



# Introduction to OSSIM C++ Library

Mr. John Stastny  
SPAWAR Systems Center, Pacific

Mr. Bryan Bagnall  
SPAWAR Systems Center, Pacific

Mr. Lucas Keenan  
SPAWAR Systems Center, Pacific

# Disclaimer

- ▼ Though SSC Pacific makes every effort to perform quality assurance on its training materials, the material in this presentation may inadvertently include technical inaccuracies or other errors. We would be grateful if users notify us of any errors or inaccuracies they may find.
- ▼ The presentation contains references to links and to third-party websites. These are provided for the convenience and interest of users and this implies neither responsibility for, nor approval of, information contained in these websites on the part of the U.S. Government. The USG makes no warranty, either express or implied, as to the accuracy, availability or content of information, text, graphics in the links/third party websites. The USG has not tested any software located at these sites and does not make any representation as to the quality, safety, reliability or suitability of such software, nor does this presentation serve to endorse the use of such sites.

# Overview of Talk

- ▼ Introduction to OSSIM Source Code (getting source code, building source code)
- ▼ OSSIM Source Code Examples:
  - ▼ ossim-image-info – the image handler, geometry, and projection.
  - ▼ ossim-mosaic – mosaic one or more images
  - ▼ ossim-mask-filter for land masking
- ▼ Ossim Image Data to IplImage Example – issues without tiling
- ▼ Ossim TileToIplImage example – how to use OpenCV in an ossimFilter.
- ▼ Conclusions

# Introduction to OSSIM Source Code

- ▼ Over 1 million lines of C++ code
- ▼ Perform remote sensing/image processing functions
- ▼ Handles very large images (> 6GB)
- ▼ Handles wide range of data types (nitf, tif, jpeg, HDF, .xml) as well as .shp files
- ▼ Capable of performing wide range of remote sensing tasks
- ▼ Can be extended to increase functionality (ie writing plugins, writing ossim Filters, etc)

# OSSIM Source Code Overview

- ▼ Break very large images up into tiles (typically 512x512)
- ▼ Just as with ImageLinker, we have image chains in code. We always start with an image handler and often finish with an image writer
- ▼ Using OSSIM filters, we can perform land masking, image mosaicing, edge detection, and many other functions
- ▼ OSSIM source code can easily be extended as we will show during our next lesson – writing OSSIM filters

# OSSIM Open Source Software Links

## Open Source Software Image Map (OSSIM)

Download OSSIM: <http://download.osgeo.org/ossim/ossimplanet-1.7.15-minimal.exe> (download the minimal Windows installer)

OSSIM Project website: <http://www.ossim.org>

OSSIM Wiki: <http://trac.osgeo.org/ossim/>

OSSIM Project tutorials: <http://download.osgeo.org/ossim/tutorials/>

OSSIM API Documentation: <http://trac.osgeo.org/ossim/doxygen/>

OSSIM Mailing list: <https://lists.sourceforge.net/lists/listinfo/ossim-developer>

# OSSIM Mailing List

**OSSIM Mailing list:** <https://lists.sourceforge.net/lists/listinfo/ossim-developer>

**Importance of signing up for mailing list:**

- 1) OSSIM is always under development and as such is always changing**
- 2) Developers monitor the list and answer questions about developing using OSSIM**

# OSSIM Resources

**OSSIM Mailing list:** <https://lists.sourceforge.net/lists/listinfo/ossim-developer>

**Importance of signing up for mailing list:**

- 1) OSSIM is always under development and as such is always changing**
- 2) Developers monitor the list and answer questions about developing using OSSIM**



# OSSIM Resources

## List of the required support libraries for OSSIM

libjpeg v 7 - <http://www.ijg.org/>

libgeotiff 1.2.5 - <http://trac.osgeo.org/geotiff/>

gdal 1.7.0 - <http://trac.osgeo.org/gdal/wiki/DownloadSource>

tiff 3.8.2 - <http://www.libtiff.org/>

OpenCV v 2.1.0 - <http://opencv.willowgarage.com/wiki/>

OpenThreads 2.9.5 - <http://www.openscenegraph.org/projects/osg>

- OpenThreads comes as part of OpenSceneGraph.-

OSSIM source code - [http://www.ossim.org/OSSIM/OSSIM\\_Home.html](http://www.ossim.org/OSSIM/OSSIM_Home.html)

