute engine, analyzers, etc.>
```

See VSIm User Guide: Running Vorpak from the Command Line for more instructions on command-line operations.

2.3.6 Mac OS X VSIm Software Installation

Unpacking the DMG file:

The VSIm distribution package for Mac OS X is a .dmg installer. Invoke the installer by double clicking on it. Drag the VSIm-10.0 folder into your Applications folder (visible in the installer window). From the Application folder, double click on the VSImComposer icon in the VSIm-10.0 folder. See Fig. 2.2. This default installation path is:

```
/Applications/VSIm-10.0
```

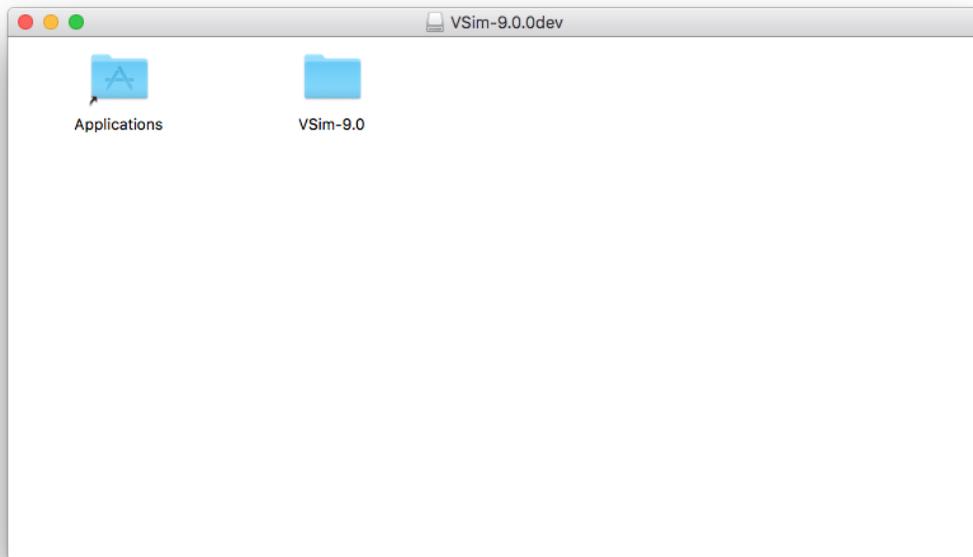


Fig. 2.2: Dialog for the Mac installer

Workaround for signing issues:

As of El Capitan, Apple instituted GateKeeper, which by default prevents one from executing an unsigned application. The symptom can range from inability to install a license to being told the application is damaged as shown in Fig. 2.3.

If you are experiencing this, then close VSImComposer and open an XTerm and run the commands:

```
$ cd /Applications # or wherever one has installed VSIm-10
$ xattr -rd com.apple.quarantine VSIm-10.0
```

Then reopen VSImComposer – the message should be gone.

2.4 VSIm Documentation

In addition to this PDF version of the VSIm documentation, all of the documentation is accessible from within the VSImComposer interface, as well as online at the Tech-X web site, [VSImDocumentation](#).

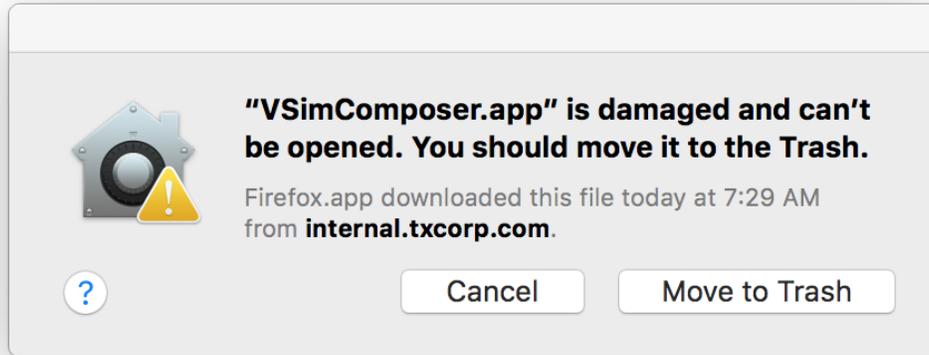


Fig. 2.3: Warning that VSimComposer is damaged, put up by GateKeeper.

2.4.1 VSim Installation

VSim Installation Instructions guides the user through the installation process for VSim. Release notes are also provided in this document.

2.4.2 VSim User Guide

VSim User Guide contains comprehensive VSim documentation, including directions for running VSim from the command line support resources. When you are ready to create your own simulation, consult this document for in-depth information about VSim features.

2.4.3 VSim Examples

VSim Examples provides numerous tutorials for both beginning and advanced VSim users.

2.4.4 VSim Customization

VSim Customization discusses macros and analyzers in further detail, for users who would like a higher degree of customization in their simulations and postprocessing.

2.4.5 VSim Reference

VSim Reference is a quick-reference manual for VSim users to look up specific VSim features and code block syntax for use in editing a VSim input file.

2.4.6 Searching the Documentation

All documentation is available through the VSimComposer itself by clicking on the **Help** icon on the far left or by going to the top menu bar and selecting *Help -> Help Contents*.

The *Search* field in particular is a fast way to find terms and examples, and supports the following options:

- Wildcard search

Use an asterisk * to represent a sequence of characters such that the term you entered can either be the whole or a part of the results displayed. For example:

- Searching for “process*” will display “process” as well as results like “processes”, “processing”, etc.
- Searching for “*process” will display “process” as well as results like “postprocess”, “preprocess”, etc.
- Searching for “*process*” will display “process” as well as results like “preprocessed”, “postprocessing”, etc.

- Omit words You can omit words in your search by putting a minus - sign in front of the word to omit. For example, typing “cylindrical -coordinates” would omit results that contain the word “coordinates”, so your results would only include terms like “cylindrical capacitor”, etc.

All searches are case insensitive.

2.5 VSim Release Notes

The release notes describe new features for the VSim computational engine (Vorpal) and the VSimComposer graphical user interface. Deprecated features and bug fixes are also noted within the release notes.

2.5.1 New and Updated VSim 10.1 Features

VSim Computational Engine (Vorpal)

- Added new Photon Emission via a new Radiation Reaction feature. This allows for relativistic electrons to emit photons that can be optionally tracked in the simulation to record emission energy (weight), position, and relativistic gamma (number of photons). The Radiation Reaction feature is associated with the new particle species: relBorisRad, relBorisRadVW, relBorisRadCyl, and relBorisRadCylVW available with a VSimPA license.
- Fixed issue where alsoAfterRestore was being set to True in the InitialUpdateStep blocks from the Visual Setup. It is now set to False for Electrostatic Phi Field InitUpdateSteps.
- Fixed implementation for simpleSec for both variable weight and constant weight electrons. A single emission probability p is specified. If constant weight, a single constant weight particle is emitted with probability p , or the primary is absorbed with probability $(1-p)$. If variable weight, a single variable weight electron is emitted always, with weight reduced by a factor of p .
- Improved the VpGridBndry error message so that it is clear.
- Fixed issue where tolerance in AztecOO was never set if unspecified in .pre/.in file
- Improved the robustness of triangle absorbers when running in parallel on windows.
- Improved the output of Field Updaters, by increasing the support of the “output” attribute. If you set “output = none” inside a LinearSolver block of kind iterativeSolver, you will not see any output from the solver. You can also use “output = summary”, “output = last”, “output = warnings” and these are passed to the solver. Control of output through the verbosity is still supported as well.
- Added transparentBndry to the VSimMD license package. It was previously only in VSimPD.
- Fixed license issue that prevented speciesRmsDistToAxis and speciesRmsMomen being used.

VSim Examples

- New Antenna Array with Single Excited Element example
- New Phased Array Antenna example
- New Visual Setup version of Electron Beam Driven Plasma example

VSim Graphical User Interface (VSimComposer)

- Added the ability to create Arrays of CAD Geometry. Individual parts of step files or stl files can be duplicated into rectangular arrays by right clicking and select "Create Array". This function works just like Array creation with CSG Objects.
- Added the ability to perform boolean operations between CAD and CSG Geometry, as well as between CAD Objects. This can be used to provide modifications to an existing geometry or to simplify being able to set multiple CAD objects to a single Particle or Electrostatic Boundary Condition. This feature works just like boolean operations with CSG.
- Added support for Photon Emission Reaction. This reaction allows for relativistic electrons to emit a photon which is not tracked in the simulation, however the emission energy (weight) position, relativistic gamma (num photons) are recorded automatically in a history. Note that this reaction is incompatible with physics defined electron emitters (e.g. Child Langmuir), as well as particle loading from a file. The user will be prevented from running simulations with these features. The user may get to an undefined state if using secondary electron emission either into or from these species with other electron species.
- Added support for Initial Beam specification for use with electron beam driven plasmas
- Simple secondary emitters will now take an emission probability parameter, SEY curves were not properly used.
- Field at position histories now have a point representation in the setup window
- Accumulate Particle Boundary conditions can now be used with log histories
- Excitation Reactions can now use 2Column Data for cross section
- Fixed bug that prevented proper use of some more complexly organized .step files
- Allowed for imported CAD files that lead with a number (e.g. 15Pipe.stl)
- Fixed bug that prevented use of Combo Histories in electrostatic simulations
- Allowed use of all SpaceTimeFunctions in Neumann Boundary Conditions
- Fixed bug where variable weight neutral particles would have weight 0.
- Fixed bug where initial update step in electrostatic simulations was updating incorrect field. No impact on simulation results
- Corrected particles specification from the minimum electron temperature to the maximum electron temperature.
- Added safety factor added to automatic calculation of time step in particle simulations.
- Fixed bug in Fowler-Nordheim emission so that field enhancement property is correctly applied
- Improved robustness of importing STL files. Now double-checks whether STL files whose header indicates ASCII are actually ASCII, and clean it up to avoid problems with potential binary files. Fixed issue with distinguishing facet tag from solid tag.
- More user-friendly messages for when failing to import and use tables when importing an h5/vsh5 file.
- Added protection against user's system language local settings interfering with reading of input files
- Improvements to Linux system information script to help diagnose system issues.

- Added the attribute `includeCylVolElem` to `xvLoaderEmitters` in cylindrical coordinates for inputs produced in Visual Setup. Removed the scaling done in the Visual Setup as it is now being done inside by composer (`txgml`).
- Added specification of default value of `numPSBins` in Visual Setup in order to suppress a warning in `vorpal`.
- Converted failing to `findInsideCorner` from warning to notice.
- Fixed bug where Dirichlet shape boundary conditions were not being created if the shape name itself started with a number.
- Fixed bug to ensure `emissionProb` variable is set properly for usage of secondary emitters when constant weight particles are used.
- New MacOSX Catalina security feature (notarization) supported.

2.5.2 New and Updated VSIM 10.0 Features

VSIM Computational Engine (Vorpal)

- Improved robustness and user experience for non-UniCart grids.
- Fixed an issue where fields were not being interpolated correctly onto particles if the field overlap was larger than the underlying grid overlap.
- Corrected `XvLoaderEmitter` so that particles with zero weight (by default) are not loaded. Users can still use the `loadZeroWeightPtcls` flag.
- Improved accuracy and stability of cut-cell meshing algorithm especially for simulations with large number of MPI ranks (more than 10,000).
- Simplified `XvLoaderEmitter` parameters: `useCornerMove = "false"` is now `entranceMove = "default"` (or do not specify `entranceMove`); `useCornerMove = "true"` is now `entranceMove = "corner"`; `useStairStepMove = "false"` is now `entranceMove = "default"` (or do not specify `entranceMove`); and `useStairStepMove = "true"` is now `entranceMove = "stairstep"`
- Corrected issue that prevented usage of scalars (time-dependent, spatially independent values) with linear solvers.
- Improved Histories so that syncing is more sophisticated and efficient.
- Expand the cases where `feedbackMeasured Histories` can be used.
- Added capability for initializing beam self-fields as particles are loaded. This simplifies the new linear algebra infrastructure
- Fixed an issue where particles could pass through a cut-cell boundary without being absorbed if the boundary coincided with a cell face.
- Reducing and clarifying output from simulations.
- Added support for tagged particles in cylindrical coordinates.
- The particle combination algorithm was improved to make the choice of final velocity quadrant more efficient and deterministic.
- Added Dey-Mitra field extrapolation for cut cells.
- Added support for `speciesNumberOf` and `speciesNumberGener` histories to be used with binary combination history.
- Improved the absorption/emission calculations that affect long-running simulations.
- Deprecated `VpPoissonUpdaterInput`.

- Deprecated all EmField-based input as they have fully replaced by setup described by MultiField-based input.

VSim Examples

- New VSimPD “Townsend Discharge” example.
- New VSimEM “Dipole Antenna” example.
- Replaced VSimEM Photonics examples with improved versions: Removed “Cylindrical Dielectric Fiber,” “Dielectric Waveguide Mode Calculation using Point Permittivity” (Visual and Text), “An example of modeling a Microring Resonator,” and “An example of extracting the mode of a Microring Resonator” and added “Multimode Fiber Mode Calculation,” “Multimode Fiber Mode Extraction,” “Multimode Fiber with Mode Launcher,” “Dielectric in Electromagnetics,” “Dielectric in Electrostatics,” and “Ring Resonator” examples.
