OSSIM Users Guide

Version 1.6.5

Open Source Software Image Map – OSSIM, pronounced “Awesome”

Document version 1.1
July 10 , 2006

Written by

Garrett Potts
Mark Lucas

1

2 About OSSIM

This document provides an overview of the Open Source Software Image Map (OSSIM) project http://www.ossim.org . OSSIM is a contrived acronym pronounced “Awesome”. The core of OSSIM is a C++ software library that provides advanced remote sensing, image processing, and geo-spatial functionality. A quick summary of OSSIM functionality includes ortho-rectification, precision terrain correction, rigorous sensor models, very large mosaics, and cross sensor fusions, a wide range of map projections and datums, and a large range of commercial and government data formats.

The architecture of the library supports parallel processing with mpi, a dynamic plugin architecture, and dynamically connectable objects allowing rapid prototyping of custom image processing chains.

Around the core ossim library, the software distribution includes a large number of command line utilities that can be easily scripted for batch production systems and higher level GUI applications – imagelinker, and iview. Additionally, bindings have been generated for other languages.

Also included in the distribution are the osgPlanet and osgPlanet_qt modules. These modules build on top of OSSIM and OpenSceneGraph to provide geo-spatially accurate 3D visualization capabilities.

On 4 Feb 2006, the Open Source Geospatial Foundation http://www.osgeo.org was formed in Chicago IL. The OSSIM project joined the OSGF as one of the founding projects. Future efforts will integrate these projects into formidable open source image processing, mapping,
and GIS systems.

3 Quick Start Tutorial

One of the best ways to understand the power of the OSSIM library is to demonstrate it through the ImageLinker application. ImageLinker was originally developed as a prototyping tool to test and demonstrate the power of the OSSIM library. It has a graphical user interface implemented using the Trolltech Qt library and runs on any platform supported by them – Windows, Linux, Solaris, Mac OSX.

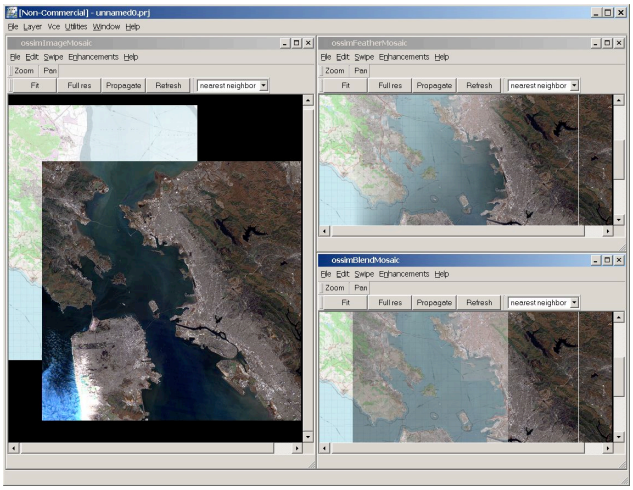


Figure 1 Combiners in the Quickstart Tutorial

This quick walk through will demonstrate just a few of the capabilities of OSSIM and ImageLinker. It is designed so that the user can follow along with step-by-step instructions using the available sample data sets. The tutorial will demonstrate different ways to combine a map and satellite image.

OSSIM and ImageLinker provide several ways to combine geo-spatial data sets. This exercise will demonstrate simple mosaics, blends, and feathers with a map and image of the San Francisco area. The data sets in this example are part of the sample data set that is available on ossim.telascience.org/ossimdata (see section 3.2 for download instructions).

Refer to the Installing section 3 of this document for instructions on how to get and install the software for your platform. For this example we will use the Landsat image and raster map of San Francisco:

Uncompressing the demodata.tgz file on ossim.telascience.org/ossimdata you should find the following:

```
sanfran/sanfran.ccf
sanfran_map/highres_utm_map.tif
```

These files have been prepared as a test set for the quickstart guide to the ImageLinker program. ImageLinker will take advantage of auxiliary files if they are present. There are several OSSIM command utilities to create these files for data setsThe initial files were sanfran.tif and highres_utm_map.tif - both geotiff files. GeoTiffs are tiffs that have geospatial tags embedded in the file.

The respective ovr files are overview files containing reduced resolutions sets. These files can be generated with the img2rr (image to reduced resolution) command line utility. The geom files contain geometry information and were created with the create_geom command.

Histogram files can be created with **create_histo** or can be created when initially ingested into ImageLinker.

The readme.txt files are human readable files about the data sets.

These two files will be used to demonstrate some of the basics of ImageLinker.

So start up ImageLinker and follow the following steps in the rest of this section.

3.1 Open the source files

Open the data files for the map and UTM projected color fused Landsat image of San Francisco by navigating the file browser to your test_data installation and open them. The following command assumes they are located in <install location>/test_data.

> **File->Open**, navigate to test_data/sanfran/san_fran.tif
> **File->Open**, navigate to test_data/sanfran_map/ highres_utm_map.tif

The ImageLinker canvas should reflect that the map and the Landsat image have been loaded.

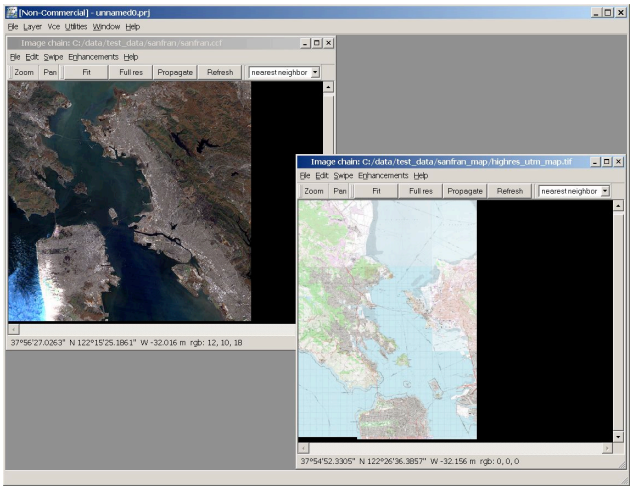


Figure 2 Initial opening of the UTM projected San Francisco Landsat (left) and the San Francisco map (right).

3.2 Managing Windows

In this introductory tutorial we will get a feel for some of the window management and display commands. Each project in ImageLinker lives in its own canvas. Standard maximize, minimize and collapse menu options work on the project canvas within the desktop. Additionally, there are a couple of useful commands for organizing the windows. Lets try a couple.

Select **Window->Cascade**
The open displays are cascaded within the project canvas.

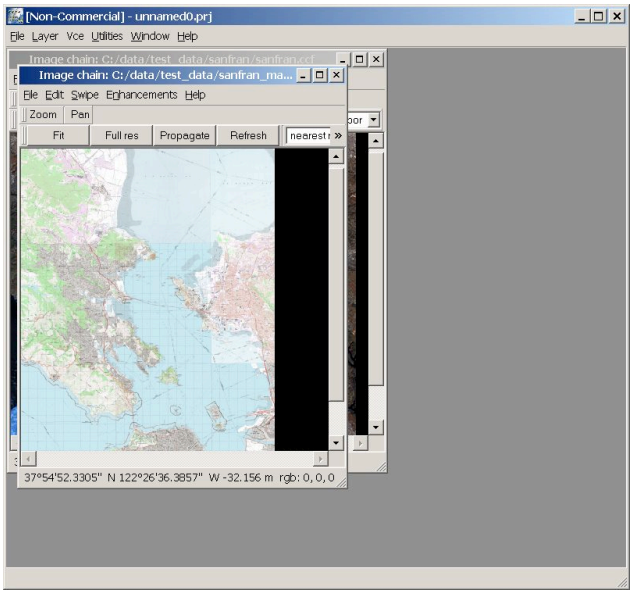


Figure 3 Cascade.

Select **Window->Minimize All**
Collapses the displays into the bottom of the Canvas.

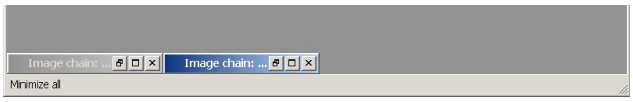


Figure 4 Minimize all.

Select **Window->Restore All**

Will restore the minimized images to their original location.

To place the images side by side use the Tile menu command

Select **Window->Tile**

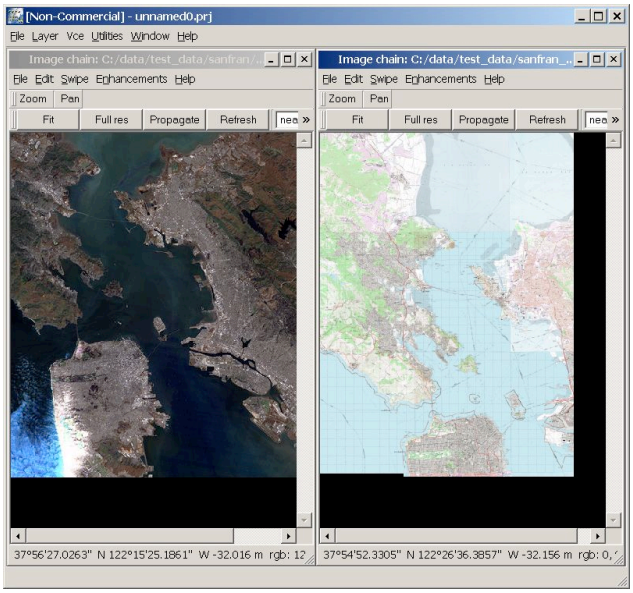


Figure 5 Tiled.

3.3 Pan, Zoom and Propagate in the Displays

The displays have several modes and capabilities that make it easy to navigate and synchronize displays within a project. In the center of the Landsat scene is Treasure Island, place the cursor over the island and click. Note that the Latitude and Longitude of the mouse click is displayed at the bottom of the window. For more thorough position information open the **Edit->Position information** menu option found on the display window.

Click on Treasure Island
Click on the Full res button

You should see a display similar to that shown above. Note the tracking cursor lines in the map display track the geographic location of your cursor. Tracking cursors will appear in all geographically coincident windows.

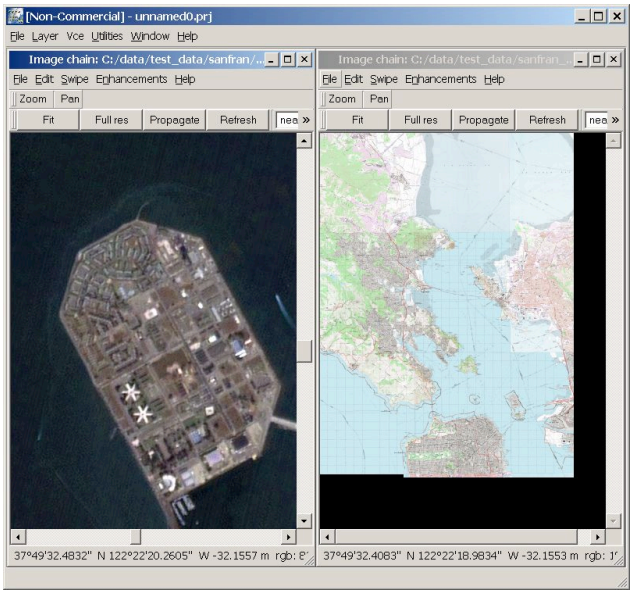


Figure 6 Full resolution of the Landsat San Francisco

Note: Depending on the data set you are using, your view may differ slightly. The test_data.tgz data set has been downsampled and therefore will display at a 15m versus 5m resolution when zooming to full resolution.

In the Image Window Press the **Propagate** Button
The geometric view is updated in all open windows to the same scale and orientation. You can toggle the cursor modes between zoom and Pan with the icon buttons at the top of the display for general navigation.

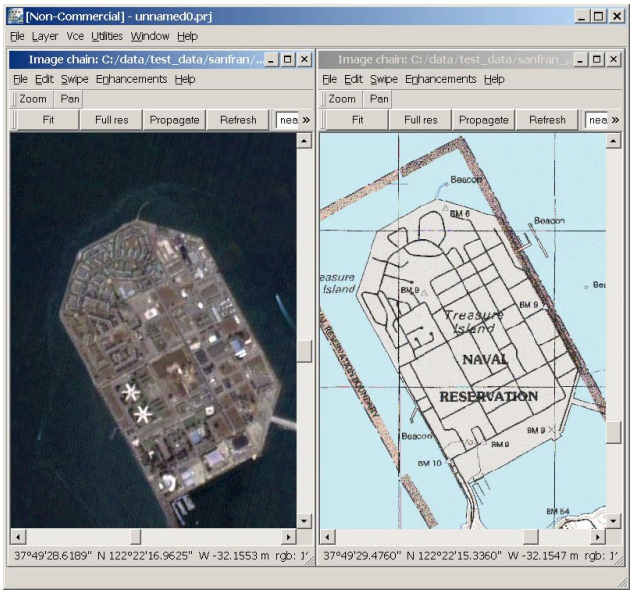


Figure 7 Full resolution of the Landsat San Francisco scene propagated to all other displays.

With the displays zoomed to treasure island, as shown above, blend the views. In the main menu select Blend under the Combine submenu under the Layer menu.

Layer->Combine->Blend

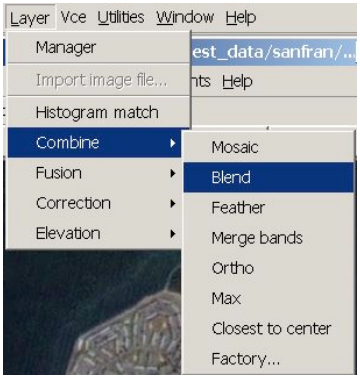


Figure 8 Combine->Blend menu option

A dialog box will appear with a list of image chains that can be combined. Select both the map and the image and press the **Apply** button.

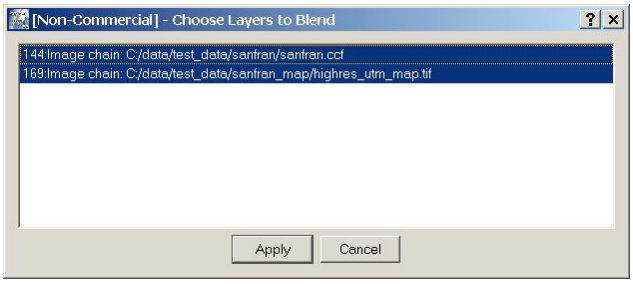


Figure 9 Blend layer chooser.

A new display appears that is blended, press the **Fit button** on the new display to get an overview. You will notice that the data sets are blended together in the overlap area.

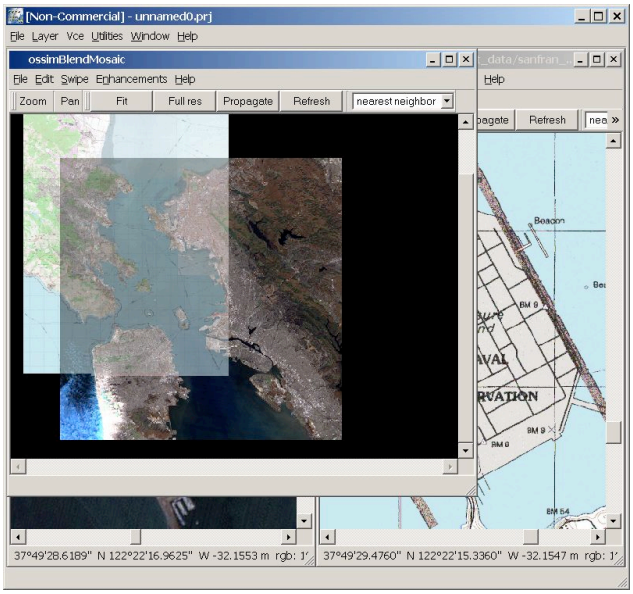


Figure 10 Blended layers.

Tile the windows by pressing the menu option **Window-->Tile** and in the Landsat image window press the **Propagate** button.

The view is propagated to the blend and map windows.

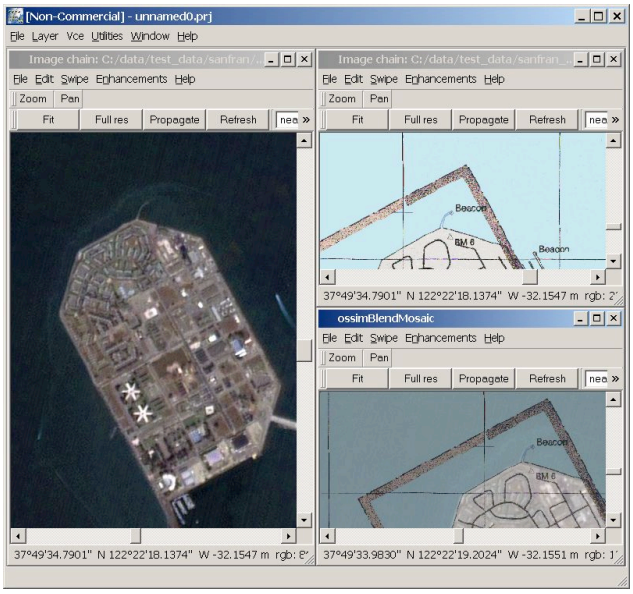


Figure 11 Blended layers tiled and propagated.

Minimize the source windows for the map and Landsat image by clicking in their display minimize buttons. This should leave only the blended display. Now we will create feathered and mosaic displays.

Layer-Combine->Mosaic and select just the map and Landsat image chains

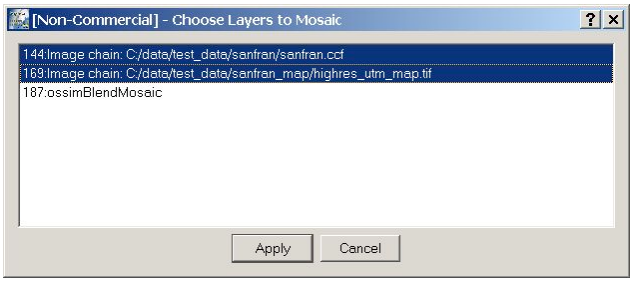


Figure 12 Mosaic layer chooser.