- The components we build from OSSIM are:

- OSSIM

- ossim\_plugins

# OSSIM Resources

- **Windows:**

- [http://download.osgeo.org/ossim/dependencies/windows\\_vcexpress2008/](http://download.osgeo.org/ossim/dependencies/windows_vcexpress2008/)

- **Linux/Fedora**

- `yum install svn cmake gcc-c++ qt-devel opencv-devel libgeotiff-devel libjpeg-devel libtiff-devel OpenSceneGraph-devel gdal gdal-devel zlib-devel openmpi-devel minizip-devel libcurl-devel libcurl expat-devel expat yasm libtool postgis mapserver`

# Getting the OSSIM Source Code

- The OSSIM source code is hosted on an SVN server located at:
  - <http://svn.osgeo.org/ossim/trunk>
- Basics of SVN
  - <http://subversion.apache.org/>
  - Common commands:
    - Checkout code: `svn co <source-location> <destination> [-r]`
    - Update code: `svn update [-r]`
    - Difference code: `svn diff`
- To get OSSIM:
  - `svn co http://svn.osgeo.org/ossim/trunk .`

# Building the OSSIM Source Code

- OSSIM is built using CMAKE
- Basics of CMAKE
  - <http://www.cmake.org>
  - Simple platform and compiler independent configuration files.
  - Generates native makefiles and workspaces that can be used in the compiler environment of your choice. (Microsoft Visual Studio/GNU Makefiles)
  - Out-of-source builds
- CMAKE Configuration Files:
  - Examples located at:
  - `ossim_trunk/ossim_package_support/cmake/build_scripts`

# Building the OSSIM Source Code

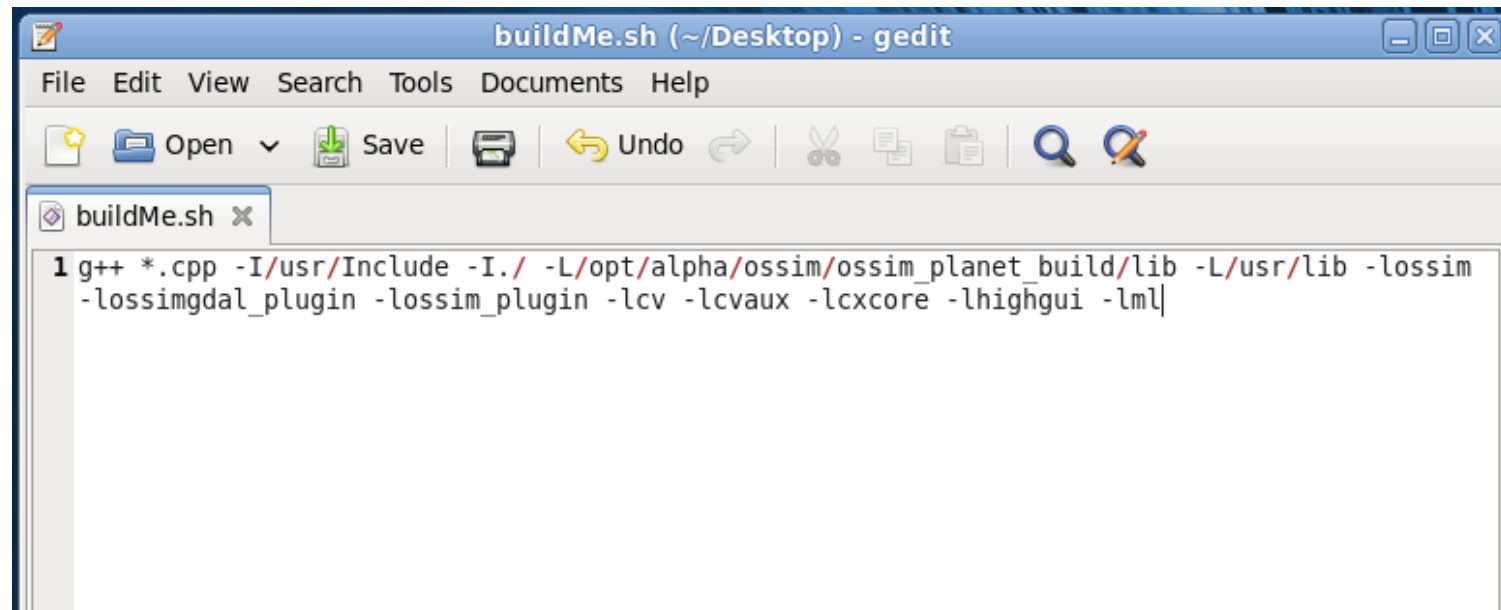
- Let's take a look at the CMAKE file we used
  - `//opt/alpha/ossim/ossim_planet_build/ossim_configure.sh`
- Interesting Lines
  - `-G "Unix Makefiles" \`
  - `-DBUILD_OSSIM=ON \`
  - `-DBUILD_OSSIM_PLUGIN=ON \`
  - `-DBUILD_OSSIMGDAL_PLUGIN=ON \`