- New VSimEM “Scattering off MIM Waveguide” example.
- New VSimMD “Vaughan Secondary Electron Emission” text-based example.
- New VSimMD “Laminar Brillouin Flow” text-based example.
- New VSimMD “Smith-Purcell Radiation (SPR)” example.
- New VSimPD “Penning High Intensity Ion Source” example.
- New VSimPD “Ion Source” example.
- New VSimPA “Dielectric Wall Wakefield Acceleration” text-based example.
- The “Laser Ionization” example was moved from VSimPD to VSimPA.
- The MIM Waveguide example was moved to the Photonics section and renamed drudeLorentzMIM.
- Removed text-based examples that have visual-setup equivalent examples.

VSim Macros and Translator

- Particle Log Histories now make use of the kind speciesAbsPtcIData2.
- Current Density field no longer needs to be explicitly added, but is only created if necessary for the simulation.
- Improved verification that electrostatic boundary conditions are not overlapping.
- Guard Cells properly created in “no field solver” and “prescribed fields” simulations.
- Improvements to specular boundary conditions preventing particles from being removed from the simulation.
- Improvements to selective processing.
- Performance improvements to feedback stfuncs.
- Corrected field ionization collisions in electrostatic simulations.
- Corrected emission offset property of particle emitters in cylindrical coordinates.
- Fixed Constant Weight Particles Particle Loader with physical density allowing the use of spatial profiles.
- Corrected calculation of time dependent external fields.
- Improved interpolation of fields for Prescribed Fields solver when multipacting.
- Field Scaling Electrons are now Constant Weight. Previously variable weights were used to index particle creation. This can now be accomplished with the Binning Feature of Visualization.

- Can now use all built in SpaceTimeFunctions when setting velocity of particle emitters. Note that if a built in function is used only global velocity coordinate systems can be used, and if used for the thermal velocity component a gauss function will not automatically be applied.

VSim Analyzers

There are 13 new analyzers in VSim 10.0:

- `annotateSpeciesDataOnPlane.py`: provides particles files corresponding to the data in the history
- `compareFields.py`: compares fields, outputting the difference field
- `computeDielectricModes.py`: computes the profiles and effective indices of guided modes
- `computeFieldCrossProduct.py`: performs a cross product of two different vector fields
- `computeFieldMaxAmplitude.py`: reads field data and writing the max amplitude to a history
- `computeFieldRelIntensityHilbert.py`: uses a Hilbert transform of the E field to obtain the time invariant E amplitude
- `computeGradient.py`: writes the gradient of a history into a new history
- `computePctlBalance.py`: plots the number of particles on each rank for small computations
- `computePctlImpactSpectrum.py`: generates particle number density, and macroparticle spectrum fields
- `hfssToVsh5.py`: converts an HFSS field files to a vsh5 file
- `removePctlComponent.py`: removes additional particle components
- `truncateHistory.py`: removes records from start and end in a scalar history
- `writeCutCellField.py`: puts the cut-and-kept and cut-and-excluded cells from a geometry into a separate field object

VSim Graphical User Interface (VSimComposer)

- Managed Weights “Macroparticles per cell for Combining” property can now be assigned constants and parameters.
- Addition of a Separable Current Distribution. This provides performance improvements but requires the spatial component and temporal component of the current distribution be specified separately. The spatial component may be specified using functions or an external field.
- Addition of External Mode Launching Field for electromagnetic simulations. This will import a 2D field generated by the `computeDielectricModes` analyzer and launch a mode from it with a defined temporal profile. The 2D field may be positioned anywhere in a 3D simulation. Note that in simulations with dielectrics the “D” field should be used in place of the “E” field.
- Addition of a flux conserving particle combination algorithm for managed weight particles.
- Addition of physical offset parameter to particle emitters, this can specify an emission location in physical distance rather than in grid cells.
- Allow particle log histories to be assigned any number of particle attributes.
- Addition of Plasma Dielectric Specification in electromagnetic simulations.
- Addition of Drude-Lorentz and Debye-Lorentz dielectrics in electromagnetic simulations.
- Particle Combination Histories now support any number of histories to combine.
- Specification of Time Variance in Prescribed Fields Simulations independent for Electric and Magnetic Fields

- Interior Partial Transmitter particle boundary conditions now support option to reflect non-transmitted particles
- Can now use .step files with restricted character names (:, -, >, = etc.)
- Can now set Field Scaling Particle Loaders with Grid alignment.
- Can now use all built in SpaceTimeFunctions when setting velocity of particle emitters. This includes external python functions.
- Improved saving run settings in simulation input files.
- Improved efficiency when opening large .stl files.
- Improved controls for visualization plots.
- Improved user experience with rerunning a simulation.
- Fixed issues with the setup tree interface.

2.5.3 New and Updated VSim 9.0 Features

VSim Computational Engine (Vorpal)

A new reaction framework that has more and faster reactions was implemented. The speed of the new reaction framework comes from implementing the no-time-counter algorithm.

Previous reactions implemented with this new capability are

- elastic collisions
- charge exchange collisions electron ionization
- impact ionization
- field ionization
- recombination (e.g., $H^+ + e^- \rightarrow H$)
- 3-body recombination (e.g., $H^+ + e^- + e^- \rightarrow H + e^-$)
- electron impact dissociation (e.g., $H_2 + e^- \rightarrow H + H + e^-$)
- excitation (e.g., $H + H/e^- \rightarrow H^* + H/e^-$)
- dissociation (e.g., $H_2 + H/e^- \rightarrow H + H + H/e^-$)
- electron attachment (e.g., $H + e^- \rightarrow H^-$)
- negative ion detachment (e.g., $H^- + H/e^- \rightarrow H + e^- + H/e^-$)

New reactions are

- dissociative ionization (e.g., $H_2 + e^- \rightarrow H^+ + H + e^-$)
- dissociative recombination (e.g., $H_3^+ + e^- \rightarrow H_2 + H$)
- general inelastic binary reaction with 2 reactants \rightarrow 2 products, momentum conserved, and a specified energy lost. (e.g., $H_2 + H_2^+ \rightarrow H_3^+ + H$)

In addition, field ionization removes the ionization energy from the field.

One can now specify the species in fluid and particle blocks by element name, which sets the mass, charge, ionization energy, and excitation energy. Once a predefined species has been set, it is ready to be used within the rxn framework. In addition, these properties can be defined for a custom species with (species=custom) in the ptcl/fluid block. Further, one can still override the values for any species, e.g., select Hydrogen but set its ionization energy to be different from 13.8 eV.

The secondary emission process was generalized so that the impact of any species on a wall can lead to the emission of itself or any other species. (In the prior implementation, the allowance of secondary emission was determined by the order of appearance in the input file.) The user now has some control over secondary particles, including customizable weighting and tagging, as well as correlated secondary emission of multiple species.

Memory usage was reduced for electromagnetic simulations.

The computation of surface fields at boundaries was improved, making the motion of particles more accurate, even in cells that overlap the boundary.

Slab histories with reduced communication and memory usages were developed, enabling larger simulations on multi-core CPUs.

The methodology for second-order dielectric updaters with conformal boundaries and dispersion was developed and implemented.

Restarts were enabled for simulations with grid boundaries having cuts at domain boundaries.

All random processes can be controlled by a seed.

There are now emitter diagnostics that record individual particle data of emitted particles as well as sums and averages over all particles emitted in each time step.

More robust cut-cell absorption with optional diagnostics recording the absorption location, the surface normal at that location, and the exact time of absorption.

For variable weight particles, collisions now demonstrate the correct collision frequency based on the supplied cross-section (with version 9.0.1)

VSim Analyzers

All analyzers updated to a common interface to reduce the need to look up certain individual properties.

Custom analyzer development was simplified through restructuring so that basic services (file reading, options) flow automatically.

New analyzers:

- S Parameters from History (`computeSParamsFromHists.py`)
- S Parameters from Overlap Integral Calculation (`computeSParamsViaOverlapIntegral.py`)
- Compute Geometry Cavity Merit Factor G (`computeCavityG.py`)
- Extract Modes via Operator (`extractModesViaOperator.py`)
- Compute Accelerating Voltage and Transit Time of a Cavity Mode (`computeTransitTimeFactor.py`)
- Create Field Data on an Unstructured Mesh Representing Surface Geometry (`putFieldOnSurfaceMesh.py`)
- Annotate fieldOnLine history files (`annotateFieldOnLine.py`) (added back in 9.0.2)
- Generates particle number density, and associated fields based on particles stored in history data files (`annotateSpeciesAbsPtclData2.py`) (added back in 9.0.2)
- Sum a history, for example to get a total charge from a current measurement (`computeCumulativeSumHistory.py`) (added back in 9.0.2)
- Computes the emittance as a bunch travels through the simulation (`computeEmittanceFromDump.py`) (new in 9.0.2)
- Computes the emittance as a bunch travels through the simulation on a plane (`computeEmittanceOnPlane.py`) (new in 9.0.2)

- Computes history output containing maximum and minimum particle coordinates for each timestep (computePtcLimits.py) (added back in 9.0.2)
- Computes spectrograms from time-series data in history files (computeSpectrogram.py) (added back in 9.0.2)
- Converts XYZ particle data around the x axis to cylindrical components (convertPtcComponentsCartToCylX.py) (added back in 9.0.2)
- Converts XYZ particle data around the z axis to cylindrical components (convertPtcComponentsCartToCylZ.py) (added back in 9.0.2)
- Creates data files with containing particle paths over time (createParticleTracks.py) (added back in 9.0.2)
- Reads in a single H5 file for a particle species and exports it as a text file or as a sequence of text files (exportSpecies.py) (added back in 9.0.2)
- Performs arithmetic operations on two histories from the same simulation and writing to a third history (performTwoHistoryArithmetic.py) (added back in 9.0.2)

The following analyzers were removed:

- addPtcComponentKEeVx.py
- addPtcComponentKEeVy.py
- addPtcComponentKEeVz.py
- calculateEmittance.py