Press the **Apply** button

Next we will go ahead and create a feathered display as well. In similar fashion select the menu option

Layer->Combine->Feather.

Again, just select the first two lines - the source files and press the Apply button. Clean up the displays by selecting **Window->Tile.**

Select any of the displays, press the **Fit** button, then press the **Propagate** button and you should end up with something similar to this:

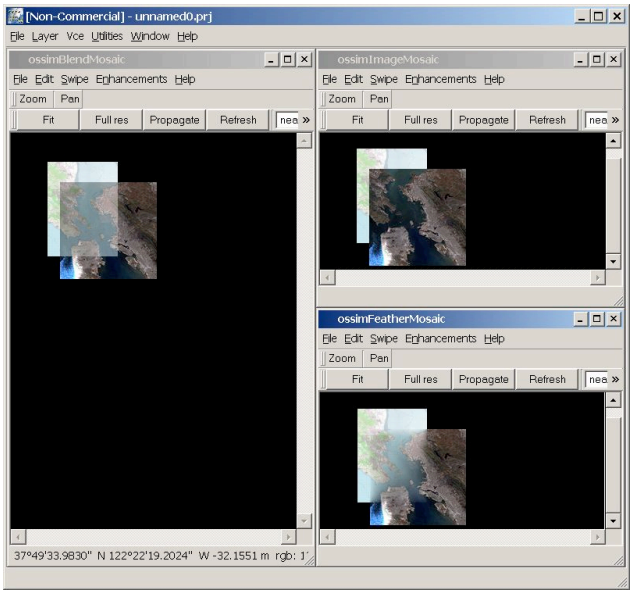


Figure 13 Three basic mosaics.

The difference between these three combiners is how it treats pixels in the overlap area. A mosaic simply chooses one source over the others, the blend averages the values of the pixels, and the feather changes the blend gradually as the distance from the seam increases. To change the weighting, sources, or any relevant adjustable parameters, select the desired display and **Edit->Layers** to bring up the appropriate dialog box.

3.4 Save your project

Main Window File->Save Project as mapblend.prj.

3.5 Summary

This completes the quickstart tutorial for some of the combiner algorithms available in OSSIM and ImageLinker. Using a map and image of San Francisco we demonstrated basic navigation of displays and window management tools in ImageLinker, followed by the creation of various types of mosaics and overlap area combiners. These types of products are very easy to create in the tool sets. OSSIM automatically handles differences in input resolution, projections, file formats and radiometry types. Transformations are performed automatically to produce the product needed by any given display. Adjustable parameters can be accessed and modified through the Layer->Edit menu of any display. Covered in more detail later, any display can generate a product through the **File->save as** command.

4 Installing

This section describes how to download and install the OSSIM software. OSSIM is hosted at www.ossim.org. There you will find download links for the source code and and binaries. If you are only interested in running the software, then download the binary installer for your platform. If you are a developer and want to build the software yourself, you have the option of downloading the latest stable release version from the download site or checking the latest repository from the CVS tree.

4.1 Binaries

The easiest way to get up and running is to download the binary installer for your platform, double click, and follow the instructions. The binary package installers are available for many of the common platforms. The binary packagers will typically install the OSSIM library in the proper system location, command line utilities, and the ImageLinker and Iview applications. Most users will want to start with the ImageLinker application to get a better feel for the capabilities of the OSSIM package.

4.2 Sample Data

You can download the test data at ossim.telascience.org/ossimdata

demodata.tgz - full rez (large) map and image
test_data.tgz - test data as shown below

Some of the directories you should see listed in test_data.tgz are:

doqq
doqq_overlapping
drg
elevation
geoid03
geoid96
geoid99
katrina
orthoigen
sanfran
sanfran_map

These test sets are used throughout the users guide. Each directory is explained:

doqq directory is used to demonstrate the fusion algorithms(see section 4.6.4 Fusion below). You should see a 5 meter color image and a 1 meter grey-scale image of the same area. This is used to simulate a 1 meter high-resolution color product by using the 1 meter grey-scale image for detail enhancement.

doqq_overlapping directory has 4 tiles that overlap each other. This is used to show different mosaics.

drg directory holds an image without geometry. This is used to show how to drop ground points to create an ossim projection (see section 4.2.9).

elevation directory holds a couple 30 meter cells SRTM cells.

geoid03 directory holding NGS distributed geoid 2003 grids.

geoid99 directory holding NGS distributed geoid 1999 grids.

geoid96 directory holding the geoid 96 grid distributed by geotrans.

katrina directory holding some sample aerial data for hurricane Katrina. Example of the Applanix sensor model.

orthoigen directory holds the sample keywordlists and tile data files to run the samples in the orthoigen section 7.2.

sanfran directory holds a 5 meter color product originally created from a color Landsat and Spot fusion

sanfran_map directory that contains a UTM map of the San Francisco area.

4.3 Source Code for Developers

The main site for accessing the source download link is at www.ossim.org. Selecting the downloads will take you to the sourceforge project tree. The released source code is available for download and also contains dependency source code.

For CVS access we host and own the following modules:

ossim
ossim_qt
ossim_plugins
libwms
osgPlanet
osgplanet_qt

4.4 Preferences Setup

OSSIM preference file are global preferences for all command line a GUI applications. Preferences are enabled either by passing the file as command line using the –P option or bey specifying it as an environment variable OSSIM_PREFS_FILE. Assume you have a preference file stored at /home/myaccount/ossim_preferences then you can set your environment variable OSSIM_PREFS_FILE=/home/myaccount/ossim_preferences or pass as command line options –P /home/myaccount/ossim_preferences.

4.4.1 Elevation Setup

OSSIM supports loading of SRTM and DTED elevation data.

Let's assume you placed the test_data under c:\ossim\test_data

```
elevation_source1.filename: c:\ossim\test_data\elevation\srtm\1arc
elevation_source1.type: srtm_directory
```

Note: elevation_source1.type is optional and if not present it will use auto-detection to determine the type. The other supported type is dted_directory, srtm_cell, and dted_cell. For an up-to-date keyword list template please look in source distribution ossim/etc/templates/ossim_preferences_template for a complete list of supported detectors.

If you have other source directories for elevation you just increment the number and add another one:

```
elevation_source2.filename: …..
elevation_source<n>.filename: …
```

You can mix SRTM and DTED, and OSSIM will handle it. Note: Elevation lookups will be in the order you specified. So if you have SRTM 30 meter and DTED 30 meter and you have a cell that overlaps a lat lon value then it will use the first one in the list and if the height is NULL/Invalid then it will go to the next manager in the list until a valid height is found. Currently, they are assumed to be relative to a geoid grid (Mean Sea Level).

Other keywords that can be controlled are:

```
elevation.enabled:  true
elevation.auto_load_dted.enabled:  true
elevation.auto_sort.enabled:  true
elevation.compute_statistics.enabled:  true
```

---

5 Sample ossim_preferences file

```
// $Id: ossim_preferences_template,v 1.11 2003/03/27 20:51:20 gpotts Exp $
//
// Description:  ossim_preferences_template
//
// This file will be automatically loaded by ossim applications provided the
// environment variable "OSSIM_PREFS_FILE" is set to point to some form of
// this file.
//
// Note:  c++ comments "//" can be used to comment a line.
//
// To set the environment variable for automatic preference file loading:
//
// This assumes a preference file in your home called "ossim_preferences".
// Typically this would be put in a dot file read at startup of a shell.
//
// tcsh and csh users:  setenv OSSIM_PREFS_FILE ~/ossim_preferences
//
// bash and sh users:   export OSSIM_PREFS_FILE=~/ossim_preferences
//
// windoze users:       I'll have to look this up...
//
// You can also use the "-P <preference_file>" option at application startup.
// where <preference_file> is full path and filename.
//
// ------------------------------------------------------------------------

// ---
// Keyword:  dted_directory
// Path to DTED elevation cells.  Can have more than one diretory.
dted_directory1: /Data/ele1/DTED3arc
dted_directory2: /Data/ele1/DTED1kps
// ---


// ---
// Keyword:  dted_cell
// Path to a DTED elevation cell.
// ---

// ---
// Keyword:  default_elevation_path
// Default path for the elevation manager popup "Add" to start at.
default_elevation_path:  /Data/ele1
// ---

// ---
// Keyword:  elevation.enabled
```

```
// If disabled calls to the elevation manager getHeightAboveMSL and
// getHeightAboveEllipsoid will return a null height.  (default=true)
// Use: “true”, “yes”, “y” or “1” to enable,
//      “false”, “no”, “n” or “0” to disable.
elevation.enabled:  true
// ---

// ---
// Keyword:  elevation.auto_load_dted.enabled
// Enable autoloading of dted elevation as need from above directories.
// (default=true)
// Use: “true”, “yes”, “y” or “1” to enable,
//      “false”, “no”, “n” or “0” to disable.
elevation.auto_load_dted.enabled:  true
// ---

// ---
// Keyword:  elevation.auto_sort.enabled
// Enable sorting of elevation cells on an add by the lowest(best) mean post
// spacing. (default=true)
// Use: “true”, “yes”, “y” or “1” to enable,
//      “false”, “no”, “n” or “0” to disable.
elevation.auto_sort.enabled:  true
// ---

// ---
// Geoid support
// ---

// GEOID 99:  Set keyword to the directory containing the GEOID 99 grids.
geoid_99_directory:  /data/ele1/geoid/geoid99

// GEOID EGM 96:  Set keyword to the path to the egm96.grd
geoid_egm_96_grid: /data/ele1/geoid/geoid96/egm96.grd
// ---

// font support
//
//font.file1: /usr/share/fonts/default/Type1/b018032l.pfb
//font.dir1: /usr/share/fonts/default/Type1
//  for MacOSX
font.file1: /System/Library/Fonts/Helvetica.dfont
font.dir1: /System/Library/Fonts

// plugin support
// plugin.dir1: /work/gpotts/ossim-plugin
// You can also list files individually
plugin.file1:/Users/Shared/Development/imglnk/lib/libossim_plugins.dylib
```

```
//---
// Keywords for plugin registration.
//
// plugin.dir1:  < directory where plugins are >
//
// you can also list by individual file names
//
// plugin.file1: < full path and file name >

// --------------------------
// Some keywords for the OSSIM GUI application
//
//
// igen spec files output./  This is the directory location you would
// like the igen export GUI to default to when outputing spec files
ossim.igen_spec_output_directory: /work/gpotts/igen_test

// this is the igen executable.  Make sure you put the full path
ossim.igen_executable:  /work/gpotts/ossim/bin/igen

// END KEywords for OSSIM GUI
//--------------------------
```

5.1.1 Geoid Setup

Current version supports the 96, 99, and 2003 grids. The 99 and 2003 grids are distribution from NGS and the 96 grid was a distribution from Geotrans. Assuming you have downloaded the test_data and have it saved to c:\ossim then you should have a geoid96, geoid99 and a geoid03 under the c:\ossim\test_data directory. For 96 geoid support use

```
geoid_egm_96_grid: c:\ossim\test_data\geoid96\egm96.grd
```

The 1999 grids and 2003 grids are NGS distributions and use the keyword:

```
Geoid_ngs_directory: c:\ossim\test_data\geoid99
geoid_ngs_directory.byte_order: big_endian
```

Or

```
Geoid_ngs_directory: c:\ossim\test_data\geoid03
geoid_ngs_directory.byte_order: big_endian
```

In future releases we will support multiple geoid directories and searching for correct geoids to use. For now you can only use one or the other.

5.1.2 OSSIM Planetary Viewer Setup (osgPlanet)

The windows binary installer will automatically setup the preference file with a default low-resolution background reference image. This will be used as the base texture map for the earth. For other platforms the keywords may have to be added manually to your preference file:

```
ossimosgplanet.background.type: local
ossimosgplanet.background.file0: C:\Program Files\Common Files\ossim
\images\earth.tif
ossimosgplanet.background.transparent_color_flag:0
ossimosgplanet.background.transparent_color: 0 0 0
ossimosgplanet.background.opacity: 255
```

The type currently should be local and the files can be listed with first keyword starting at file0 and going to file<n>. You can have any number of images as a background reference, and can be any image loadable by the OSSIM core library.

5.1.3 Plugin setup

At the time of this document a **G**eospatial **D**ata **A**ccess **L**ibrary (**GDAL**) was the only supported plugin. I t gives access to a host of other data formats. Under windows this is automatically setup for you. Here is a sample of the keywords for unix:

```
plugin.file1: /Users/Shared/Development/ossim_plugins/lib/
libossimgdal_plugin.dylib
```

If you are working from the latest snapshot there is a registration plugin started:

plugin.file2: /Users/Shared/Development/ossim_plugins/lib/libossimreg_plugin.dylib

6 ImageLinker

ImageLinker is a GUI based application demonstrating the power of the OSSIM open source software library. It is one of many programs that have been built on top of the functionality in OSSIM. Users are encouraged to register online at http://www.ossim.org and join the collaborative community there. The site is the focal point for software development, user support, and related information.

ImageLinker is an application for viewing and processing remote sensing and Geographical Information System (GIS) data built on the underlying OSSIM library. Capabilities include advanced remote sensing, photogrammetry, and image processing. Underlying design goals and capabilities are covered in more detail in the Design Goals section of this document.

ImageLinker brings out many of the underlying functions through commands that are accessed through pull down menus and dialogs. ImageLinker also allows the user to construct and manipulate complex image flows through the Visual Chain Editor (VCE). The VCE allows the user to dynamically connect and re-order multiple “image chains” through the visual chain editor canvas. Each function is represented by a thumbnail that reflects the state of the imagery at that point. Users can double click any function to access custom dialogs and parameters. Projects can be saved and retrieved at will and later used as processing templates for batch processing.

ImageLinker also supports a plugin architecture that allows new functions and processes to be loaded at runtime. The design of the plugins allows scientists and developers to quickly add new functionality without the need of learning the internals of the OSSIM software libraries.

In addition to viewing and manipulating image chains, ImageLinker can generate output products by passing the parameters to the igen command line tool (called through menu commands).

ImageLinker and OSSIM are open source software projects that are being widely deployed in various government and commercial organizations.

6.1 File Menu

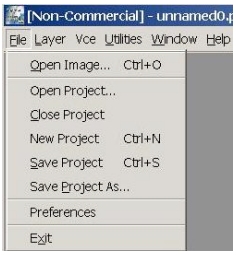


Figure 14 File menu from Windows version of ImageLinker.

The project menus contains all of the commands necessary to loading, saving,closing and creation of new projects. Project files can be thought of as sessions. Multiple images can be loaded and filters and parameters can be applied to their respective image chains. All of these settings including window positions and user settings are captured and saved at will when the user saves a project. A project typically has a .prj extension.

6.1.1 *Open Image*

File->Open Image will bring up a file browser dialog. This is a multi-selection file browser allowing one to select more than one image to import. Drag-and-drop is supported by ImageLinker and is another alternative to the file browsing option for data loading. All data loaded into ImageLinker will be stored into the global Layer Manager.

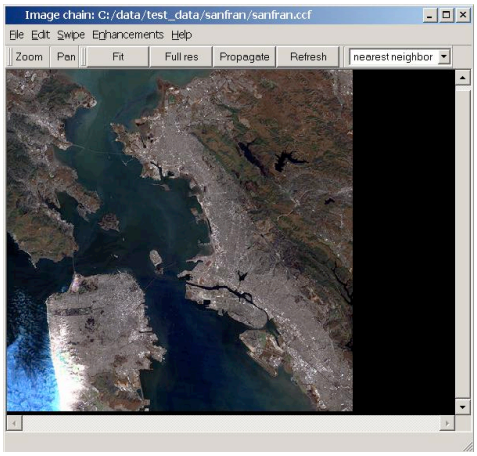


Figure 15 San Francisco image opened through the "File->Open Image" menu option. Image is part of the test data distribution under sanfran/sanfran.ccf.

The display windows have a number of capabilities and menus that can manipulate the underlying image chain for the display.

Each display window contains a title bar that denotes the source. Display menus and menu items operate on the underlying image chain for the window. The window also has a number of modes for panning and zooming. When a display is initially created, it determines if there is an underlying geographic context associated with the imagery. If none is available through associated metadata or internal geo-specific tags, a dialog is presented to the user asking if a default geographic projection should be assigned to the image.

6.1.2 Open Project

The Open command invokes a file navigation dialog allowing the user to navigate the file system and open a .prj project file. A project file saves the state of your session so that you can retrieve it later. OSSIM also supports drag-and-drop of project files. It will first close the current project and open the new project file.

6.1.3 Close Project

Closes the active project and asks if you would like to save the project file.

6.1.4 New Project

Creates a new project and associated project window.

6.1.5 Save Project

Saves the current project and invokes a dialog box if the project hasn't been assigned a name.

6.1.6 Save Project As

Invokes a Save dialog box and allows the user to assign a new name to the project.

6.2 Display Menus

When the **File->Open Image menu** is invoked the image comes up in a display window. Several menu options are available for advanced image manipulation.

Figure 16 Image display menu options and toolbar buttons.

On the top row we have the basic file, edit, swipe, enhancements, and help menu options. On the second row we have a toolbar with quick access to zooming, panning, fitting, full resolution, propagate, refresh and specification of the resampling type. Each are explained in further details.

6.2.1 Zoom and Pan Modes

The window palette contains two icons for selecting zooming and panning modes for the display. Select the mode then control by clicking with the left mouse button in the content of a display.

When in the zoom mode, clicking in the window will do a power of 2 zoom in. Shift click will cause a power of 2 zoom out. Dragging a rectangular area will zoom the display to that area of interest and if the shift is used then it will zoom out.