# Building the OSSIM Source Code

- Steps for building OSSIM
  - Install OSSIM dependencies
  - Get source code using SVN
  - Create a build directory
  - Copy ossim\_configure.sh into the build directory (and modify)
  - Run ossim\_configure.sh (Produce the makefiles/Visual Studio solutions)
  - From a command prompt cd to your build directory and run make (or open Visual Studio solution)
- After make finishes
  - <build-directory>/lib and <build-directory>/bin

# Adding Additional Include Directories

- ▼ Modify the `buildMe.sh` script that we wrote earlier
- ▼ Add `-I/opt/alpha/ossim/ossim_planet_build/lib`
- ▼ Add `-lossim -lossimgdal_plugin -lossim_plugin -lcvaux -lml`



The screenshot shows a gedit editor window titled "buildMe.sh (~/Desktop) - gedit". The window has a menu bar with "File", "Edit", "View", "Search", "Tools", "Documents", and "Help". Below the menu bar is a toolbar with icons for "Open", "Save", "Print", "Undo", "Redo", "Cut", "Copy", "Paste", "Find", and "Replace". The editor area shows a single line of code:

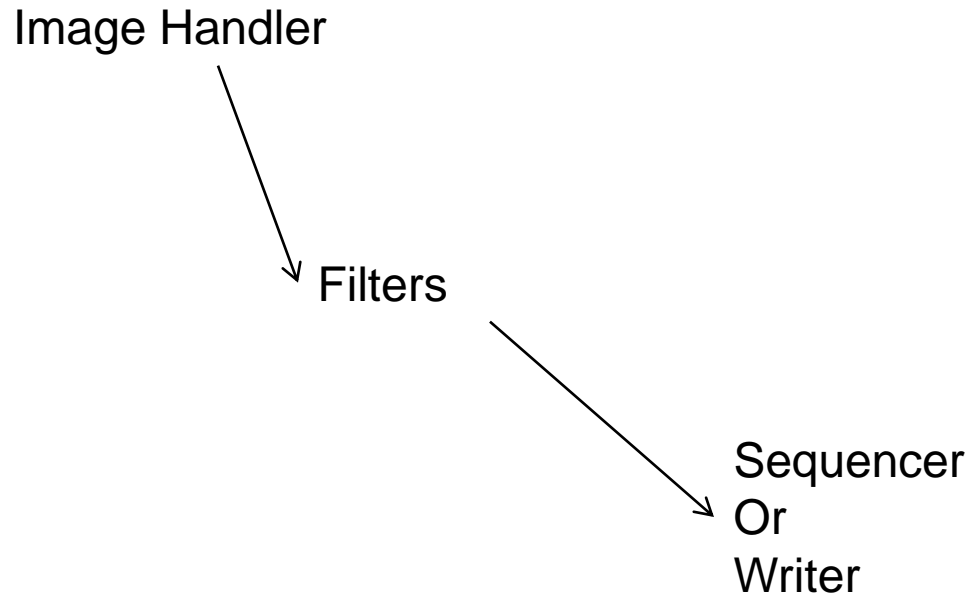
```
1 g++ *.cpp -I/usr/Include -I./ -L/opt/alpha/ossim/ossim_planet_build/lib -L/usr/lib -lossim  
-lossimgdal_plugin -lossim_plugin -lcv -lcvaux -lcxcore -lhighgui -lml
```