- calculateFieldMaxAmplitude.py
- computeFarFieldFourierComponent.py
- computeFieldCrossProduct.py
- computeLineIntegral.py
- computeSurfaceFlux.py
- getFieldComponentsOnPlane.py

VSimComposer

The input file is now generated more quickly. Status messages have been expanded to give more detail. The Run Panel was restructured to allow all parameters to be visible at once, and to show the VSim recommendation for certain parameters, such as the time step. The Analyze Panel was restructured to allow several analyzers to be open at once, with separate output windows for each.

Visual Setup generalized to allow easier specification of many simulation parameters:

Basic Settings

- Phase shift boundary conditions
- Easier switching between 2D and 3D simulations
- Decomposition direction specification
- Dump in groups (for use by extract modes by operator)
- Suppress dumps of certain fields

Fields

- Expanded options for Linear Solver
- 2nd order dielectrics, up to 9 dielectrics
- Feedback driven ports
- Import external VSIM and function defined fields

Particles

- Charge accumulation particle boundaries
- Partial transmitter particle boundary
- Diffuse reflector particle boundary
- Monte Carlo interactions
- Particle load From file

Collisions

- Background gas can be a fluid
- Specification of Charged Particles by species name as an option
- Managed weight particles weight setting

Histories

- Accelerating Voltage and Field Slab History

Space Time Functions

- Feedback SpaceTimeFunctions can now use absorbed particle current histories as the feedback history. (with version 9.0.1)

VSIM Documentation

The documentation was rewritten and expanded. There are now five manuals, VSIM Installation, VSIM User Guide, VSIM Examples, VSIM Customization, and VSIM Reference. The User Guide contains an extensive section on the basic concepts of simulation.

VSIM Examples

VSIM examples contains many new examples, in particular concerning photonics and plasma discharges. In detail:

- a6Magnetron1Modes.sdf
- a6Magnetron2Power.sdf
- arrayedWaveguideGrating.sdf

- cylFiber.sdf
- cylindricalWaveguide.sdf
- dielectricWaveguide.sdf
- dielectricWaveguideMode.sdf
- dipoleOnConductingPlane.sdf
- microringResonator.sdf
- microringResonatorMode.sdf
- pillboxCavity.sdf
- singleParticleCircularMotion.sdf
- antennaOnHand.sdf (new in 9.0.2)

One example was removed (advancedDipoleAboveConductingPlane.sdf).

VSIm Distribution and Licensing

Installers and installations reduced in size.

2.5.4 VSIm 9.0 Bug Fixes

VSIm Computational Engine (Vorpal) Bug Fixes

- Collisions between a single species were occurring at a factor of 2 too often, this has been fixed and the reactions have been validated (with version 9.0.1)
- Accuracy of reaction rate improved for nearly mono-energetic collisions (with version 9.0.2)
- Power law and exponential polynomial cross-sections have improved documentation and performance (with version 9.0.2)
- Bug fix in reactions using fluids with moving window (mainly laser ionization) for parallel artifacts (with version 9.0.2)
- Reactions of fluids with or resulting in variable weight particles now have more accurate results and flexibility by introducing new attribute to fluids - numMacroPPCRxns - that determines how many equivalent macro particles per cell are used for the fluids within reactions. For variable weight, this determines the maximum number of macroparticles that could result from the reaction in each time step. (with version 9.0.2)

Visual Setup Bug Fixes

- Corrected potential bug in update period specification for reactions framework (with version 9.0.1)
- Improved partial pre-processing for reactions and monte carlo interactions (with version 9.0.1)
- Corrected improper warning message with MAL boundaries (with version 9.0.1)
- Allowed assignment of non-expression space time functions for relative density particle loaders (with version 9.0.1)
- Automated interpolation for proper particle loading on cylindrical grids (with version 9.0.1)
- Enabled right click assignment of constants and parameters to strings when the default value is a string (with version 9.0.1)

- Removed unused menu item (with version 9.0.1)
- Fixed bug where external help window would block composer window (with version 9.0.1)
- Fixed bug where Analyzer lost focus when set to default (with version 9.0.1)
- Fixed name of starting run log file (with version 9.0.1)
- Fixed bug where removed stfuncs would still be listed (with version 9.0.1)
- Fixed bug where stfuncs could be removed while being referenced (with version 9.0.1)
- Fixed composer freezing if translator could not be found (with version 9.0.1)
- Improved speed of loading SDF files (with version 9.0.2)
- Fixed issues with “Abort Setup” button (with version 9.0.2)
- Prevented crash when changing the name of variable to pre-existing variable name (with version 9.0.2)
- Improved usability of long item lists in the right click context menu of the setup properties (with version 9.0.2)
- Improved usability of save dialog (with version 9.0.2)
- Fixed issue with new setup tree items having the same name as a previous item (with version 9.0.2)
- Fixed bug in certain analyzers on Windows (with version 9.0.2)
- Added more time logging when opening files (with version 9.0.2)
- Made importing STP files faster (with version 9.0.2)
- Fixed bug where unselected shapes were not made invisible (with version 9.0.2)
- Centered default values of lineout intercepts (with version 9.0.2)
- Fixed bug so that window title bar updates when shortDescription is changed in a text-based simulation (with version 9.0.2)
- Ensured user chosen histories are respected when reloading data (with version 9.0.2)
- Fixed bug where variable names inside expressions in setup were not updated when the variable name was changed (with version 9.0.2)
- Allowed user setting of temporary directory (TMPDIR) to address issues when the default directory is too long (with version 9.0.2)
- Fixed bug with disable of “Save and Setup” button when a run is canceled (with version 9.0.2)
- Fixed issue with loading sdf files with arrays of constructive solid geometry in them (with version 9.0.2)
- Fixed bug with pseudo potential and accelerating voltage histories in electrostatics (with version 9.0.2)
- Added verification that dielectric materials are not used in electrostatic simulations (with version 9.0.2)
- Enabled field imports for 2D simulations (with version 9.0.2)
- Bug fix to include cylindrical axis only if necessary (with version 9.0.2)
- Bug fix for partial Neumann boundary conditions had wrong offset in some situations (with version 9.0.2)
- Added of PEC Normal Basic Setting to select the normal of PEC materials. This feature is useful for CAD imports that may have a reversed normal from expectation. (with version 9.0.2)
- Added option to include effects of unmodeled photons in decay reactions (with version 9.0.2)
- Allowed binary combination histories to be used in feedback functions (with version 9.0.2)
- Changed default parameters of electrostatic solvers to more commonly used values (with version 9.0.2)

- Clarified basic setting of how a Perfect Electric Conductor is assigned to a shape (with version 9.0.2)

2.5.5 New and Updated VSim 8.2 Features

The following features are new or have been updated for the 8.2.0 release.

VSim Computational Engine (Vorpai)

- Added a new Monte-Carlo elastic collision algorithm with threshold energy functionality
- Removed assumption in elastic collisions of first incoming particle being light compared with second incoming particle with the result being energy conservation in the collisions of particles of similar masses.
- Fixed memory leaks with some histories on long runs

VSim Graphical User Interface (VSimComposer)

- Removed gstreamer-0.10 requirement on Linux

VSim Macros and Translator

- Capitalization fix in embcs.mac

Other

- Improved general robustness of license management
- Made cluster and node-locked license tokenfile behavior consistent between Vorpai and VSimComposer

2.5.6 Known Issues in VSim 8.2

- Remote VSimComposer does not work * Windows Cluster support was lost with upgrade to MPI on windows needed for certain simulations. * EM Cylindrical not implemented for visual setup * Occasionally, for parallel runs, the stl reader can spuriously set some regions outside of objects to being inside. When this happens, it is manifest in the visualization of the corresponding geometry field in Data Overview. In all cases observed so far, this has been fixed by offsetting the grid a small amount, changing the number of cells, and/or changing the number of parallel processes.

2.5.7 New and Updated VSim 8.1 Features

The following features are new or have been updated for the 8.1 release.

VSim Computational Engine (Vorpai)

- Scattering processes added for antimony (Sb), including electron-electron scattering
- Added transport and emission from antimony (Sb) using the density of states
- Added model to specify percentage of specular and diffusive emissions
- Added Species kind “bandSpeciesES” to the VSimSD license

- Fixed issues with several command-line arguments including -h and -help
- Improved warnings when improper values are given for simulation parameters
- Fixed issue with line dumping without a name
- Fixed issue where all 3 components of deltaAtBreaks were not checked to determine if the grid is uniform

VSIM Examples

These examples now have Visual Setup:

- A Loop Antenna created from a coaxial cable (VSIMEM)
- Horn Antenna (VSIMEM)
- Patch Antenna Far Field (VSIMEM)
- Rectangular Waveguide (VSIMEM)
- Electron Gun (VSIMMD)
- Gyrotron Mode (VSIMMD)
- Helix Traveling Wave Tube 1: Dispersion (VSIMMD)
- Helix Traveling Wave Tube 2: Impedance and Attenuation (VSIMMD)
- Helix Traveling Wave Tube 3: Power Run (VSIMMD)
- 2D Magnetron (VSIMMD)
- Multipacting Resonance in Waveguide (VSIMMD)
- Multistage Collector (VSIMMD)
- 3D Stripline Multipacting (VSIMMD)

Other example features include:

- Updated examples to take advantage of new macro functionality
- Added the numerical Cerenkov filter option to EM examples
- Improved Photonic Crystal examples

VSIM Macros and Translator

- New macro to allow numerical Cerenkov filters in Visual Setup as a parameter of the field solver
- New macro, stfuncs.mac, to handle built-in Space-Time functions including Python file import and feedback
- Added support to set localVelocity parameter off of shape emitters
- Updated the reflecting particle boundary macros to handle rails and corners
- Added history to let user set pseudoPotential by coordinates in addition to via grid index
- Added support for secondary emitters to test particles
- Added support to simplify creation of rectangular and coaxial waveguides
- Added pipe primitive
- Fixed issue with HalfWave Dipole Antenna failing on GPU
- Fixed several issues dealing with handling of cylindrical coordinates

- Improved performance of filters macro
- Improved embcs.mac and VSimEm.mac for a faster implement of MAL boundaries
- Improved particles.mac (support for particle species loader with repeat loading)
- Removed unused macros, unnecessary parameters, and cleaned up referenced inside macros
- Fixed several issues with histories.mac for logging of ptclQty and speciesAbsPtclData