When in the pan mode, clicking in the window will re-center the view to the clicked position. This is not the case if the image is already "fit" to the current view window.

6.2.2 Fit

The **Fit** button calculates and scales the image so that the entire coverage is visible in the display window as shown above in **Figure 15**. In this example a Landsat image has been scaled to fit in the display windows.

6.2.3 Full Res

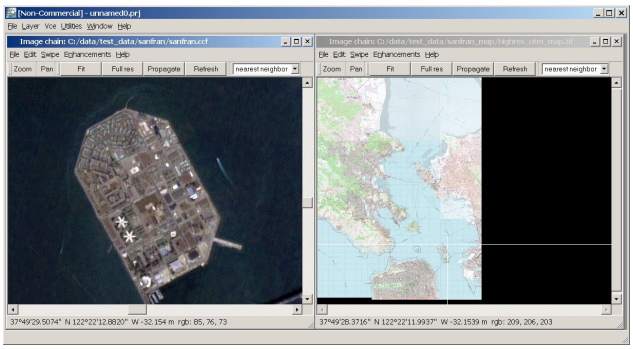


Figure 17 Click on treasure island in the left image and then click the full res button.

The "Full res" button scales the image to full resolution centered about the last cursor click in the window. Each geographic image has a full resolution parameter, which defines the native resolution mode of the data set. In this example the Landsat image on the left was zoomed to full resolution.

6.2.4 Propagate

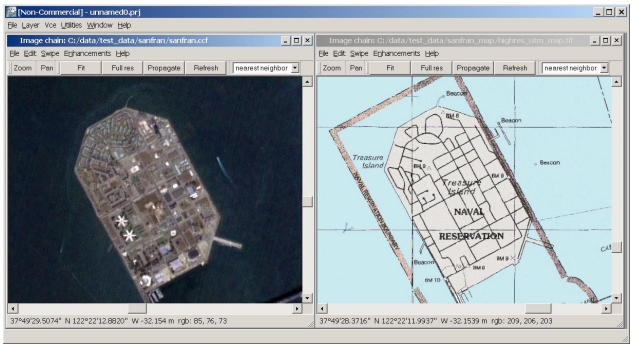


Figure 18 Propagate the view of the left image to all other opened displays.

The propagate mode syncs the views of coincident geographic data sets to the same scale and resolution as the selected window. It transmits a command to the other windows to present themselves in the same view as the window where the button was pressed.

In this example the full resolution view of the Landsat image on the left was propagated to the map on the right. Both images are presented in the same scale and area of coverage.

7 View

Accessing **Edit->View** will bring up the view dialog box.

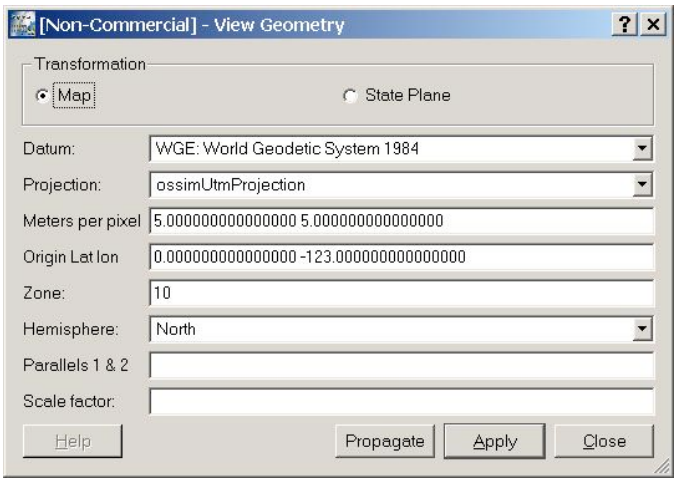


Figure 19 View dialog box.

This dialog displays and modifies the view projection of the displayed image. Currently only map products are supported. Transforming to another projection / datum is as simple as selecting the desired output from the pull down menus. Parameters for resolution, origin, … etc are also available from this dialog. The OSSIM application is projection and resolution independent. All transformations and resamplings are performed automatically to meet the needs of any display. To apply changes you can either hit the return key or click on the apply button. The propagate button takes the same action as the propagate button on the display window. It will take the current view setting and propagate to all other displays.

7.1.1 Geometry Adjustment

Geometry adjustments allow for manual tweaking of mis-registered images. If the input is coming from a sensor supported by OSSIM you can apply geometry adjustments to the image. Supplied under the katrina directory are two Applanix images that are loaded through the Applanix sensor model and can be fine-tuned with the geometry adjustment interface.

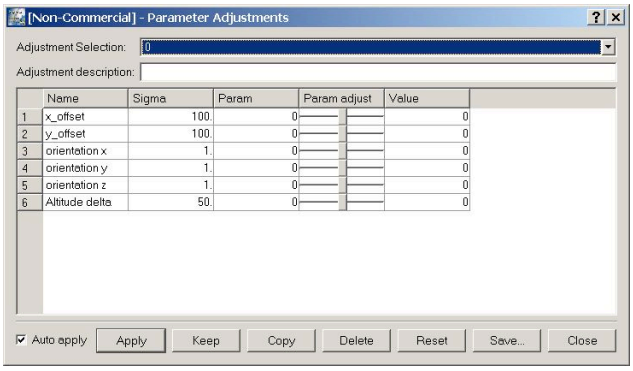


Figure 20 Geometry adjustments for the katrina image.

The Adjustment selection allows you to switch between different adjustments. Each adjustment can have a description by editing the Adjustment description field.

This dialog is available for sources that are derived with sensor models. OSSIM has support for rigorous sensor models and adjustable parameters. Each parameter has 5 attributes: Name, Sigma, adjustable param, param adjustment slider, and the calculated value. Not shown is the parameters center. The calculation of each parameter is computed as:

```
value = center + param*sigma
```

Each field can be edited and changed. The slider is there for convenience and if you need to have more precision then you need to hand edit the field in the

param column. The bottom of the dialog we have the buttons apply, keep, copy, delete, reset, save, and close.

The **apply button** will apply the current adjustments to the image. This is not needed if the auto apply button is enabled. If the auto apply is enabled then the image changes immediately when any parameter changes. This is true by default.

The **keep button** will make a copy and re-center the adjustments. So if you move the top slider over and hit “keep” button it will make a copy of the adjustment, adjust the center to that value, reset the param back to 0:

```
center = center + param*sigma
param = 0
```

You can go back to the previous adjustment by selecting the adjustment number in the adjustment selection at the top of the dialog.

The **copy button** simply makes a copy of the current adjustment. The previous adjustment can be retrieved through the adjustment selection list at the top of the dialog.

The **delete button** will delete the current adjustment. If the current adjustment is the first adjustment it will simply reset the adjustment to the initial condition.

The **reset button** will reset the current adjustment to the sensors initial adjustment values.

The **save button** will write the adjustments to disk. This is important if you want your adjustments to stick around for future use or for use by external OSSIM applications. By default it should fill the correct <filename>.geom file in the file browser dialog.

The **close button** will close the dialog box.

7.1.2 Image Chain

In the image window selecting the menu option **Edit->Image Chain** will bring up the image chain editor.

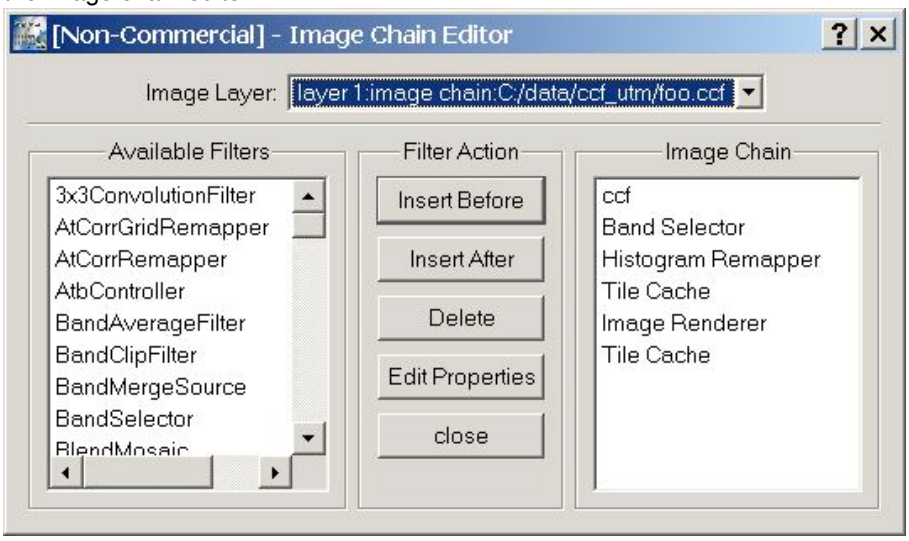


Figure 21 Image window image chain. The left are filters that you can add to the current chain and on the right is the current processing chain for the window.

OSSIM supports dynamic image chains. This will become more apparent later in the demonstration with the visual chain editor (see section 4.7). An image chain typically begins with a geo-spatial file on disk and is connected through a series of band selectors, caches, filters, and resamplers. This dialog shows the current image chain (right side) associated with the particular display. The left hand list contains all the registered filters in OSSIM.

Using the San Francisco image in **Figure 15** we can add an edge detector to the chain by first selecting the “EdgeFilter” in the available filters list and then select the ccf filter on the “image chain” list

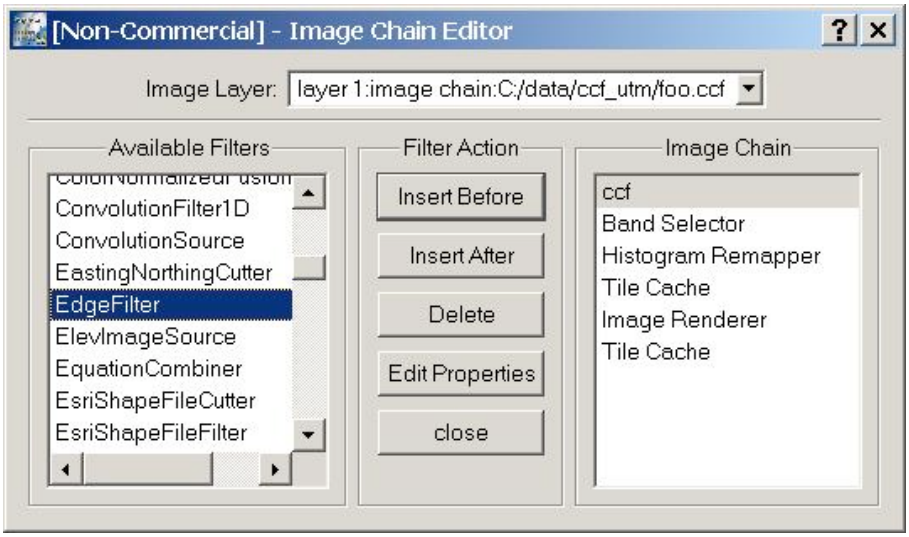


Figure 22 Chain editor. Select EdgeFilter left and select the image handler ccf on the right. Top is the input source followed by the rest of the processing filters.

and then do an insert after

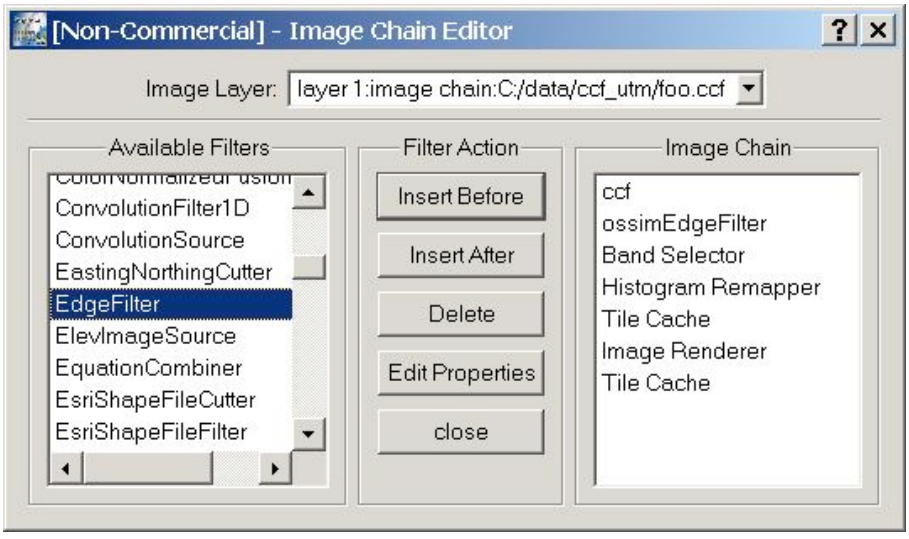


Figure 23 Chain editor. EdgeFilter "inserted after" the selected ccf image handler.

and finally the result.:

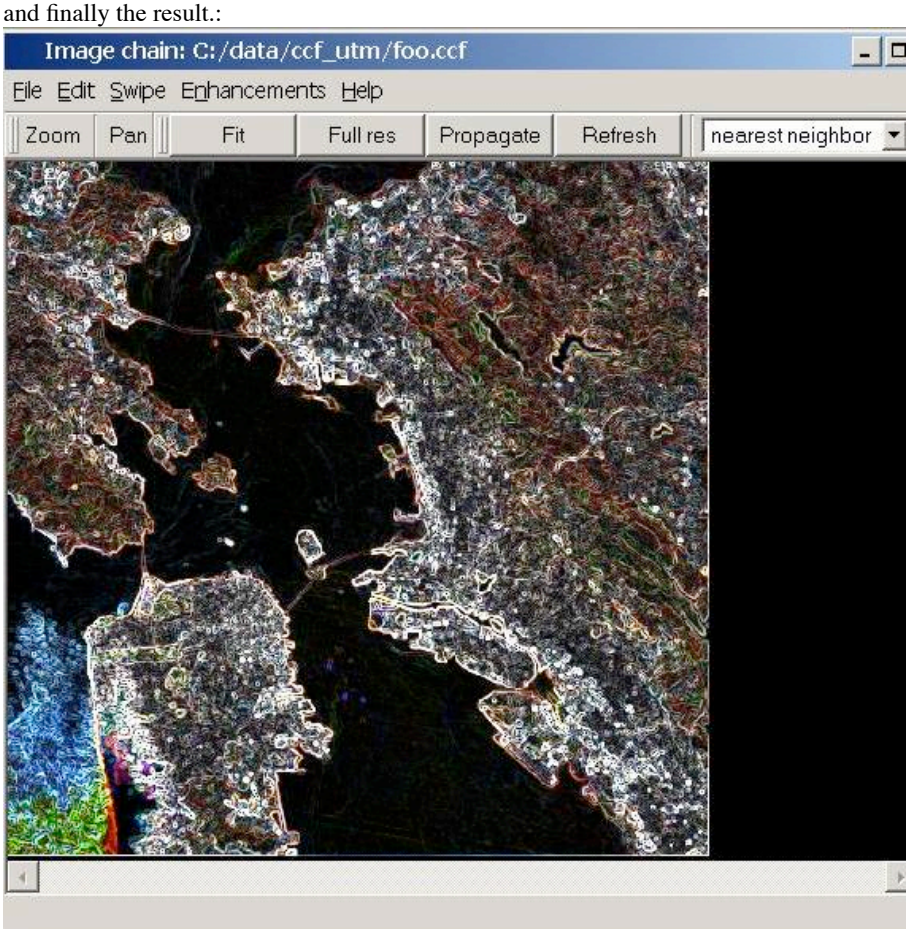


Figure 24 Result of inserting the EdgeFilter.

7.1.3 Image Info

Selecting the **edit->image info** menu option will bring up the image information window for the current image in the display.

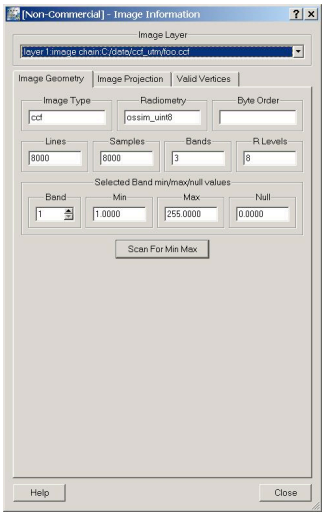


Figure 25 Image information window. Brought up through the edit->image info menu option on the Image display window.

This dialog contains geometry, projection and vertex information on the displayed image. The scan for min max is currently disabled and can only be ran by the external application cmm, which stands for compute min max. Each section is available through the tabbed interfaces on the dialog.

The **Image Projection** tab shows the images input projection information.

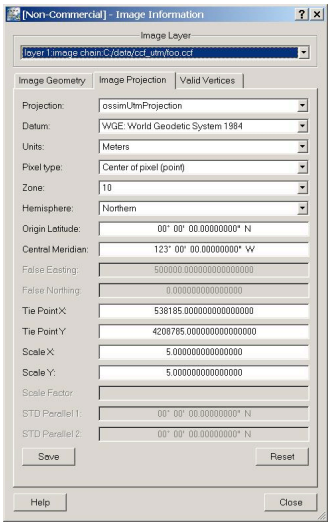


Figure 26 Image information window "Image Projection" tab.

Image Projection information is currently only given for map projected data. Finally, valid vertices are displayed by selecting the **Valid Vertices** tab:

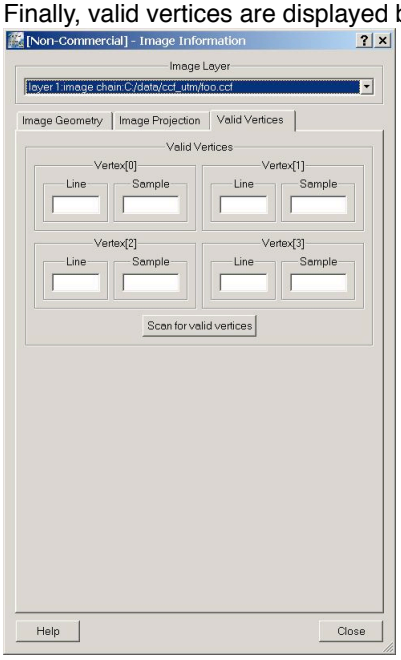


Figure 27 Image information window "Valid Vertices" tab.