# OSSIM Basics

- OSSIM comes with many basic data types to make your life easier
  - `ossimString`
  - `ossimGpt`
  - `ossimDpt`
  - `ossimPolyon`
  - `ossimQuaternion`



# OSSIM Basics



# OSSIM Example- ossim-info

# ossim-info Example

- ▼ Navigate to the source code found at:
  - ▼ `//opt/alpha/ossim/ossim_trunk/ossim/src/apps/ossim-info/ossim-info.cpp`
    - ▼ `ossimInit::instance()->initialize();` --- needed in every ossim application
    - ▼ `ossimRefPtr<ossimInfo>` --- used for garbage collection
    - ▼ `ossimNotify(ossimNotifyLevel_WARN)` --- used for displaying messages of varying severity
- ▼ Navigate to the binary found at:
  - ▼ `//opt/alpha/ossim/ossim_planet_build/bin/ossim-info`
- ▼ Run ossim-info on an image:
  - ▼ `cd /opt/alpha/ossim/ossim_planet_build/bin`
  - ▼ `./ossim-info /opt/alpha/Day\ 2\ Images\San\ Francisco\ Example\TerraColor_SanFrancisco_US_15m.tif`

Other ossim example code and applications can be found in:

`ossim_trunk/ossim/src/examples`

`ossim_trunk/ossim/src/apps`

`ossim_trunk/ossim/src/test`

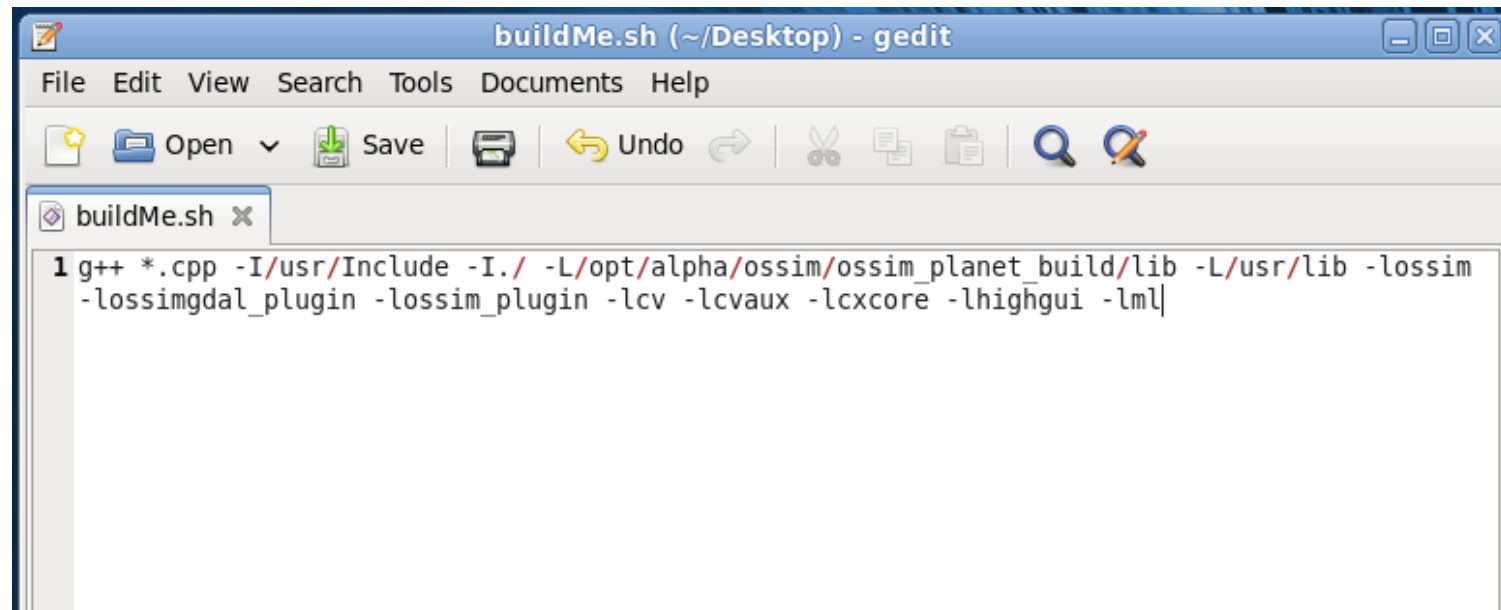
# OSSIM Example- ossim-mosaic

# ossim-mosaic Example

- ▼ Navigate to the source code found at:
  - ▼ `//opt/alpha/ossim/ossim_trunk/ossim/src/apps/ossim-mosaic/ossim-mosaic.cpp`
- ▼ Navigate to the binary found at:
  - ▼ `//opt/alpha/ossim/ossim_planet_build/bin/ossim-mosaic`
- ▼ Run ossim-info on an image:

# Adding Additional Include Directories

- ▼ Modify the `buildMe.sh` script that we wrote earlier
- ▼ Add `-I/opt/alpha/ossim/ossim_planet_build/lib`
- ▼ Add `-lossim -lossimgdal_plugin -lossim_plugin -lcvaux -lml`



The screenshot shows a gedit editor window titled "buildMe.sh (~/Desktop) - gedit". The window has a menu bar with "File", "Edit", "View", "Search", "Tools", "Documents", and "Help". Below the menu bar is a toolbar with icons for "Open", "Save", "Print", "Undo", "Redo", "Cut", "Copy", "Paste", "Find", and "Replace". The editor area shows a single line of code:

```
1 g++ *.cpp -I/usr/Include -I./ -L/opt/alpha/ossim/ossim_planet_build/lib -L/usr/lib -lossim  
-lossimgdal_plugin -lossim_plugin -lcx -lcvaux -lcxcore -lhighgui -lml
```

# OSSIM Example- ossim-mask-filter-test



# ossim-mask-filter-test Example

- Source:
  - `//opt/alpha/ossim/ossim_trunk/ossim/src/test/ossim-mask-filter-test.cpp`
  - `ossimTimer::instance()->setStartTick();`
  - `ossimImageHandlerRegistry::instance()->open(inputImgName);`
- Binary:
  - `//opt/alpha/ossim/ossim_planet_build/bin/ossim-mask-filter-test`
- `./ossim-mask-filter-test 4 0 /mnt/hgfs/My\Documents/chile/TerraColor_SanFrancisco_US_15m.tif /mnt/hgfs/My\Documents/chile/TerraColor_SanFrancisco_US_15m.shp /home/alpha/work.tiff`

# Reading in images, converting from OSSIM data to OpenCV IplImages

# Converting OSSIM ImageSource to an OpenCV IplImage

Next, we look at how to convert from an OSSIM image to an OpenCV IplImage without tiling the image using OSSIM.

The example code we'll discuss is found in  
`/Lesson03_OpenCV_and_OpenCV_with_blobs/examples/OSSIM_To_OpenCV/  
OSSIMToOpenCVImage`

Open the `OSSIMtoOpenCVImage.cpp` file

# Converting OSSIM ImageSource to an OpenCV IplImage

Questions –

- 1) What are some possible limitations to this method?
- 2) Does this solve our problem of “very large images”.

# OSSIM Image Tiling and converting to IplImages

Next, we look at how to tile an image in OSSIM, and to take each tile image and convert it to an OpenCV IplImage.

This is the basic premise behind virtually all filters that you can create.

```
/opt/alpha/OSSIM Lessons &  
Examples/Lesson04_Building_OSSIM_and_OSSIMc++lib/examples/ossim_filter  
_TileToIpl
```

```
./a.out /opt/alpha/Day 2 Images/San Francisco  
Example/TerraColorimage.tiff
```

```
If It can't find libossim.so  
export LD_LIBRARY_PATH=/opt/alpha/ossim/ossim_planet_build/lib
```