- Fixed issue with parallel restarts on Win10 with CAD geometries
- Fixed issue with specifying name of external magnetic fields
- Better handling of global variables in macros

VSim Analyzers

- Improvements to data analysis executables
- New analyzer, exportSpecies.py, that exports a particle species to a text file or sequence of text files
- New analyzer, getFieldComponentsOnPlane.py, to get field components on a plane
- Updated the computePtclNumDensity.py analyzer to perform spatial averaging
- Improved addPtclComponentKEeV.py, fixing a bug involving mismatched components after multiple runs, adding check on restart, executing dump files in numerical order, better specifying of KEeV, KEeVx, KEeVy, or KEeVz, all in one script

VSim Graphical User Interface (VSimComposer)

- Improvements to tree-based/visual setup
- Fixed issue in visual setup of current sources, changing field specified from E-Field to a proper J-Field
- Better handling of Space-Time functions in Visual Setup tree
- Added support for Field Scaling Electrons
- Improved Constructive Solid Geometry tessellation and meshing for geometry primitives
- Improved speed of Visual Setup rendering
- Added support for geometries described by Python functions
- Improved expression handling in Visual Setup tree
- Fixed issues with certain plots incorrectly displaying in Visualization Tab
- Fixed automatic scrolling issues in Run and Analyze Tabs
- Added support for scaling and translation of STL files
- Added support for all macro changes above in Visual Setup tree
- Fixes issues with resizing of the grid
- Improved usability and robustness of Visual Setup tree interface
- Improved wildcard searching in Documentation/Help
- Improved searching with omitting terms with minus sign in Documentation/Help
- Reduced the verbosity of search results in Documentation/Help

- Added instructions on searching in Documentation/Help
- Improved navigation in sidebar contents in Documentation/Help
- Fixed issue with equation rendering speed in Documentation/Help.
- Improved appearance of overall Documentation/Help formatting
- Improved documentation of Visual Setup

Other

- Over 50 use issues fixed.

2.5.8 Known Issues in VSIm 8.1

- Remote VSImComposer does not work
- Windows Cluster support was lost with upgrade to MPI on windows needed for certain simulations.
- EM Cylindrical not implemented for visual setup
- Occasionally, for parallel runs, the stl reader can spuriously set some regions outside of objects to being inside. When this happens, it is manifest in the visualization of the corresponding geometry field in Data Overview. In all cases observed so far, this has been fixed by offsetting the grid a small amount, changing the number of cells, and/or changing the number of parallel processes.

2.5.9 New and Updated VSIm 8.0 Features

The following features are new or have been updated for the 8.0 release.

VSIm Computational Engine (Vorpai)

- New package: VSIm for Semiconductor Devices for modeling electron transport in diamond or Gallium Arsenide
- Fast dielectric algorithm can take multiple dielectrics assigned to multiple shapes
- Windows Cluster Support with graceful shutdown of job
- Automatic detection of GPU acceleration
- Ability to use cross-section data files from a user definable location
- 17+ new simulation object kinds (including Species, Particle Sources/Sinks, Emission Models, Histories, Field Updaters, Grids, Grid Boundaries, and Domains)
- Improved reliability of fast electrostatic solves on Linux

VSIm Examples

- Charge Carrier Dynamics in Diamond (VSImSD)
- Electron Transport in GaAs (VSImSD)
- MESFET Device (VSImSD)
- Metal Schottky Contact and Transport in Diamond (VSImSD)

- Transport and Emission in Diamond (VSimSD)
- Electron Emission in GaAs (VSimSD)
- Langmuir Probe (VSimPD)
- Turner Plasma Discharge, Case 2 (VSimPD)
- Electron Gun (VSimMD)
- Half-Wave Dipole in Free Space (VSimEM)
- Linear Phased Scanning Array (VSimEM)
- Gaussian Laser Beam and Photonic Crystal Cavity (VSimEM)
- Dipole Source and Photonic Crystal Cavity (VSimEM)
- Scattering off Multiple Objects (VSimEM)
- Drude-Lorentz Metal-Insulator-Metal Waveguide (VSimEM)
- Spherical Lens (VSimEM)
- Halfwave Antenna (VSimBase)
- Cylindrical Capacitor (VSimBase)

VSim Macros

- Macros standardized and simplified
- Ability to use macros from a user definable location.

VSim Analyzers

- VSim now comes with more than 30 data analysis executables.
- Standardization of all analyzers so that user defined input will show up as entry boxes.

VSim Graphical User Interface (VSimComposer)

- Easy, configurable tree-based/visual setup for many problems
- Ability to import materials from a file.
- Easily switch between different simulation types (electrostatic, electromagnetic, with or without particles) with GUI adapting to the selection.
- Import CAD objects from multiple formats (step, stl, ply, vtk) and set materials of those objects
- Multiple Geometry shapes can be visually added to simulation.
- Ability to define parts through constructive solid geometry.
- Ability to define expressions and parameters and use those in defining the simulation.
- Ability to assign materials to parts, regardless of how they were imported or constructed.
- Minimize data reloading during visualization.
- Uniformization of output across all platforms.
- Multiple simulations can be saved to one directory.

Other

- Over 373 use issues fixed.

2.5.10 Known Issues in VSim 8.0

- Remote VSimComposer does not work
- EM Cylindrical not implemented for visual setup
- Occasionally, for parallel runs, the stl reader can spuriously set some regions outside of objects to being inside. When this happens, it is manifest in the visualization of the corresponding geometry field in Data Overview. In all cases observed so far, this has been fixed by offsetting the grid a small amount, changing the number of cells, and/or changing the number of parallel processes.

2.5.11 Deprecated Features for VSim 8.0

VSim Computational Engine (Vorpal)

The following engine features are deprecated as of the VSim 8.0.0 release and may no longer be supported in future releases of VSim:

Attributes Deprecated in Version 8.0

- The kinds *bitRevDensSrc*, *bitRevDensSrcVW*, *gaussDensSrc*, *gridDenSrcVW*, and *planarPtclEmitter* of the *ParticleSource* block have been deprecated. Please use the *xvLoaderEmitter* kind instead.
- *kind* is deprecated for the *Decomp* block. The regular decomposition is the only one available and therefore this attribute is not needed at all. Please simply omit “*kind =*” from the *Decomp* block
- *stCadRgn* kind of *STRgn* block has been deprecated. Please use *gridRgnBndry* kind of *GridBoundary* block instead. The *gridRgnBndry* is a much faster method for importing STL geometries.
- *cell* kind of *Species* block has been deprecated.

Features Deprecated in Version 8.0

- Chinese is no longer supported for VSimComposer

2.5.12 Attributes Deprecated in Version 7.0

- The attributes *GridDenSrc*, *CoordProdGridPosGen*, *CoordProdGridDenSrcVW* of the *ParticleSource* block have been deprecated. Please use *gridPosGen* of the *xvLoaderEmitter* kind instead.
- The attribute *numPhysCells* of the *Grid* block has been deprecated. Please use *numCells* instead.

2.5.13 Deprecated in Version 6.0

Attributes

- The attributes *kind=yeeEmField* and *kind=exp24EmField* of the *EmField* block have been deprecated. Use *MultiField* block instead

- The attributes *kind=block*, *kind=cosSqPulse*, *kind=expPeak*, *kind=gaussianGrad*, *kind=gaussianLapl*, *kind=linearRamp*, *kind=mask*, *kind=muWaveMode*, *kind=periodicSTFunc*, *kind=qFormPolyn*, *kind=radSymFunc*, *kind=sawtoothWave*, *kind=strap* and *kind=waveguide* of the *STFunc* block have been deprecated. Use *kind=expression* and specify desired functional expression instead
- The attributes *kind=smoothWide1D* and *kind=spaceScalarFieldFuncUpdater* of the *FieldUpdater* block have been deprecated. Use *kind=smooth1D* instead
- The *Component* sub block of the *EmField* of *kind=funcEmField* has been deprecated. The components are now set using *STFunc* blocks with the names E0,E1,E2,B1,B2 and B3.

VorpalView

- VorpalView, an IDL-based visualization tool, is a deprecated feature as of VSim 6.0.0.

2.5.14 Deprecated in Version 5.2

- The *STFunc* block with the name *loadProb*, which is a sub-block of the *ParticleSource* block, which is, in turn, a sub-block of the *Species* block, has been renamed to an *STFunc* block with the name *macroDensFunc*.
- The attribute *loadInGuardCellFlag* of the *ParticleSource* block with *kind=ptclEmitter*, which is a sub-block of the *Species* block, has been renamed *loadInGuardCell*.
- The attributes *kind=speciesTrackTagInternals*, *kind=speciesTrackTagTraj*, *kind=speciesTrackTraj*, and *kind=speciesTracVel* have been deprecated. Use *kind=speciesTrackTag* instead. We are now only supporting history tracking if the user uses tagged particles.

2.5.15 Deprecated in Version 5.0

Attributes

- The *components* attribute in the *FieldUpdater* blocks of *kind=gpuSTFuncUpdater*, *kind=STFuncUpdater*, and *kind=unaryFieldOpUpdater* has been renamed *writeComponents*.
- The attributes *kind=nonRelESCell* and *kind=2ndOrderRelBorisCell* of the *Species* block have been deprecated. The same models can be accessed with *kind=cell* and *pusher=nonRelEs* for *kind=nonRelESCell* and *kind=cell*, *pusher=relBoris* and *stencil=spline2ndOrder* for *kind=2ndOrderRelBorisCell*.

Blocks

- *Collision*
- *ImpactCollider*
- *Ionizer*. Use the Monte-Carlo framework (i.e., the *MonteCarloInteractions* blocks) in place of using the above deprecated blocks.

2.5.16 Attributes Deprecated in Version 4.2

- The attribute *function* of the *BoundaryCondition* block and the *EmField* blocks of *kind=yeeStaticElecField*, *kind=funcEmField*, and *kind=yeeStaticElecFieldTrillinos*, has been replaced with an *STFunc* block with the name *function*.
- The *UserFunc* block now requires specific names. Arbitrary names for this block have been deprecated.

2.5.17 Attributes Deprecated in Version 4.0

- The attribute *periodicityDirs* of the *Decomp* block has been renamed *periodicDirs*.
- The attribute *indices* of the *BoundaryCondition* and *InitialCondition* blocks has been replaced with the attribute *components*.
- The attribute *function* of the *STFunc* block has been replaced by the attribute *kind*.
- The attribute *singleEmission* of the *ParticleSource* block, which is a sub-block of the *Species* block, has been deprecated. The option *singleEmission=true* has been replaced by the option *ptclCountType=noCounting*, while the default setting is equivalent to the option *singleEmission=false*.

2.5.18 Attribute Deprecated in Version 2.0

- The attribute *xtendUpdate* of the *FieldUpdater* block has been replaced by the attributes *cellsToUpdateAboveDomain* and *cellsToUpdateBelowDomain*.

2.6 Software Licensing

VSim uses several open source packages and here we provide grateful acknowledgement and a list of these licenses

2.6.1 Trilinos

Trilinos is used by the Vorpil computational engine, and is BSD-licensed generally although some packages are GPL (See <http://trilinos.sandia.gov/license-11.4.html>).

We include the following notice:

```
(Begin notice)

Under the terms of Contract DE-AC04-94AL85000 with Sandia Corporation,
the U.S. Government retains certain rights in this software.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

Redistributions of source code must retain the above copyright notice,
this list of conditions and the following disclaimer.
Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the distribution.
Neither the name of the Corporation nor the names of the contributors
may be used to endorse or promote products derived from this software
without specific prior written permission.
THIS SOFTWARE IS PROVIDED BY SANDIA CORPORATION "AS IS" AND ANY EXPRESS
OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL SANDIA CORPORATION OR THE CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
```

(continues on next page)

(continued from previous page)

CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(End notice)

Within Trilinos, we use Amesos:

(Begin notice)

Under terms of Contract DE-AC04-94AL85000, there is a non-exclusive license for use of this work by or on behalf of the U.S. Government.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

(End notice)

The LGPL license is available from the Free Software Foundation webpage.

Both Vorpal Computational Engine and Composer use HDF5 (<http://www.hdfgroup.org/ftp/HDF5/current/src/unpacked/COPYING>).

The notice for HDF5 is:

(Begin notice)

Copyright Notice and License Terms for
HDF5 (Hierarchical Data Format 5) Software Library and Utilities

HDF5 (Hierarchical Data Format 5) Software Library and Utilities
Copyright 2006-2014 by The HDF Group.

NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities
Copyright 1998-2006 by the Board of Trustees of the University of Illinois.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted for any purpose (including commercial purposes) provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice,

(continues on next page)

(continued from previous page)

this list of conditions, and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the following disclaimer in the documentation and/or materials provided with the distribution.
3. In addition, redistributions of modified forms of the source or binary code must carry prominent notices stating that the original code was changed and the date of the change.
4. All publications or advertising materials mentioning features or use of this software are asked, but not required, to acknowledge that it was developed by The HDF Group and by the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign and credit the contributors.
5. Neither the name of The HDF Group, the name of the University, nor the name of any Contributor may be used to endorse or promote products derived from this software without specific prior written permission from The HDF Group, the University, or the Contributor, respectively.

DISCLAIMER:

THIS SOFTWARE IS PROVIDED BY THE HDF GROUP AND THE CONTRIBUTORS "AS IS" WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event shall The HDF Group or the Contributors be liable for any damages suffered by the users arising out of the use of this software, even if advised of the possibility of such damage.

Contributors: National Center for Supercomputing Applications (NCSA) at the University of Illinois, Fortner Software, Unidata Program Center (netCDF), The Independent JPEG Group (JPEG), Jean-loup Gailly and Mark Adler (gzip), and Digital Equipment Corporation (DEC).

Portions of HDF5 were developed with support from the Lawrence Berkeley National Laboratory (LBNL) and the United States Department of Energy under Prime Contract No. DE-AC02-05CH11231.

Portions of HDF5 were developed with support from the University of California, Lawrence Livermore National Laboratory (UC LLNL). The following statement applies to those portions of the product and must be retained in any redistribution of source code, binaries, documentation, and/or accompanying materials:

This work was partially produced at the University of California, Lawrence Livermore National Laboratory (UC LLNL) under contract no. W-7405-ENG-48 (Contract 48) between the U.S. Department of Energy (DOE) and The Regents of the University of California (University) for the operation of UC LLNL.

DISCLAIMER:

(continues on next page)

(continued from previous page)

This work was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately- owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

(End notice)

Composer uses OpenSSL (<http://www.openssl.org/source/license.html>).

The notice for OpenSSL is:

```

/* =====
* Copyright (c) 1998-2011 The OpenSSL Project. All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
*
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in
* the documentation and/or other materials provided with the
* distribution.
*
* 3. All advertising materials mentioning features or use of this
* software must display the following acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
*
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
* endorse or promote products derived from this software without
* prior written permission. For written permission, please contact
* openssl-core@openssl.org.
*
* 5. Products derived from this software may not be called "OpenSSL"
* nor may "OpenSSL" appear in their names without prior written
* permission of the OpenSSL Project.
*
* 6. Redistributions of any form whatsoever must retain the following
* acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit (http://www.openssl.org/)"
*

```

(continues on next page)

(continued from previous page)

```
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE OpenSSL PROJECT OR
* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
* OF THE POSSIBILITY OF SUCH DAMAGE.
* =====
*
* This product includes cryptographic software written by Eric Young
* (eay@cryptsoft.com).  This product includes software written by Tim
* Hudson (tjh@cryptsoft.com).
*
*/
```

Composer also uses libssh (<https://www.libssh.org/development/>).

The notice for libssh is:

```
libssh Developer's Certificate of Origin. Version 1.0
```

```
By making a contribution to this project, I certify that:
```

- (a) The contribution was created in whole or in part by me and I have the right to submit it under the appropriate version of the GNU General Public License; or
- (b) The contribution is based upon previous work that, to the best of my knowledge, is covered under an appropriate open source license and I have the right under that license to submit that work with modifications, whether created in whole or in part by me, under the GNU General Public License, in the appropriate version; or
- (c) The contribution was provided directly to me by some other person who certified (a) or (b) and I have not modified it.
- (d) I understand and agree that this project and the contribution are public and that a record of the contribution (including all metadata and personal information I submit with it, including my sign-off) is maintained indefinitely and may be redistributed consistent with the libssh Team's policies and the requirements of the GNU GPL where they are relevant.
- (e) I am granting this work to this project under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at the option of the project) any later version.

```
http://www.gnu.org/licenses/lgpl-2.1.html
```

Both Vorpai and Composer also uses Python (<https://docs.python.org/2/license.html>).

The notice for Python is:

```
PSF LICENSE AGREEMENT FOR PYTHON 2.7.7

This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and the Individual or Organization ("Licensee") accessing and otherwise using Python 2.7.7 software in source or binary form and its associated documentation. Subject to the terms and conditions of this License Agreement, PSF hereby grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use Python 2.7.7 alone or in any derivative version, provided, however, that PSF's License Agreement and PSF's notice of copyright, i.e., "Copyright © 2001-2014 Python Software Foundation; All Rights Reserved" are retained in Python 2.7.7 alone or in any derivative version prepared by Licensee. In the event Licensee prepares a derivative work that is based on or incorporates Python 2.7.7 or any part thereof, and wants to make the derivative work available to others as provided herein, then Licensee hereby agrees to include in any such work a brief summary of the changes made to Python 2.7.7. PSF is making Python 2.7.7 available to Licensee on an "AS IS" basis. PSF MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, PSF MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON 2.7.7 WILL NOT INFRINGE ANY THIRD PARTY RIGHTS. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON 2.7.7 FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON 2.7.7, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This License Agreement will automatically terminate upon a material breach of its terms and conditions. Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between PSF and Licensee. This License Agreement does not grant permission to use PSF trademarks or trade name in a trademark sense to endorse or promote products or services of Licensee, or any third party. By copying, installing or otherwise using Python 2.7.7, Licensee agrees to be bound by the terms and conditions of this License Agreement.
```

Composer also uses OpenCascade (<http://www.opencascade.com>) OCE and OpenCascade have the following exception to GNU LGPL version 2.1:

```
Open CASCADE exception (version 1.0) to GNU LGPL version 2.1.

The object code (i.e. not a source) form of a "work that uses the Library" can incorporate material from a header file that is part of the Library. As a special exception to the GNU Lesser General Public License version 2.1, you may distribute such object code incorporating material from header files provided with the Open CASCADE Technology libraries (including code of CDL generic classes) under terms of your choice, provided that you give prominent notice in supporting documentation to this code that it makes use of or is based on facilities provided by the Open CASCADE Technology software.
```

Composer also uses Qt (<http://qt-project.org/doc/qt-5/licensing.html>).

The notice for Qt is:

GNU LESSER GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates
the terms and conditions of version 3 of the GNU General Public
License, supplemented by the additional permissions listed below.

0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser
General Public License, and the "GNU GPL" refers to version 3 of the GNU
General Public License.

"The Library" refers to a covered work governed by this License,
other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided
by the Library, but which is not otherwise based on the Library.
Defining a subclass of a class defined by the Library is deemed a mode
of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an
Application with the Library. The particular version of the Library
with which the Combined Work was made is also called the "Linked
Version".

The "Minimal Corresponding Source" for a Combined Work means the
Corresponding Source for the Combined Work, excluding any source code
for portions of the Combined Work that, considered in isolation, are
based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the
object code and/or source code for the Application, including any data
and utility programs needed for reproducing the Combined Work from the
Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License
without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a
facility refers to a function or data to be supplied by an Application
that uses the facility (other than as an argument passed when the
facility is invoked), then you may convey a copy of the modified
version:

a) under this License, provided that you make a good faith effort to
ensure that, in the event an Application does not supply the
function or data, the facility still operates, and performs
whatever part of its purpose remains meaningful, or

(continues on next page)

(continued from previous page)

b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.

d) Do one of the following:

0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.

1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

(continues on next page)

(continued from previous page)

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

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 that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.

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

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU 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 as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

Composer also uses Qt3D (<http://doc.qt.io/qt-5/qt3d-index.html>).

The notice for Qt 3D is the GNU Lesser General Public License, version 3, as stated above under Qt.

Composer is based in part of the work of the FreeType Team FreeType (<http://www.freetype.org/license.html>).

The notice for FreeType is:

The FreeType Project LICENSE

2006-Jan-27

Copyright 1996-2002, 2006 by
David Turner, Robert Wilhelm, and Werner Lemberg

Introduction
=====

The FreeType Project is distributed in several archive packages; some of them may contain, in addition to the FreeType font engine, various tools and contributions which rely on, or relate to, the FreeType Project

This license applies to all files found in such packages, and which do not fall under their own explicit license. The license affects, thus, the FreeType font engine, the test programs, the documentation, and the makefiles, at the very least.

This license was inspired by the BSD, Artistic, and IJG (Independent JPEG Group) licenses, which all encourage inclusion and use of free software in commercial and freeware products alike. As a consequence, its main points are that:

- o We don't promise that this software works. However, we will be interested in any kind of bug reports. ('as is' distribution)
- o You can use this software for whatever you want, in parts or full form, without having to pay us. ('royalty-free' usage)
- o You may not pretend that you wrote this software. If you use it, or only parts of it, in a program, you must acknowledge somewhere in your documentation that you have used the FreeType code. ('credits')

We specifically permit and encourage the inclusion of this software, with or without modifications, in commercial products. We disclaim all warranties covering The FreeType Project and assume no liability related to The FreeType Project.

Finally, many people asked us for a preferred form for a credit/disclaimer to use in compliance with this license. We thus encourage you to use the following text:

```
"""  
Portions of this software are copyright © <year> The FreeType  
Project (www.freetype.org). All rights reserved.  
"""
```

Please replace <year> with the value from the FreeType version you actually use.

(continues on next page)

(continued from previous page)

Legal Terms

=====

0. Definitions

Throughout this license, the terms `package', `FreeType Project', and `FreeType archive' refer to the set of files originally distributed by the authors (David Turner, Robert Wilhelm, and Werner Lemberg) as the `FreeType Project', be they named as alpha, beta or final release.

'You' refers to the licensee, or person using the project, where `using' is a generic term including compiling the project's source code as well as linking it to form a `program' or `executable'. This program is referred to as `a program using the FreeType engine'.

This license applies to all files distributed in the original FreeType Project, including all source code, binaries and documentation, unless otherwise stated in the file in its original, unmodified form as distributed in the original archive. If you are unsure whether or not a particular file is covered by this license, you must contact us to verify this.

The FreeType Project is copyright (C) 1996-2000 by David Turner, Robert Wilhelm, and Werner Lemberg. All rights reserved except as specified below.

1. No Warranty

THE FREETYPE PROJECT IS PROVIDED `AS IS' WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL ANY OF THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY DAMAGES CAUSED BY THE USE OR THE INABILITY TO USE, OF THE FREETYPE PROJECT.

2. Redistribution

This license grants a worldwide, royalty-free, perpetual and irrevocable right and license to use, execute, perform, compile, display, copy, create derivative works of, distribute and sublicense the FreeType Project (in both source and object code forms) and derivative works thereof for any purpose; and to authorize others to exercise some or all of the rights granted herein, subject to the following conditions:

- o Redistribution of source code must retain this license file (`FTL.TXT') unaltered; any additions, deletions or changes to the original files must be clearly indicated in accompanying documentation. The copyright notices of the unaltered, original files must be preserved in all copies of source files.

(continues on next page)

(continued from previous page)

- o Redistribution in binary form must provide a disclaimer that states that the software is based in part of the work of the FreeType Team, in the distribution documentation. We also encourage you to put an URL to the FreeType web page in your documentation, though this isn't mandatory.

These conditions apply to any software derived from or based on the FreeType Project, not just the unmodified files. If you use our work, you must acknowledge us. However, no fee need be paid to us.

3. Advertising

Neither the FreeType authors and contributors nor you shall use the name of the other for commercial, advertising, or promotional purposes without specific prior written permission.

We suggest, but do not require, that you use one or more of the following phrases to refer to this software in your documentation or advertising materials: `FreeType Project', `FreeType Engine', `FreeType library', or `FreeType Distribution'.

As you have not signed this license, you are not required to accept it. However, as the FreeType Project is copyrighted material, only this license, or another one contracted with the authors, grants you the right to use, distribute, and modify it. Therefore, by using, distributing, or modifying the FreeType Project, you indicate that you understand and accept all the terms of this license.

4. Contacts

There are two mailing lists related to FreeType:

- o freetype@nongnu.org

Discusses general use and applications of FreeType, as well as future and wanted additions to the library and distribution. If you are looking for support, start in this list if you haven't found anything to help you in the documentation.

- o freetype-devel@nongnu.org

Discusses bugs, as well as engine internals, design issues, specific licenses, porting, etc.

Our home page can be found at

<http://www.freetype.org>

Composer also uses Boost (<http://www.boost.org/users/license.html>).

The notice for Boost is:

Boost Software License - Version 1.0 - August 17th, 2003

(continues on next page)

(continued from previous page)

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Composer also uses VisIt (<https://wci.llnl.gov/codes/visit/license.html>).

The notice for VisIt is:

Copyright (c) 2000-2014
Lawrence Livermore National Security, LLC.
Produced at the Lawrence Livermore National Laboratory
LLNL-CODE-442911
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the disclaimer below.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the disclaimer (as noted below) in the documentation and/or other materials provided with the distribution.
- * Neither the name of the LLNS/LLNL nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL LAWRENCE LIVERMORE NATIONAL SECURITY, LLC, THE U.S. DEPARTMENT OF ENERGY OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Additional BSD Notice

1. This notice is required to be provided under our contract with the U.S. Department of Energy (DOE). This work was produced at Lawrence Livermore National

(continues on next page)

(continued from previous page)

Laboratory under Contract No. DE-AC52-07NA27344 with the DOE.

2. Neither the United States Government nor Lawrence Livermore National Security, LLC nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately-owned rights.
3. Also, reference herein to any specific commercial products, process, or services by trade name, trademark, manufacturer or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Composer also uses VTK (<http://www.vtk.org/VTK/project/license.html>).

The notice for VTK is:

VTK is an open-source toolkit licensed under the BSD license.

Copyright (c) 1993-2008 Ken Martin, Will Schroeder, Bill Lorensen
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither name of Ken Martin, Will Schroeder, or Bill Lorensen nor the names of any contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Composer also uses Tango (http://tango.freedesktop.org/#Terms_Of_Use).

The notice for Tango is:

The Tango base icon theme is released to the Public Domain. The palette is public domain. Developers, feel free to ship it along with your application. The icon naming utilities are licensed under the GPL (<http://www.gnu.org/copyleft/gpl.html>).

Though the tango-icon-theme package is released to the Public Domain, we ask that you still please attribute the Tango Desktop Project, for all the hard work we've done. Thanks.

TRADEMARKS AND LICENSING

- Vorpal™ © 1999-2002 University of Colorado. All rights reserved.
- Vorpal™ © 2002-2018 University of Colorado and Tech-X Corporation. All rights reserved.
- VSim™ except for Vorpal™ is © 2012-2018 Tech-X Corporation. All rights reserved.

For VSim™ licensing details please email sales@txcorp.com. All trademarks are the property of their respective owners. Redistribution of any VSim™ input files from the VSim™ installation or the VSim™ document set, including *VSim Installation*, *VSim Examples*, *VSim User Guide*, *VSim Reference*, and *VSim Customization*, is allowed provided that this Copyright statement is also included with the redistribution.