The **Valid Vertices** tab displays the images valid vertices. Many time such as Landsat scenes the images have large NULL areas since the image is rotated. The valid vertices can specify the valid corner points. This is currently for visual purposes only and cannot be edited here.

7.1.4 Drop Point GUI

If you would like to follow the steps here please remove the file test_data/drg/battlmn.geom.

If this is not the first time running this part and you want to see how to do a drop point projection then remove the file test_data/drg/battlemn.geom. Next, load the image test_data/drg/battlmn.tif and say yes to create a default projection.

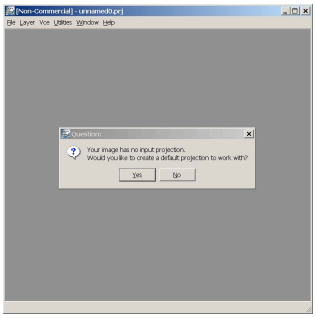


Figure 28 Load image test_data/drg/battlemn.tif

The loaded image should look like this:

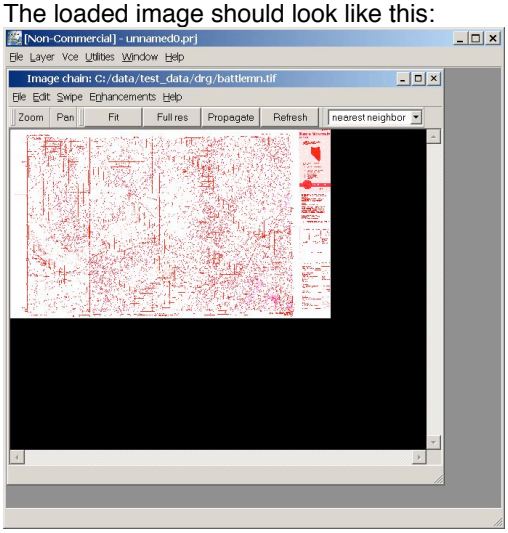


Figure 29 Loaded image

Zoom to full resolution to the upper left corner of the image.

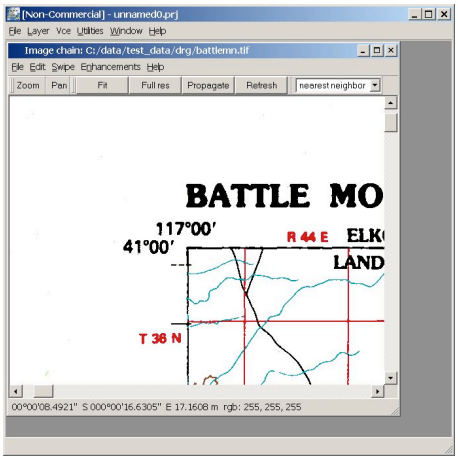


Figure 30 Zoomed to full resolution to the upper left corner.

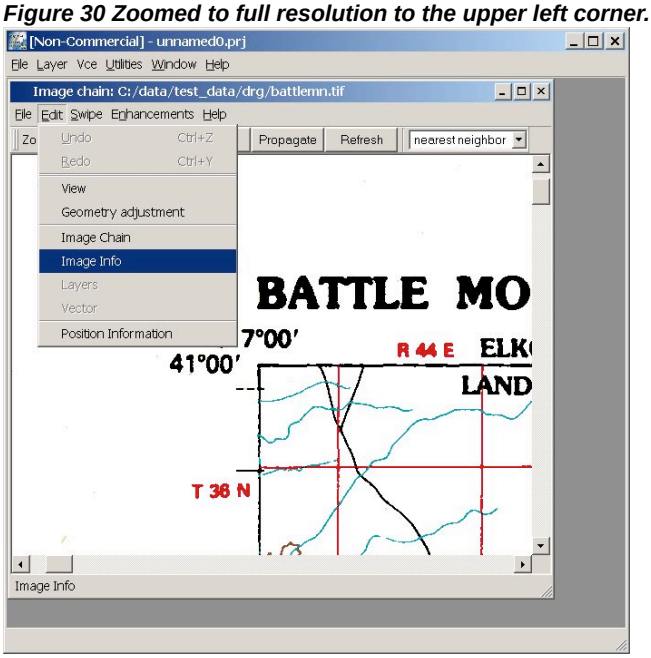


Figure 31 Access ImageInfo menu option

Go to the **edit->image info** menu option and select the **Image Projection** tab and select the ossimBilinearProjection class.

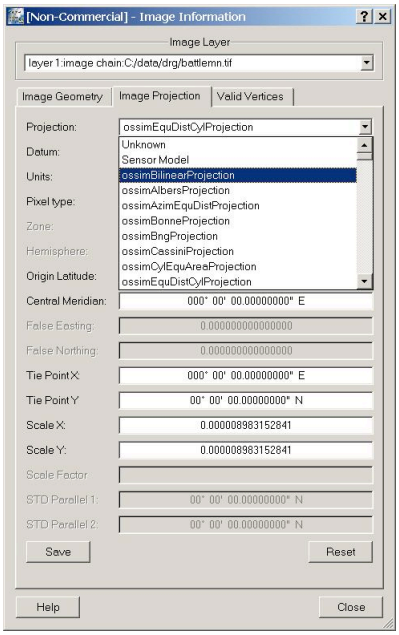


Figure 32 Change to a bilinear projection.

Once selected you will be prompted to edit the projection by using the drop point GUI. This projection requires 4 points.

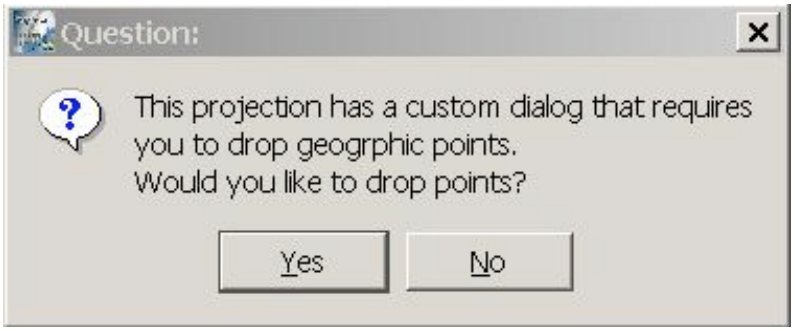


Figure 33 Question box when changing to a bilinear projection.

Select **Yes** and the bilinear projection drop-point dialog will come up.

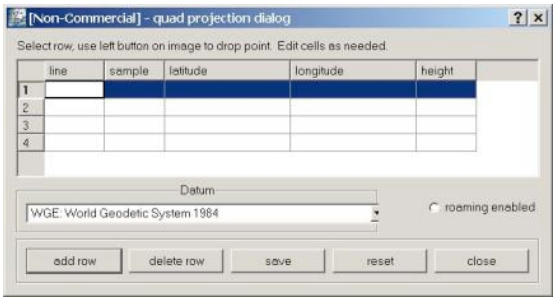


Figure 34 Drop point GUI.

Selecting in the window will add an adjustment to the dialog. Select the in the corner of the cross section where 41 degrees North and 117 degrees west meet.

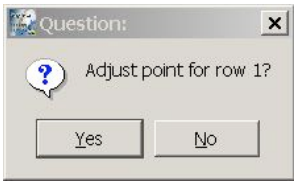


Figure 35 Popup that comes up when selecting, with the left mouse button, in the image window.

Selecting yes will add the line and sample to the selected row. Type in the lat lon location.

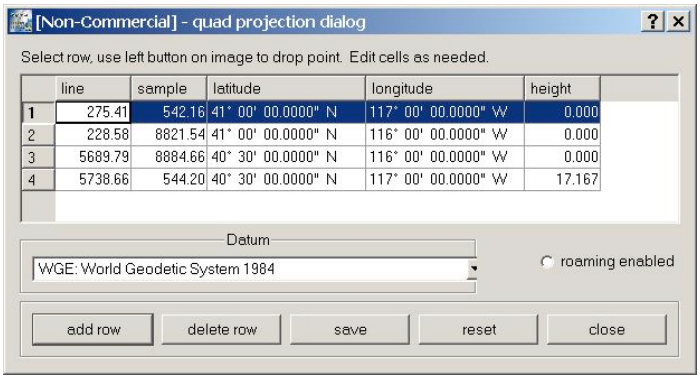


Figure 36 Drop point GUI.

Repeat for all 4 corners of the image and then hit save the geometry to test_data/drg/battlemn.geom. You now have a projection in OSSIM and you can output the product to a geotiff.

7.2 Image Generation

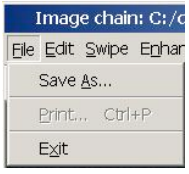


Figure 37 File menu under the Image display.

The **File->Save As** allows the user to set parameters for product output, save those specifications to an external spec file and/or save the image to disk. When the **File->Save As** menu option is hit the product output dialog box comes up:

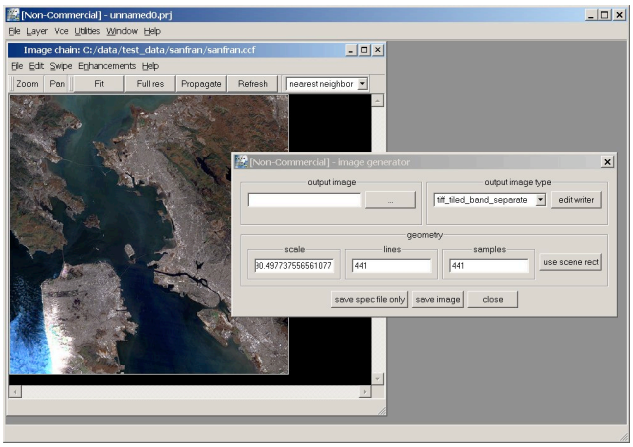


Figure 38 Image generation dialog.

Once this dialog is up it will link to the current display. Initially you should se a white bounding rect covering the entire scene. There are several was to change the dimensions of the rectangle or redoing of interest. Try changing the **scale** field in the Image generator dialog box to 5.

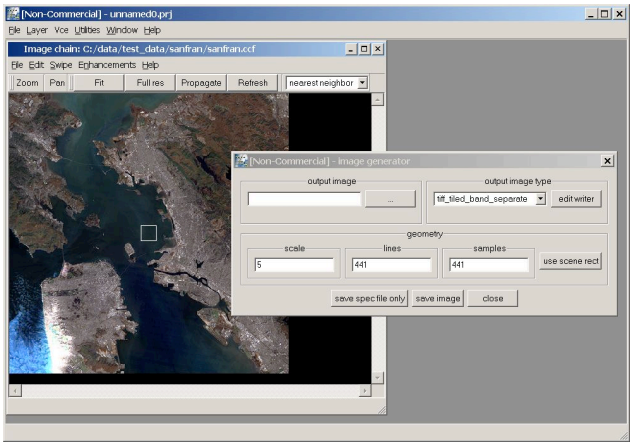


Figure 39 Scale changed to 5 meters. Region of interest box changes dimensions.

Notice the rectangle box changes size to match the scale estimate. Now change the dimensions of the box by entering 1024 in the **lines** field and 1024 in the **samples**. Make sure you hit return in each field or the change will not take affect.

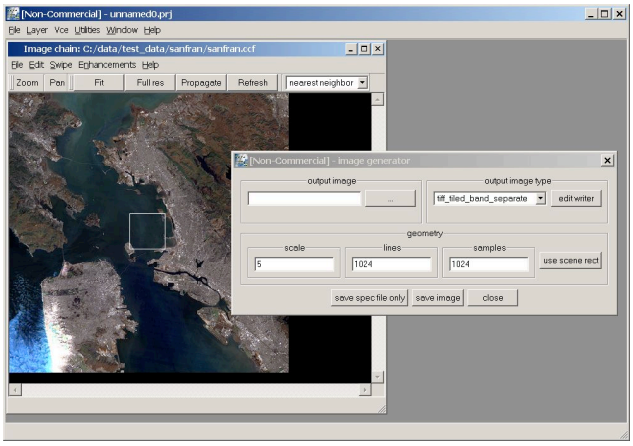


Figure 40 Lines and samples changed to 1024x1024. Region of interest box changes dimensions.

You can also move the box with your middle mouse button. Move the mouse over the box and press and hold down the middle mouse button and drag the box. The box should change to a green color. Once you have the box moved to the desired location release the middle mouse button. In this example we moved the box over Treasure Island.

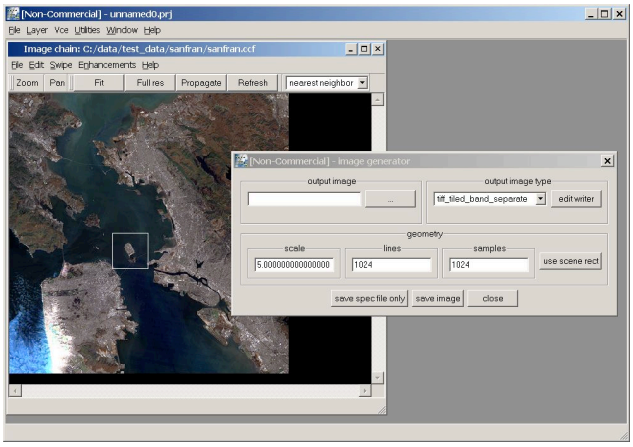


Figure 41 Box moved over Treasure Island in the San Francisco image.

Using the left mouse button can create a custom box. Press and hold the left button down while dragging. A box should be created. When you have reached the desired size release the mouse button.

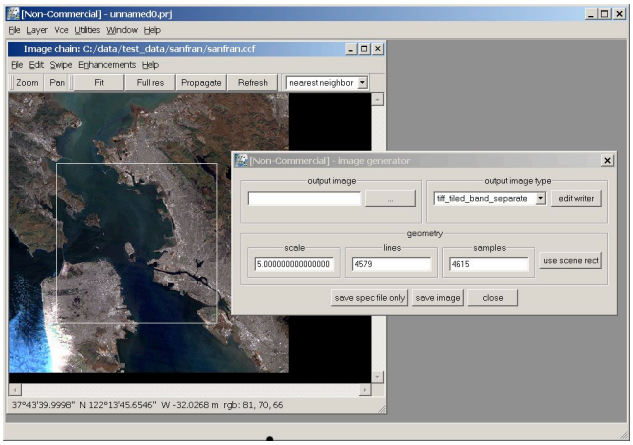


Figure 42 Custom bounding box drawn using the left mouse button.

The u**se scene rect** button located in the Image Generator dialog box will stretch the region of interest to cover the entire scene. The scale is in meters and corresponds to the meters per pixel product output. The **output Image** field has a file browse button indicated by the three dots. When pressed it will open up a file browser to identify the location of your product. If you edit this field manually without the file browser you must hit return in order for the change to take affect. Also note, we currently do not automatically set the correct extension based on

writer type and you must specify it. The output writer type will have a list of output types to choose.

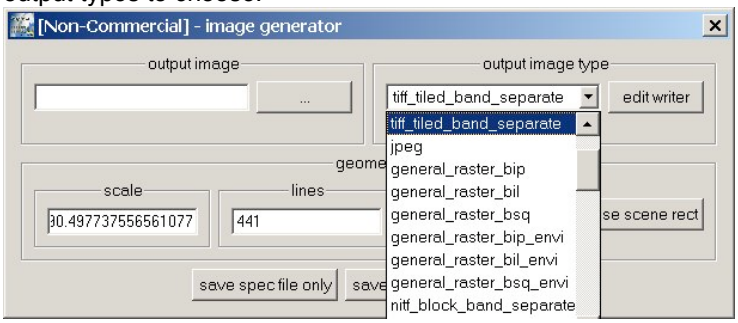


Figure 43 Image generation dialog writer type selection.

Once the writer type is selected, individual properties can be edited by selecting the **edit writer** button:

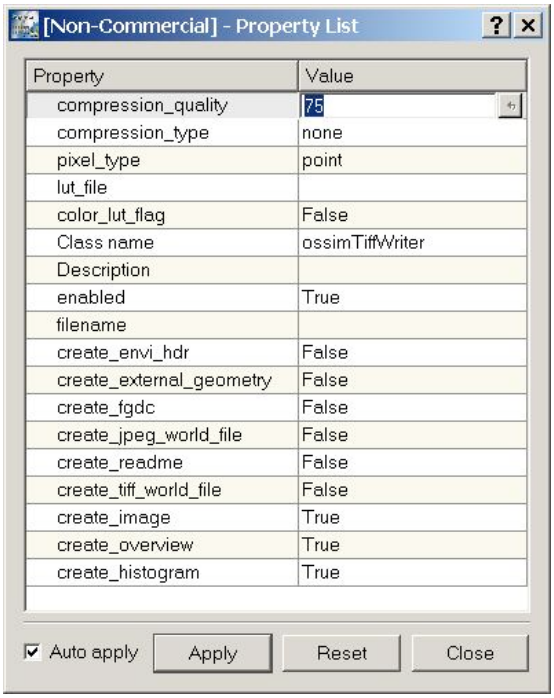


Figure 44 Writer property list.

If the GDAL plugin is enabled then you can select other writer types related to GDAL. For example gdal_HFA will allow you to output an imagine formatted file:

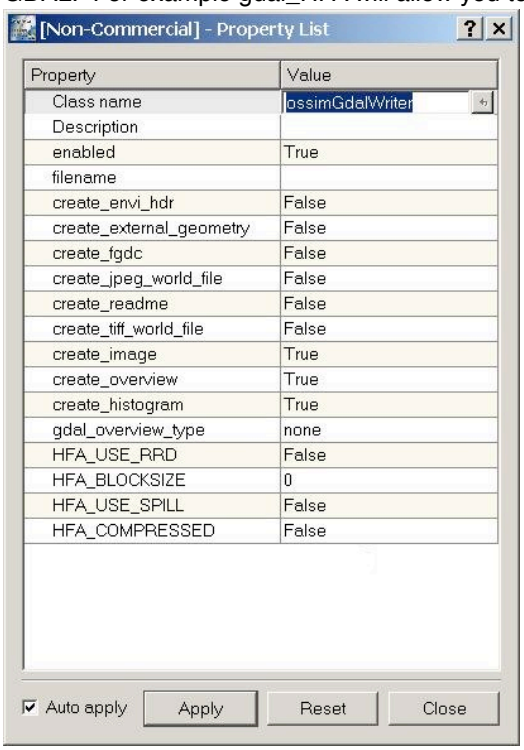


Figure 45 Writer property list for the gdal_HFA output type.

To see the supporting formats of the GDAL plugin see **Main Menu->File->preferences** or under MACS see **Main Menu->ImageLinker->Preferences**.

The final product can be saved by hitting the **save image** button. It will save the image and immediately after saving bring it back up into ImageLinker for further viewing/processing. If you are using imagelinker to create a very large mosaic and your output product will take a long time to execute then it might make since to output the spec file only by pressing the **save spec file only** button. The file you output can be fed into the igen command line application. For example if you saved out a spec file to foo.spec then you can execute it on the command line by igen foo.spec. You must have your path set to the location of the executable or give a full qualified path to the igen executable.

7.3 Swipe Menu

If we have brought in multiple images, also know as layers in OSSIM, we can swipe the datasets together. We first add the layer by accessing the Swipe->Add layer menu option. The second layer is selected through the layer selection dialog box. Two geographically coincident data sets can be stacked and rapidly compared by moving the cursor back and forth over the content region of the window. The top layer is covered and uncovered as the mouse is moved.

7.3.1 Add Layer

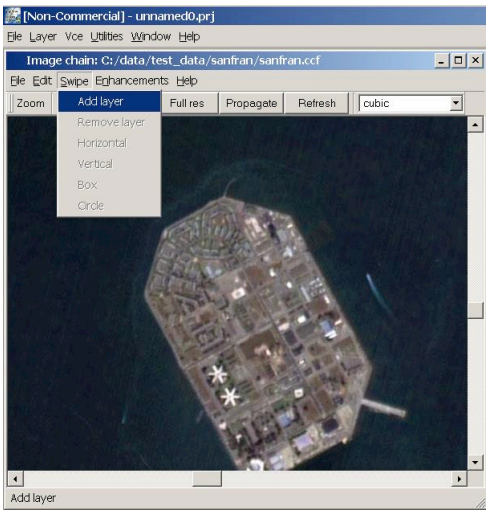


Figure 46 Adding another layer to a display for swiping.

The add layer menu item invokes a layer chooser dialog. Select the layer that you wish to place on top of the current display window and press the apply button

Figure 47 Layer chooser dialog box. Select the second layer for swiping.

The resulting swipe window is finished.

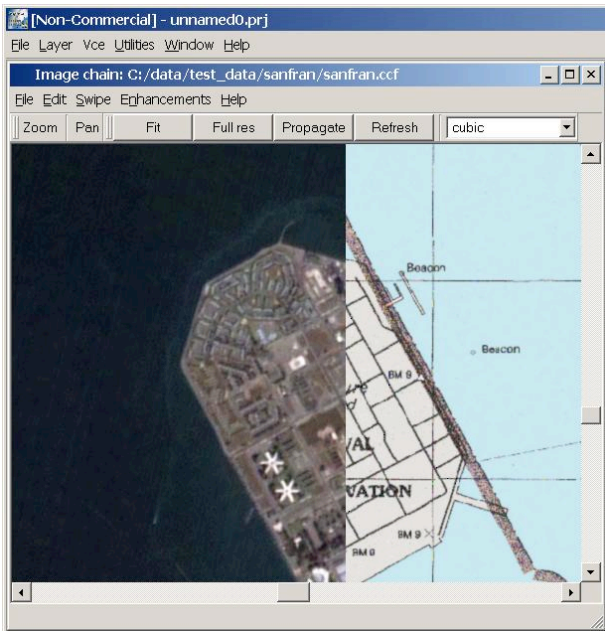


Figure 48 Result of adding a layer for swiping.

7.3.2 Remove Layer

Once an additional layer has been added it can be removed using this command. Simply select the image chain to be removed and click the Apply button.

7.3.3 Horizontal

Swipe modes are ideal for change detection or rapidly comparing two geographic coincident data sets. As the cursor is moved back and forth the top layer covers and uncovers the base layer. The horizontal swipe is the default swipe when adding a new layer. If you are not in horizontal swiping it can be selected by the **Swipe->Horizontal** menu option.

7.3.4 Vertical

The vertical mode is selected through the **Swipe->Vertical** menu option. This give another direction for detecting differences.

7.4 Enhancements Menu

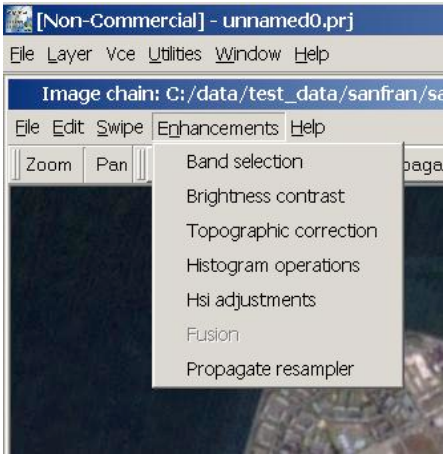


Figure 49 Enhancements menu.

7.4.1 Band Selection

Access the band selector through the Image display menu option **Enhancements->Band selection.**

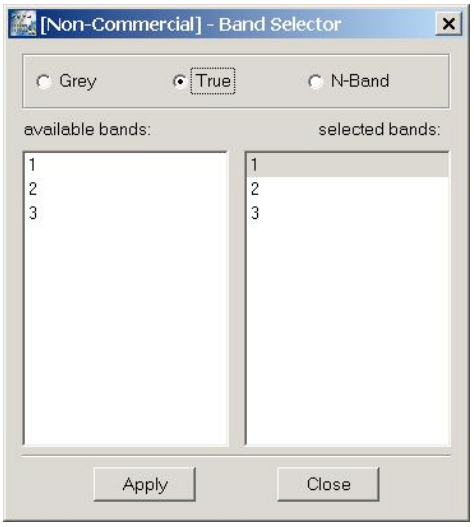


Figure 50 Band Selection Dialog.

The band selection dialog is used for selecting which source bands are to be mapped to designated output bands. Modes exist for grey scale (select one input band), True (select three bands for red, green, and blue channels), and N band (select and arbitrary number of bands. The bands can be reordered as well.

7.4.2 Brightness Contrast

This brings up a brightness/contrast GUI window. A brightness slider and a contrast slider is present to manipulate the current images brightness/contrast levels.

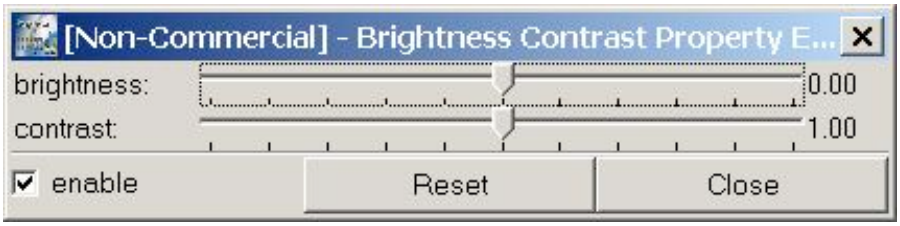


Figure 51 Brightness contrast editor.

7.4.3 Correction-->Topographic

Used for providing topographic illumination correction to imagery based on the normals generated from the underlying elevation surface. Used as an error removal process in preparation for material classification.

7.4.4 Histogram Operations

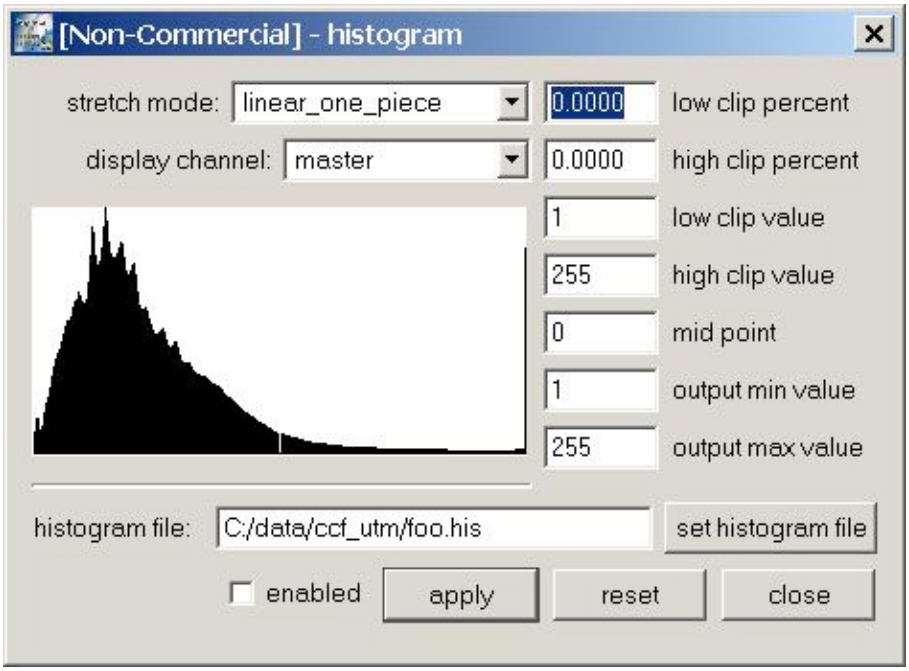


Figure 52 Histogram operation editor.

ImageLinker contains a dialog for viewing and manipulating image histograms. The current version of ImageLinks needs to attach to an external histogram file. If one is not present it will ask if you would like to create one for the input file. An alternative is to use the command line application **create_histo**. The dialog allows the setting and application of clip points and has selectable standard deviation values from the stretch mode list.

7.4.5 Hue Saturation Intensity Adjustments

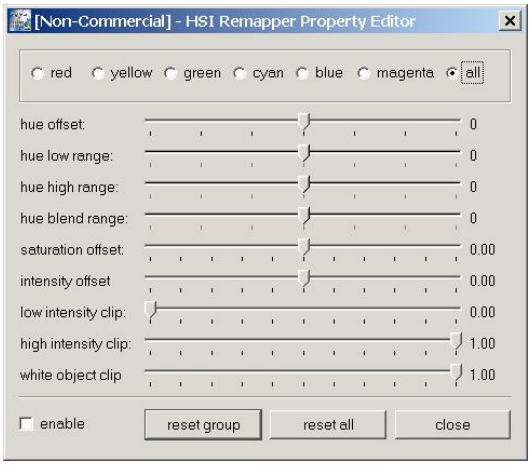


Figure 53 Hue Saturation and Intensity editor.

This dialog provides color adjustments via hue, saturation and intensity sliders. The changes can be applied to individual or multiple channels by selecting the radio buttons on the top. Clip points and white object clip points can also be directly adjusted. The enable checkbox makes the changes active on the parent display.

7.4.6 Propagate Resampler

This command propagates the current windows resampler selection to all other display windows in the project. For demonstration bring in the San Fracisco image and zoom to full resolution into the Treasure Island location and zoom a few times past full resolution so you get a block image and change the resampler type to cubic:

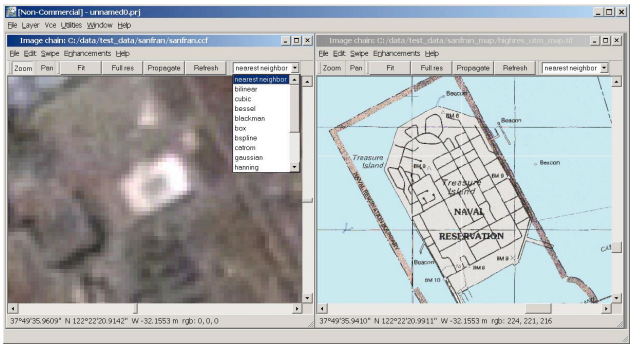


Figure 54 San Francisco color left zoomed to 1 meter.

Select the resampler type in the drop down list box.

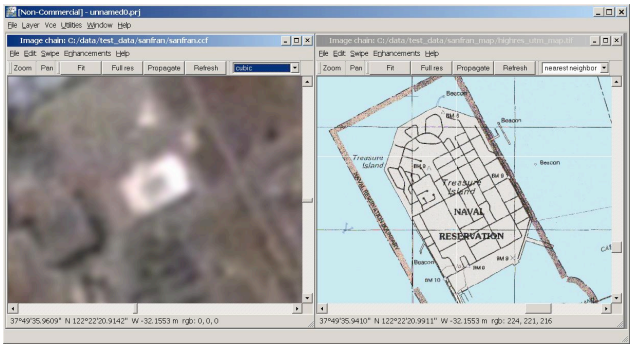


Figure 55 Cubic resampler selected.

Now make sure the view is the same on both sides and do propagate view. Now lets watch the resampler propagate to all other open displays by hitting the menu option Enhancements->propagate Resampler

Figure 56 View and resampler propagated to all open displays.

7.5 Layer Menu

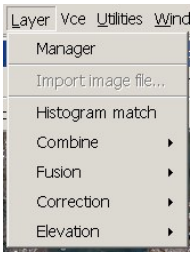


Figure 57 Layer menu found on the Main ImageLinker window.

The **Layer** menu allows creation of other layers by using the loaded data as input. Layers can also be used in other operations to create additional layers. All layers are visible throught the **Layer->Manager** menu option.

7.5.1 Manager

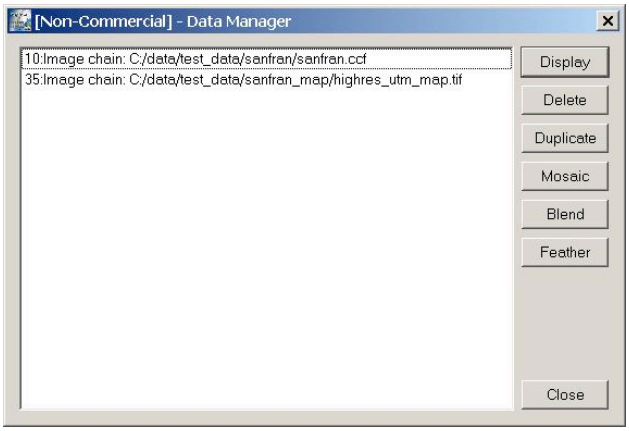


Figure 58 Layer Manager brought up through the "Layer->Manager" menu option. Displays all the current layers in the system.

We have added some quick action buttons on the right hand side indicated by **Display, Delete, Duplicate, Mosaic, Blend, and Feather** buttons.

Button	Action
Display	button will open displays for any selected image. If multiple layers are selected then a display is opened for each layer.
Delete	button will delete a layer in the system. If a display is open for that layer it will automatically close if it is the only layer for that display
Duplicate	button will duplicate the selected layers.
Mosaic	button does an A/B style mosaic of all selected items
Blend	button does a blend of all selected layers.
Feather	button does a feather along the overlaps of all images

7.5.2 Histogram Match

Currently, the matching automatically selects the target histogram to be the first layer. Once set we do not have editor in place to change it. For an example there are four files supplied in the test_data suite under doqq_overlapping directory. Load all four images into the system.

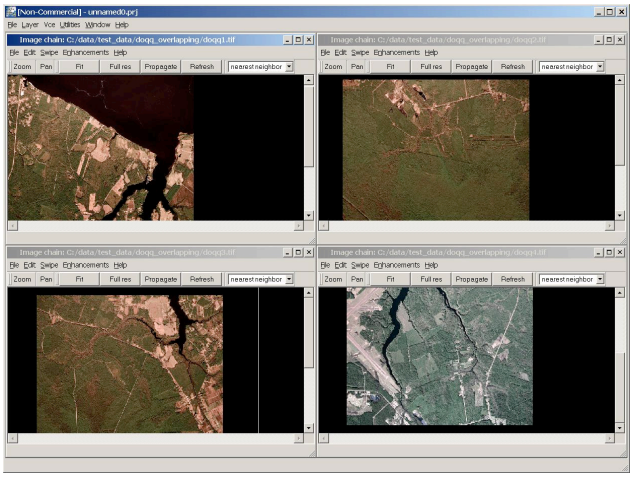


Figure 59 Four overlapping doqqs found under the doqq_overlapping directory of the test_data suite.

Make sure you go ahead and propagate the view from one display to all other displays. This will guarantee all are on the same view plane. Next, select the **Layer->Histogram Match** menu item and it should bring up a selection box and select all 4 images:

Figure 60 Histogram matching layer selection.

When apply is hit there will be 4 new layers added to the layer manager that have two equalization filter with the second filter being the target histogram. The default target histogram is the first layer and we currently do not have a GUI to

edit the target histogram. Once the four new chains are created they are mosaiced automatically together with a feather combiner.

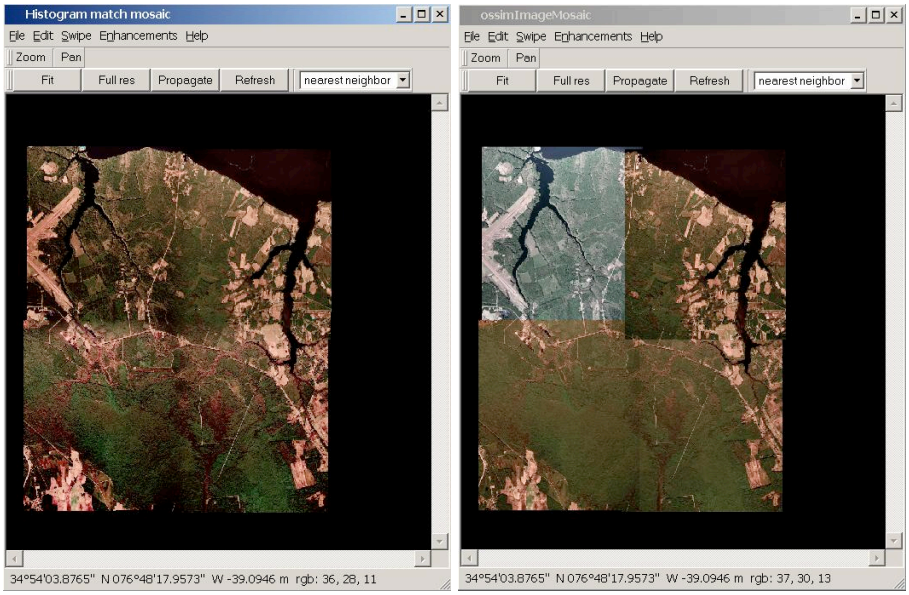


Figure 61 Histogram Match and feathered shown on the left. Original mosaic with no matching and feathering shown on the right.

7.5.3 Combine

Figure 62 Combine Sub-menu option.

The combine sub-menu holds menu items that invoke various image chain combiner methods. The Factory… menu item currently does nothing in the current release of OSSIM.

When selecting any of the combiner types it will bring up the layer chooser window with the title corresponding to the combiner just chosen. This chooser was from the **Layer->Combine->Mosaic** menu option.

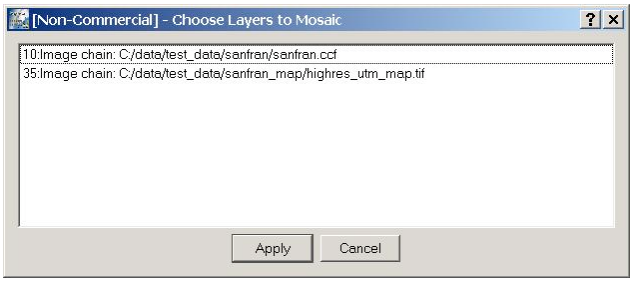
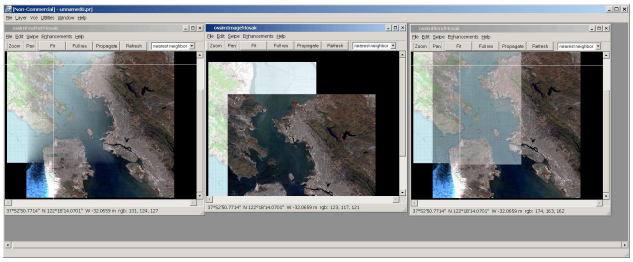


Figure 63 Chooser for any combiner selected. Brings up all layers in the Layer manager.



A couple of examples of different mosaics:

Figure 64 Feather combine left, simple combine middle, and a blend combine

Merge Bands combiner will merge all input bands into a single layer.

Ortho combiner is used if all input data was originally projected to the same view and all that is required is a simple shift to put them together. For example, if all input data was say UTM zone 10 5 meters then they could be put together with an Ortho combiner and remove the resampler out of each chain and put it after the combiner. This saves on a lot of processing.

Max combiner takes all input layers and finds the max pixel for the output of each pixel value.

Closest to center combiner will replace the output pixels with the closes **valid** pixel to the center of the image (See **Figure 105** for closest to center example)

7.5.4 Fusion

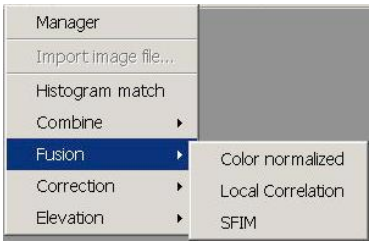


Figure 65 Feather combine left, simple combine middle, and a blend combine

The fusions implemented in OSSIM take a low resolution multi-band image and a high resolution grey/pan band as input to the fusion process. Each algorithm tries to enhance the lower resolution image with the higher resolution details of the pan image. NOTE: The color inputs and the pan inputs could have been from a mosaic layer and you are not restricted to single image layers. Typical fusion may be Landsat Multi spectral with the co-registered pan band.

Color Normalized Fusion is a slightly modified Brovey fusion. It is important to note that this fusion only works on 3 band inputs.

Local Correlation uses local linear regression on a 5x5 window to correlate the pan to the same area on the multi spectral data. The fusion is only applied along correlated areas. This fusion will probably not work well in cross-sensor fusions. This work was derived from the paper.

SFIM or **S**moothing **F**ilter-based **I**ntesity **M**odulation.

The Local Correlation and the SFIM fusion are band separate and will fuse all bands whereas, Color Normalized expects the input to be 3 bands only. When the scale resolution is more than 2x, you will probably want to adjust the fusion parameters through the display window **Enhancements->fusion** menu option.

For an example: in the directory doqq in the test_data suite are two images doqq5_meter.tif and doqq_1meter_grey.tif. This simulates a 5x difference in resolution and we would like to enhance the doqq_5meter color image to the 1 meter grey and create a new high res color product. Examples will be shown for each fusion. Note: the color normalized fusion has no adjustments available but

the Local Correlation and SFIM fusion do have parameters to tweak and is a necessity in this particular example.

First bring both of the images into imagelinker and select the grey scale image and hit the full resolution button. You should be at a 1 meter view. Now propagate the view to the color window by hitting the **Propagate** button. You will notice a blurred image since the color image was a 5 meter product and is now zoomed to 1 meter. Change the resampler type to be something smoother. For this example we chose cubic.

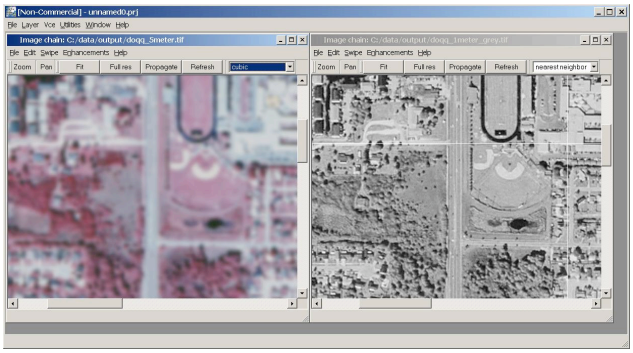


Figure 66 5 meter color image (left) zoomed to the 1 meter resolution of the grey scale image (right). Cubic resampler set for the zoomed color image.

Now select **Layer->Fusion->SFIM** bringing up the layer chooser for the SFIM fusion algorithm.

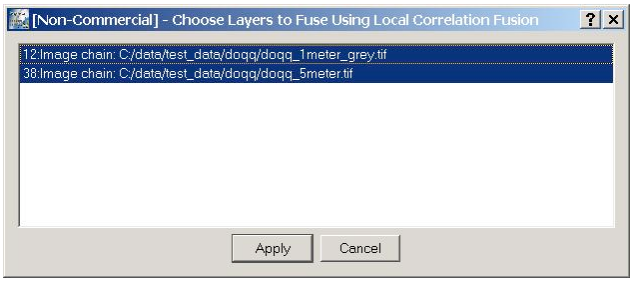


Figure 67 SFIM layer chooser.

Hitting the apply button will initiate the SFIM fusion and bring up a fusion display window.

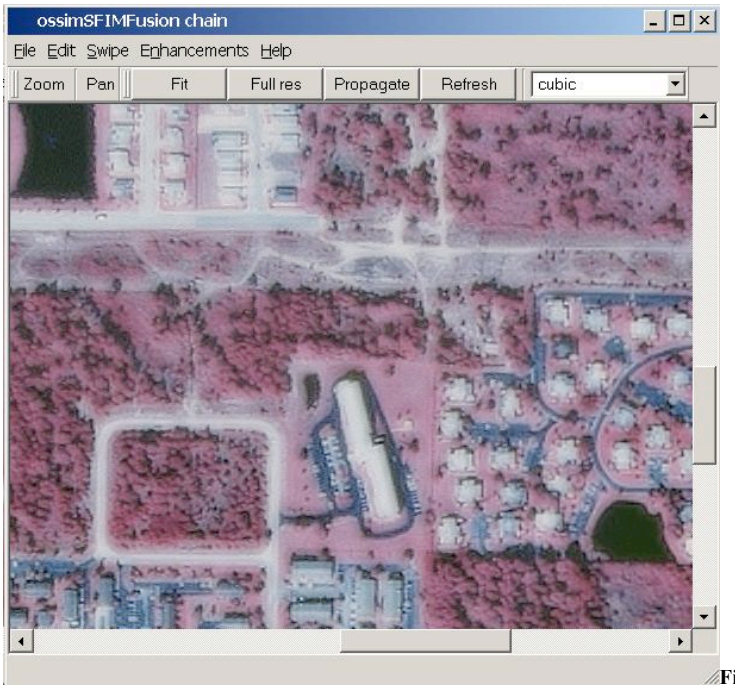


Figure 68 Initial SFIM fusion. Needs editing for 5x scale change through the Enahancements->fusion.

The fusion currently doesn't look that good since the scale changes are not auto detected and assumes 2x by default. In this example we have a 5x scale change. To make the fusion look better we can tweak individual parameters.

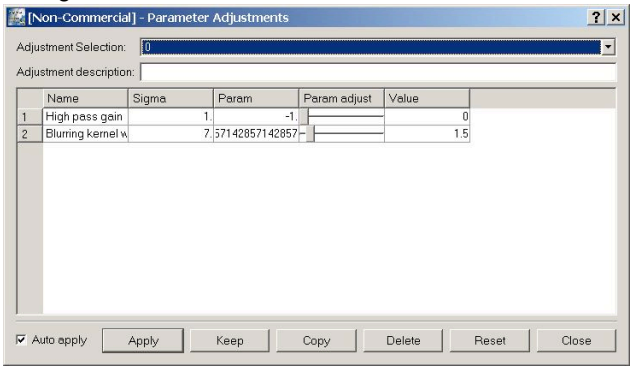


Figure 69 SFIM fusion. Parameters.

Increasing the blurring kernel width (parameter 2) to the value of 5 by editing the value column or by moving the slider to the right gives a better quality fusion. This parameter corresponds to the scale change difference between the multi-spectral and the high resolution grey scale image. The larger the scale difference the larger the kernel needs to be. The second parameter applies a gain to the high resolution pan. Currently the gain is not adaptive.

Figure 70 5 zoomed to 1 meter image using cubic resampling(left). Final SFIM sharpened image with blurring kernel set to 5 using the 1 mter grey scale image for detail preservation.

The local correlation fusion has the following parameters that can be adjusted:

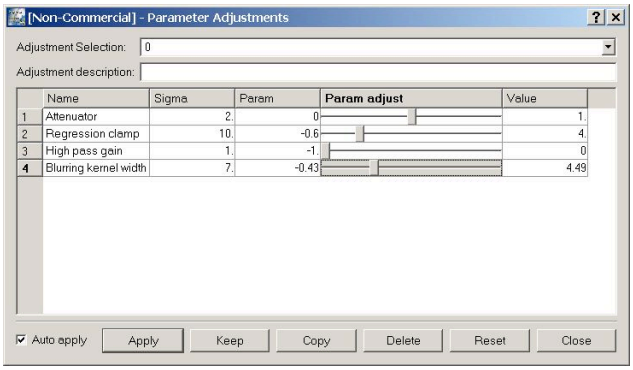


Figure 71 Fusion parameter edits for the Local Correlation Algorithm.

The **attenuator** parameter is a global attenuation on how much information is added back into the multi-spectral dataset.

The **Regression clamp** clamps the slope of the regression equation. Allows one to clamp large slopes.

The **High Pass Gain** runs a high pass over the high-resolution image before applying the fusion.

The **Blurring kernel** width is increased based on the scale change. The larger the scale difference is between the multi-spectral and the grey scale image then the larger the blurring kernel value should be.

7.5.5 Hill shading

In the test data suite we have supplied wo SRTM elevation cells under elevation/ srtm/1arc. Load both cells into the system.

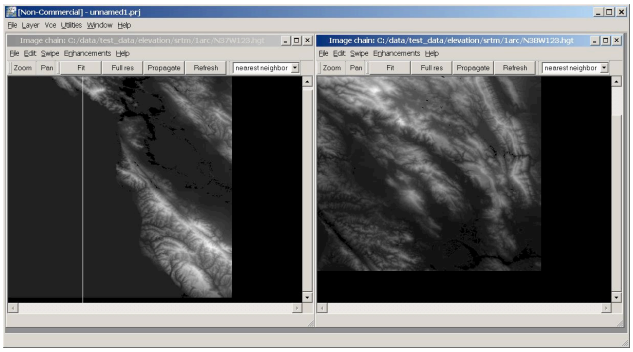


Figure 72 SRTM elevation cells.

Create an ortho mosaic through the **Layer->Combiner/Ortho** menu option and select both SRTM layers.

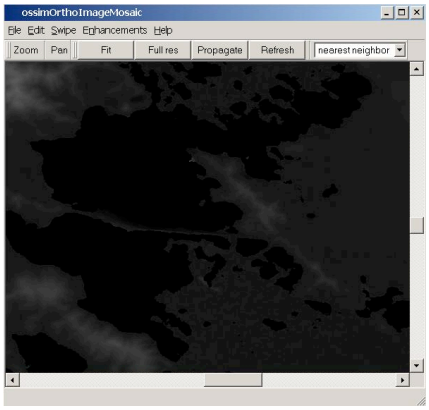


Figure 73 SRTM Ortho mosaic.

The hillshading portion is a 2 step process and future releases will merge these into one. For now, we need to first apply surface normals to the ortho mosaic so it can be used in a shading equation. Select **Layer->Elevation->Normals** menu option

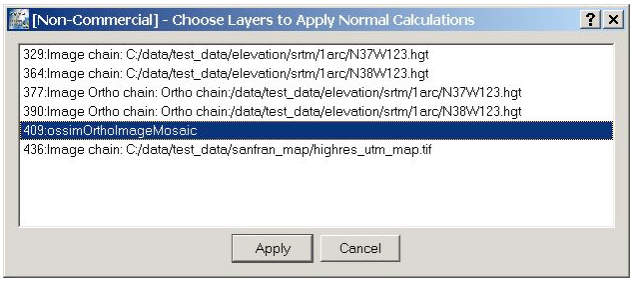


Figure 74 SRTM Normal calculations

Select the ortho mosaic chain and hit the **apply** button. Now Select **Layer->Elevation->Hillshade**.

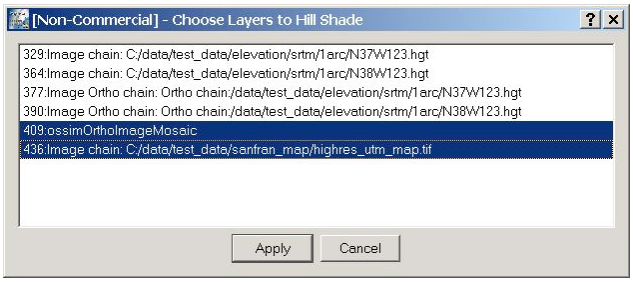


Figure 75 Choose layers for hill shading.

Resulting in the image:

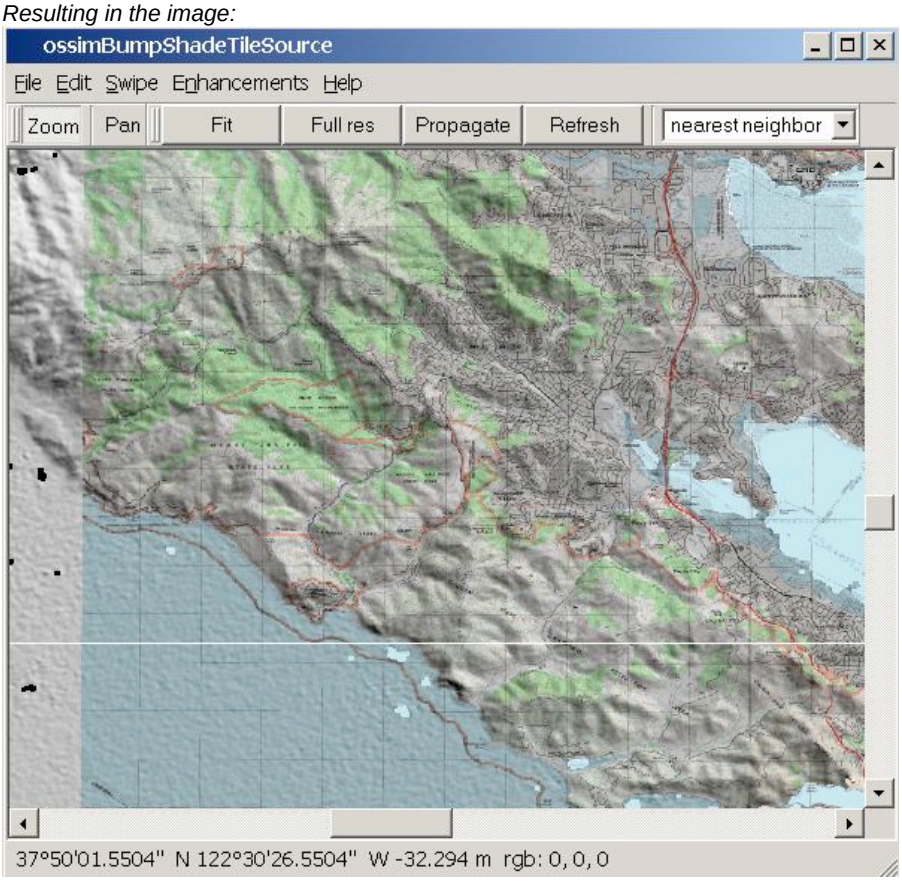


Figure 76 Hillshaded image.

The hillshade can be edited through the **Edit->Layers** menu option.

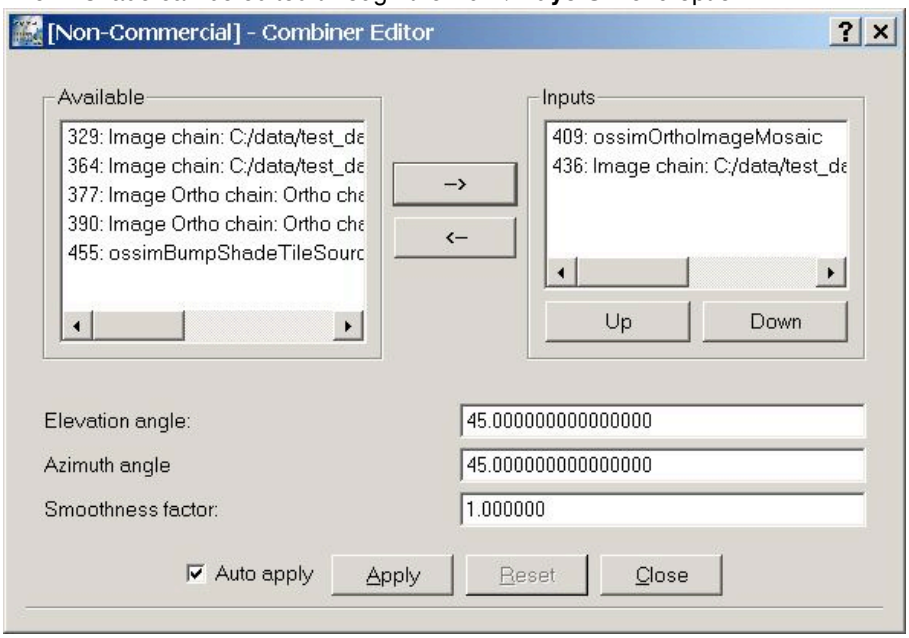


Figure 77 Adjustable parameters for the Hill Shading algorithm

Shading of the image can be adjusted by changing the sun's elevation and azimuth angles. Azimuth's 0 starts pointing in the direction up the screen so an azimuth of 0 says the sun is at the top pointing down. In this example we have the **azimuth** and **elevation** set to 45 degrees which states the sunc is coming from the direction top right and is elevated by 45 degrees above the plane. The **smoothness** factor is used to change the roughness of the scene. The smaller the value the rougher the scene and the larger the value the smoother it will look.

7.2 Vce Menu

7.3

Figure 78 Utilities menu for editing the elevation manager and unit converter.

The **V**isual **C**hain **E**ditor (**VCE**) is a start of allowing one to build your own chain flows. The **New chain** menu option will bring up a chain editor window. The **Components** menu option displays some of the OSSIM image processing factory contents to allow for visual drag and drop of processing algorithms onto the Canvas window brought up by the **New chain** menu option.

7.3.1 VCE Components

The **Vce->Components** menu option will bring up a dialog box that allows for drag and dropping of any processing filter onto the Chain editor canvas. The initial dialog comes up with the loaders tab selected.

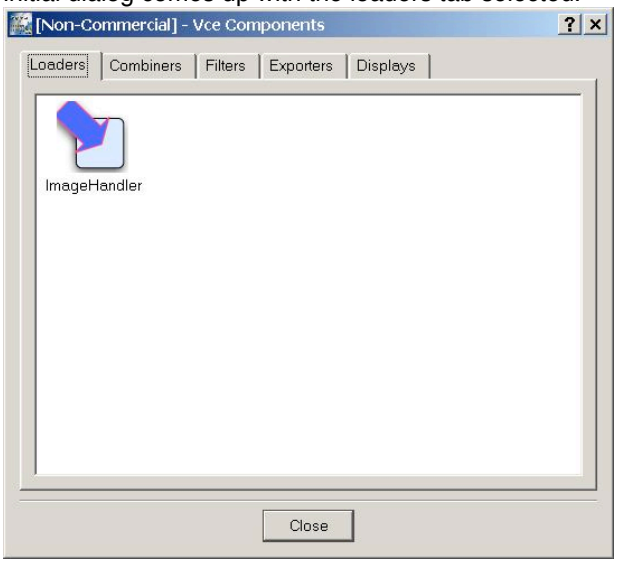


Figure 79 Loader for the VCE chain editor.

Each tab has a list of objects that can be dragged onto the canvas. Each tab has a list of objects that can be dragged onto the canvas window. Note: currently the **exporters** tab should not be used since the VCE does not support execution of the exporters. Instead of using the exporters tab selection to output products we should use the **Displays** tab to bring up a display and then do a **File->Save as**. See section 4.7.2 For using the component window in the Canvas window and connecting individual filters together.

7.3.2 Chain Editor Window

The chain editor Window is a visual canvas for building process flow from the ground up. Currently, the chain editor is not directly linked o the global data manager but does support dragging and dropping individual chains from the **Layer->Manager** window to the VCE canvas window.

From the main window hit **Vce->New chain** menu option and stretch the canvas out to give a little more workspace.

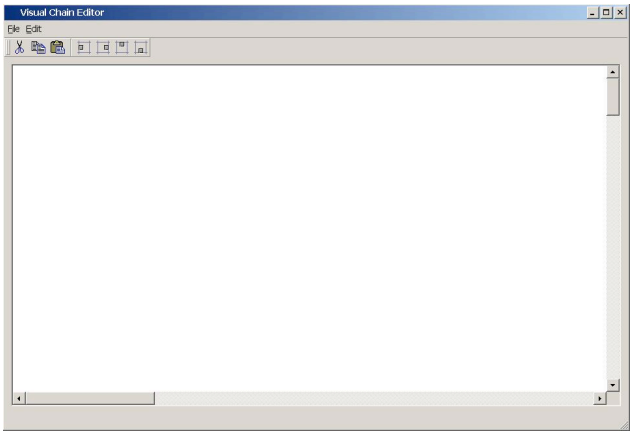


Figure 80 VCE Canvas window from Vce->New chain menu option.

7.3.2.1 Building A Chain From Scratch

In this example we build a re-projection chain from scratch. At the minimum we will need an image loader and an ossimImageRenderer filter from **Vce->Components** dialog. First bring up the components dialog and select the loaders tab.

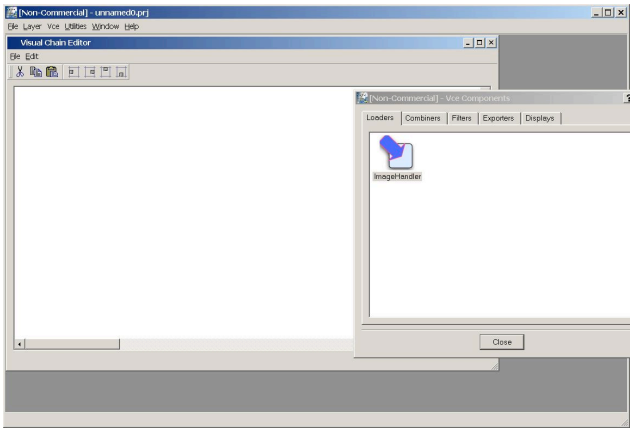


Figure 81 VCE component window with the loaders tab selected.

Select the ImageHandler component with the left mouse button. While holding down the left mouse button, drag the ImageHandler icon onto the canvas. This will immediately cause a file browser, native to your platform, to open.

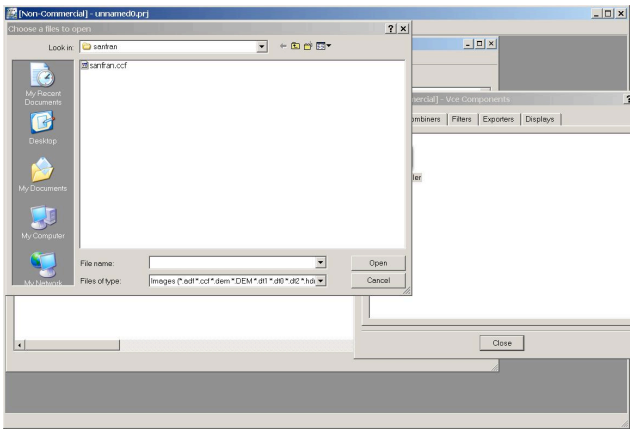


Figure 82 Drag the ImageHandler icon onto the canvas and the file browser opens automatically.

Select the image under test_data/sanfran/sanfran.ccf

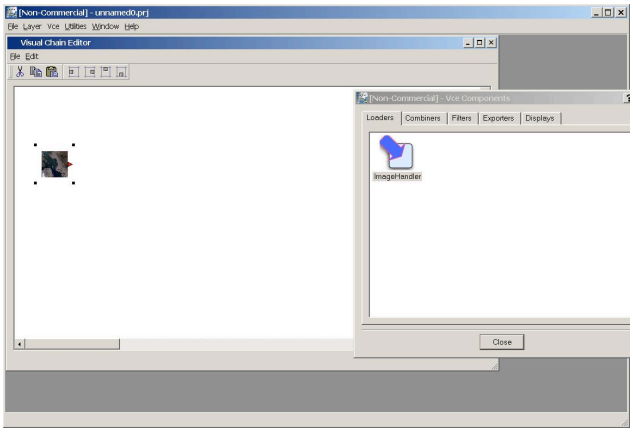


Figure 83 sanfran.ccf file loaded manually in VCE.

Now change the components dialog to the **Filters** tab.

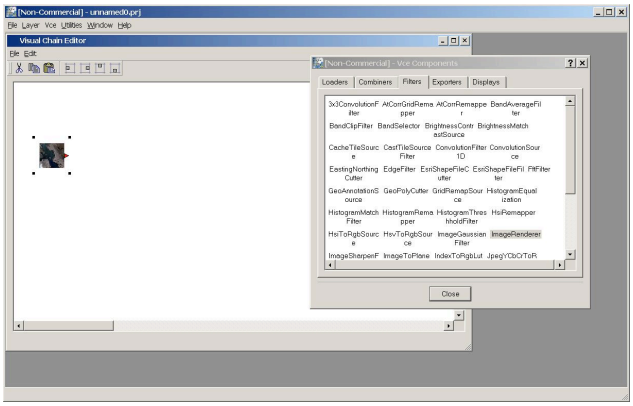


Figure 84 Filters tab for the VCE components dialog.

Select the ImageRenderer component with the left mouse button. While holding down the left mouse button, drag the ImageRenderer icon onto the canvas. Now connect the top red triangle on the ImageHandler icon to the left red triangle on the ImageRenderer filter.

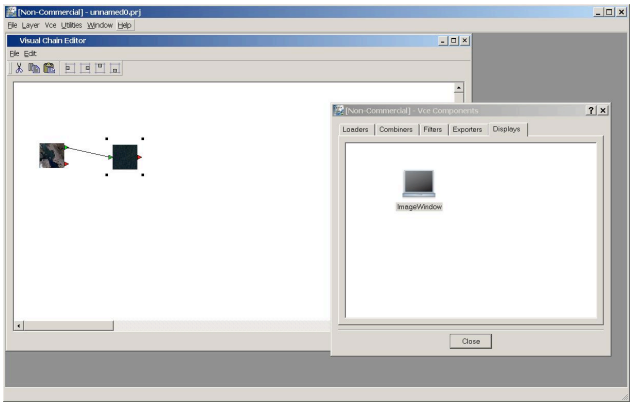


Figure 85 Connected renderer.

Now select the displays tab and drag a display onto the canvas.

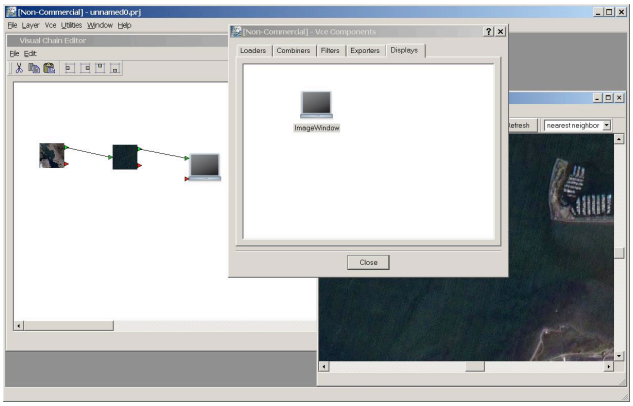


Figure 86 Final reprojection chain

The view on the ossimImageRenderer can be modified through the Display window **Edit->View** (See 4.2.5).

7.3.2.2 Building Chains from the Layer Manager

The VCE canvas supports drag-and-drop of selected chains from the **Layer->Manager** dialog onto the VCE canvas. So we have the same data loaded as the tutorial, clear the data manager of all information or just re-start the ImageLinker application. Once the data manager is cleared load images sanfran/sanfran.ccf and sanfran_map/highres_utm_map.tif under the test_data sub-directory. Once they are loaded close both displays. The data is still in the

global Data Manager even though the displays have been closed. Next, open the **Layer->Manager** and select both images and drag with the mouse cursor the chains onto the VCE canvas editor.

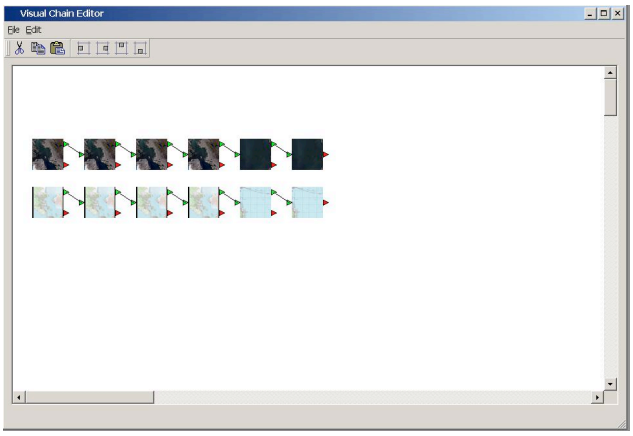


Figure 87 Chains dragged from the "Layer->Manager" window.

Select the Displays tab on the **Vce->Components** dialog and select and drag-and-drop the display onto the canvas.

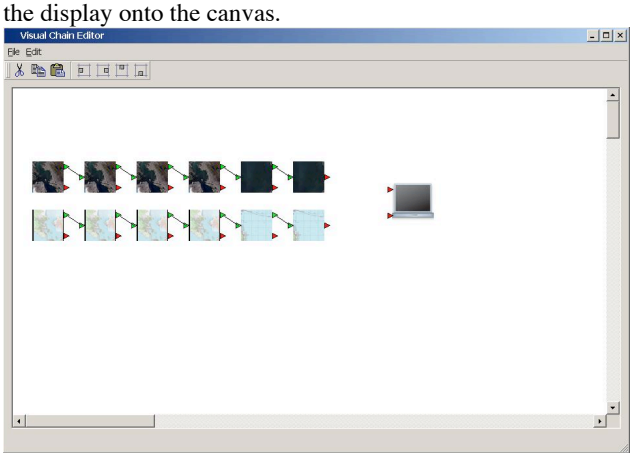


Figure 88 Display added to the canvas.

Now hold the left mouse button down on the red triangle of the right most filter in the top chain and drag the mouse over the top left red triangle on the display icon. A display window should appear in the Main window and both arrows should go to green showing that they are linked.

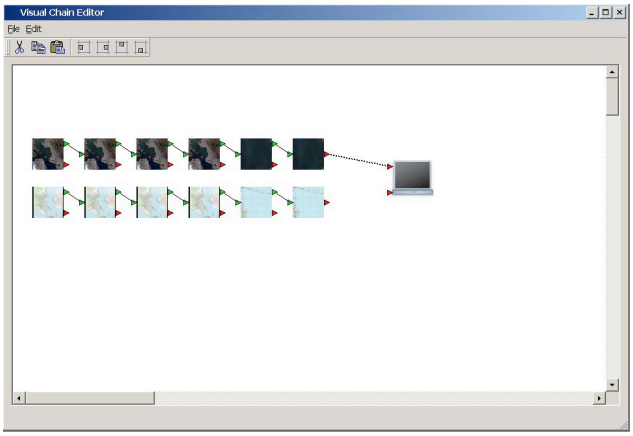


Figure 89 Display connected to the chain.

Double clicking on any filter in the chain will either bring up a property list editor for each editable property that has been exposed by OSSIM or a custom dialog editor. The property list editor is a fallback editor if no custom editor has been written. Double click on the left most filter of the top chain.

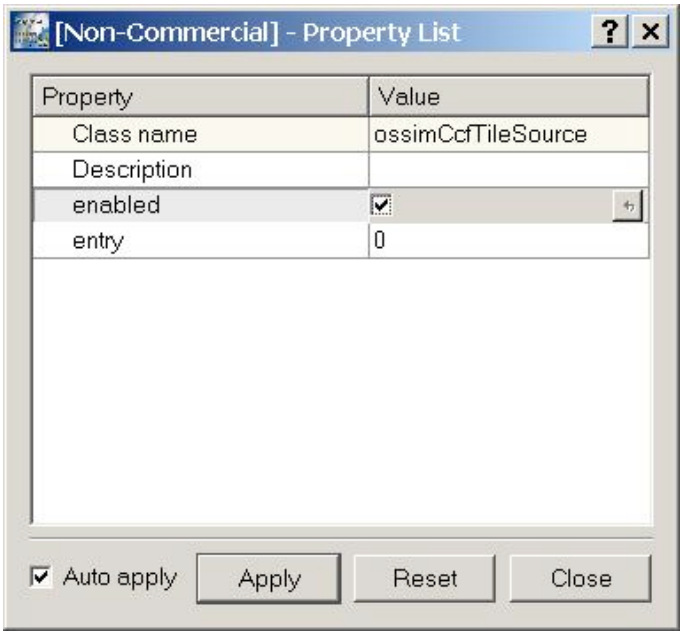


Figure 90 Property editor for the image handler.

Try toggling the enabled button off and on. When it is disabled, data does not flow and a NULL/empty image is sent through the chain indicated by the Black output.

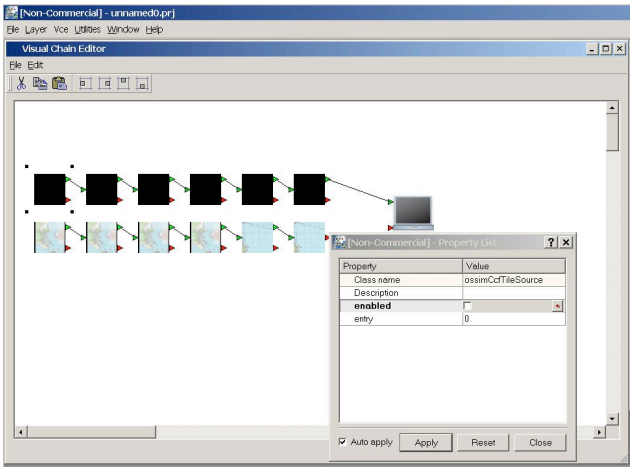


Figure 91 Toggling of the Image Handler's "enabled" property

Filters can be added to the chain by drag-and-dropping a filter from the Components dialog's filters tab onto the canvas. Drag an EdgeFilter to the canvas.

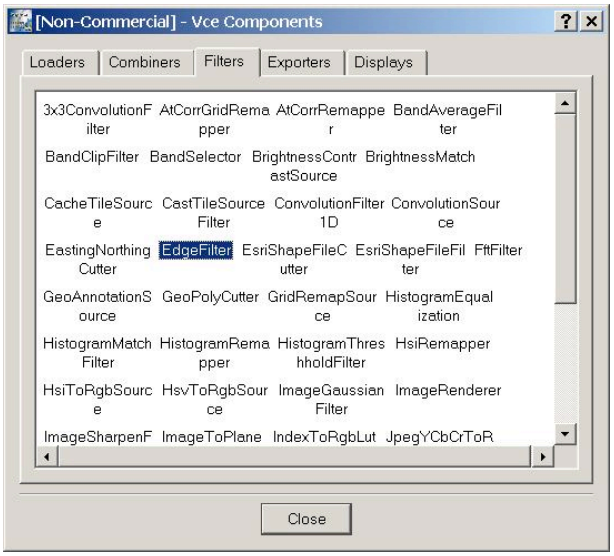


Figure 92 EdgeFilter under the Filters tab.

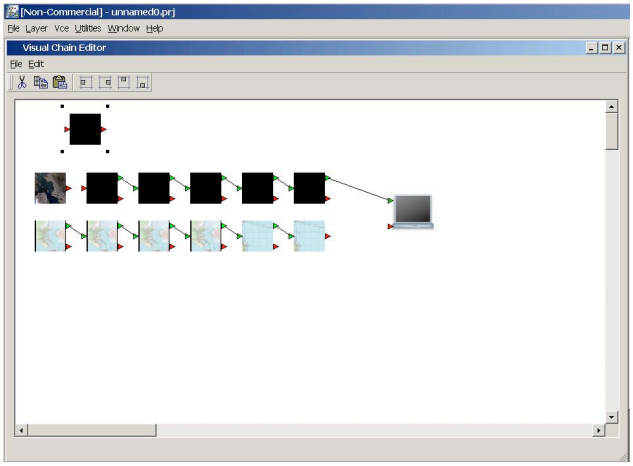


Figure 93 EdgeFilter dragged from the component/filters window to the canvas.

Now cut the link from the left most filter and the second filter and connect them to the new edge filter.

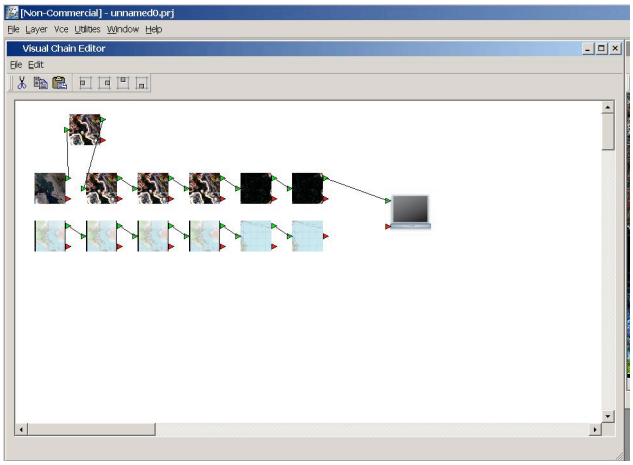


Figure 94 Final edge filter insertion into the chain.

Bring up the property editor by double clicking on the filter. Change the **Edge type** option to see different edge detectors. Initially it holds the value of Sobel. Select the field and it will have a drop down combo box for selecting other edge detection options.

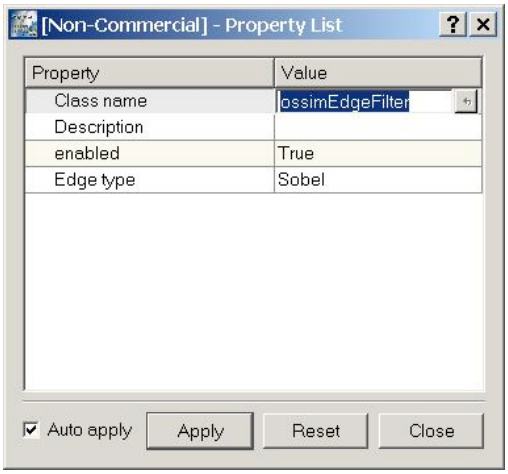


Figure 95 EdgeFilter properties.

7.4 Utilities Menu

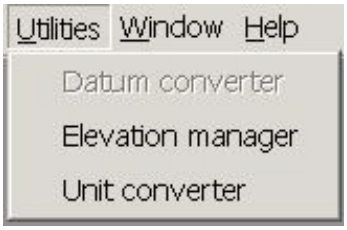


Figure 96 Utilities menu for editing the elevation manager and unit converter.

Utilities menu gives quick access to some utility dialogs. Currently we have utilities for manipulating the elevation manager and doing unit conversions.

7.4.1 Elevation Manager

The elevation manager can be accessed through the **Utilities->Elevation manager** menu option. By default, any elevation directories registered with the system is auto loaded. Any current elevation cell will appear in the list.

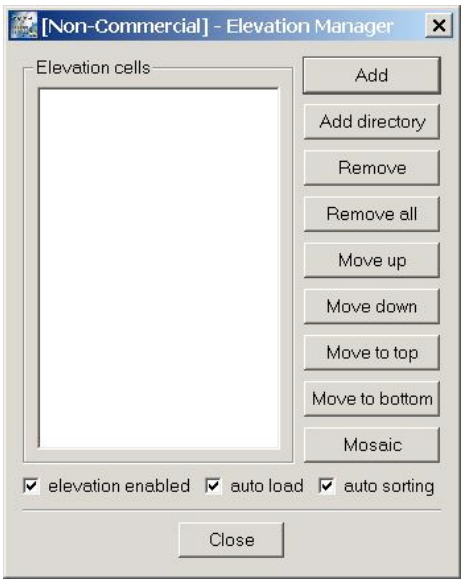


Figure 97 Elevation manager dialog interface.

Add button will allow the user to add individual elevation cells. Currently, only SRTM and DTED formatted cells are loadable.

Add director button will find all cells in the specified directory and add them to the elevation manager.

Remove button will remove the currently selected cell from the elevation manager.

Remove all button will remove all elevation cells.

Move up button will move the currently selected cell up one location.

Move down button will move the currently selected cell down on location.

Move to top button will move the currently selected cell to the top of the list.

Move to bottom button will move the selected cell to the bottom of the list.

Mosaic button will mosaic all selected cells together and add it to the data manager. Note, no overviews are created in this process. If your cells don't have overviews and you want to be able to zoom efficiently then you will need to create overviews externally from ImageLinker with the command line application call img2rr.

elevation enabled checkbox enables/disables the elevation lookup values in the elevation manager.

auto load checkbox will enable/disable automatic loading of cells. If a height for a lat lon position is not found in the current cells then it will find the cell in the manager and load it. If the autoloading is disabled (i.e. not checked) then it will not automatically try to find the cell to load.

auto sorting button will sort the loaded cells based on accuracy.

7.4.2 Unit Converter

The Unit Conversion tool is accessed through the **Utilities->Unit converter** menu option.

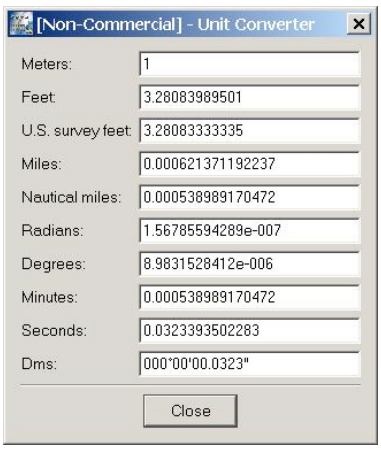


Figure 98 Unit converter dialog interface.

To use the converter type a value in any of the fields above and then tab out of the field.

7.5 Window Menu

Figure 99 Window management menu.

The window menu allows for managing the layouts and group closing of all displays in the project.

Window->Cascade will cascade all open or minimized displays.

Window->Tile will grid the and fit all displays onto the main window.

Window->Minimize all menu option minimizes all shown displays.

Window->Restore all menu will restore all minimized windows.

Window->Maximize menu option causes the current display to be maximized.

Window->Refresh all menu option causes all open displays to be refreshed.

Window->Close all Closes all open displays. Note: The data is still loaded in OSSIM and can be accessed in the **Layer->Manager** menu option. To delete a layer from memory this can only occur in the layer manager.

8 osgPlanet

see the separate tutorial on osgPlanet. osgPlanet is a 3D planetary viewer that is built on top of ossim and OpenSceneGraph.

9 OSSIM Internals

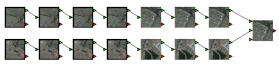
Open **S**ource **S**oftware **I**mage **M**ap (OSSIM) is a cross platform, open source software distribution of remote sensing, image processing, and geo-spatial mapping technologies.

The OSSIM distribution is centered around a C++ object oriented software library. Key functionality is implemented in the library for use in command line tools, GUI applications, and web based services. The software distribution includes utility programs, image viewers, and sophisticated custom production applications. As improvements are made to the core library all applications and services that use the library benefit.

OSSIM can load and process a wide range of geo-spatial and image formats. It supports rigorous sensor models, map projected products, and Residual Polynomial Coefficient (RPC) mechanisms. Most government and commercial formats are supported by OSSIM.

When geospatial data is loaded the associated meta-data is typically processed to correctly map project and provide precision terrain correction over elevation data where appropriate. OSSIM understands and handles map projection and datum transformations and resampling to arbitrary resolutions.

9.1 Dynamic Image Chains



Basic to OSSIM is the support of Dynamic Image Chains. The user can dynamically connect loaders, combiners, filters, and outputs within a running program. This building block approach allows complex image processing flows to be interactively constructed and modified. Each object or image unit in the chain may have its own controls and adjustable parameters. The entire state of the chain, including adjusted parameters, can be easily saved and retrieved for OSSIM enabled programs.

9.2 File formats supported

Name	Reading	Writing	Georeferencing
Tiff Tiled	Y	Y	Y
Tiff Tiled Band Separate	Y	Y	Y
Tiff strip	Y	Y	Y
Tiff Strip Band Separate	Y	Y	Y
Tiff compressed	Y	Y	Y
Tiff index	Y	Y	Y
NITF 2.0	Y	N	Y
NITF 2.1	Y	Y	Y
GeneralRaster - Band Interleave by pixel (BIP) - Band Interleave by line (BIL) - Band Sequential (BSQ)	Y	Y	Y
Jpeg	Y	Y	Y
Landsat fast format	Y	N	Y
ADRG	Y	N	Y
Esri General Raster	Y	P	Y
DOQQ V1	Y	N	Y
DOQQ V2	Y	N	Y
DTED All levels	Y	N	Y
SRTM	Y	Y	Y
usgs_dem	Y	N	Y
CIB	Y	N	Y
CADRG	Y	N	Y
CCF	Y	N	Y
Jpeg2000	P	P	P
MrSid	P	P	P
Imagine (HFA)	P	P	P
Arc Info Ascii Grid	P	P	P

In the table **P** represents the GDAL **P**lugin. For file formats that don't support geo encoding an external geometry is supported. Jpeg format and the general raster formats are examples of a supported external georeference. GDAL has a host of other exporting capabilities and a complete list can be found at www.gdal.org. The ones listed in the table are the only ones we have tested.

9.3 Projections and Transformations Supported

OSSIM supports RPC, Landsat7 and Landsat5, Spot5, Applanix ECEF and UTM orientations, and Coarse grid models and a host of map projections and state plane codes and datum shifting transformations. Much of the Map projections and datum shifting code was brought over from Geotrans. Geotrans is a NGA certified geospatial coordinate transformation library.

Here is a listing of all sensor based ossimProjection classes:

```
ossimApplanixEcefModel
ossimApplanixUtmModel
ossimCoarseGridModel
ossimRpcModel
ossimQuickbirdRpcModel
ossimNitfRpcModel
ossimLandSatModel
ossimNitfMapModel
ossimFcsiModel
ossimRpcProjection
ossimSpot5Model
```

Note: the ossimRpcProjection is a simplified version of the ossimRpcModel. Here is a listing of all Map projection based models:

```
ossimAlbersProjection
ossimAzimEquDistProjection
ossimBonneProjection
ossimBngProjection
ossimCassiniProjection
ossimCylEquAreaProjection
ossimEquDistCylProjection
ossimEckert4Projection
ossimEckert6Projection
ossimGnomonicProjection
ossimLambertConformalConicProjection
ossimLlxyProjection
ossimMercatorProjection
ossimMillerProjection
ossimMollweidProjection
ossimNewZealandMapGridProjection
ossimObliqueMercatorProjection
ossimOrthoGraphicProjection
ossimPolarStereoProjection
```

```
ossimPolyconicProjection
ossimSinusoidalProjection
ossimStereographicProjection
ossimTransCylEquAreaProjection
ossimTransMercatorProjection
ossimUpsProjection
ossimUtmProjection
ossimVanDerGrintenProjection
ossimSpaceObliqueMercatorProjection
```

Here is a listing of some miscellaneous projections:

```
ossimQuadProjection
ossimWarpProjection
ossimAffineProjection
ossimBilinearProjection
```

9.4 LAM/MPI

Local **A**rea **M**ulticomputer/**M**essage **P**assing **I**nterface or LAM/MPI is one implementation of the MPI standard. The location of LAM/MPI's main web site at the time of this document is at www.lam-mpi.org. This is a robust MPI implementation and is currently tested under OSSIM v1.6.4 within the UNIX flavor environments. NOTE: The Windows version does not come with LAM/MPI support pre-built into the OSSIM libraries.

Orthoigen, igen, and pixelflip have LAM/MPI support and have achieved near linear speedups. The speedups become apparent when the processing time increases and requires a process flow that is more complex than just an equivalent image copy. Adding reprojections and other area analysis filters would benefit from a LAM/MPI environment.

Future releases of OSSIM will be migrated to support LAM's new implementation called OpenMPI. This should be no more than a recompile of the OSSIM core libraries to the new OpenMPI libraries.

10 Command-line Applications

All command line applications have the following common options:

Options	Description
-h or --help	Should display some help for the command line application. It will display help for the common options and any additional options required by the command line application being ran.

--disable-elev	This disables elevation lookups. It takes no arguments. If you are manipulating already map projected data it is more efficient to disable elevation overhead. This can give you a substantial increase in performance depending on how you have your elevation setup. Sample Use: <executable> --disable-elev
--disable-notify	This disables a specified notification. ALL, WARN, NOTICE, INFO, FATAL, DEBUG. Sample Use: <executable> --disable-notify ALL
--disable-plugin	This option will disable loading of all plugins. Sample Usage: <executable> --disable-plugin
-K	Specify an individual keyword name=value pair to be passed and added to the preferences. It takes one string argument Sample Use: <executable> -K "name=value"
-P	Load a specific preference file. Takes one argument that is the full path to the preference file to load.
-T	Specify the classes to trace. Takes one argument that is a quoted string in a regular expression format. The string indicates which classes to output debug messages for. Sample Use: <executable> -T "ossim.*" Will trace all classes that start with the word "ossim". <executable –T "ossimElev.*|ossimImage.*" Will trace all classes that start with ossimElev and ossimImage.
--ossim-logfile	Takes a log file as an argument. All output messages will be appended to the given log file name. Sample Use: <executable> --ossim-logfile /mypath/logfile.txt

Important to note that –disable-notify and –K can appear more than once on the command line. For example: <executable> --disable-notify WARN –disable-notify NOTICE will disable WARN and NOTICE messages. For –K option we can have –K "name1=value1" –K

"name2=value2" … etc for all name value pairs you wish to update/add to the global preferences.

10.1 icp

Image copy or icp can translate one image format into another. If the image is map projected it will try to carry over the geometry information into the output file. Typing icp <return> will display the help screen followed by a list of supported output types. If the gdal plugin is enabled then you will see a number of gdal supported output types prefixed with gdal_. At the minimum you must specify and output format, and input file and an output file.

Usage: icp [options] <output_format> <input_file> <output_file>

Options	Description
-a or –-use-scalar-remapper	Will use a scalar remapper to transform to an 8-bit image
-b or –-bands	Takes a quoted comma separated list of bands to select. For example: --bands "1,2,4" Will create a 3 band output, where 1 is the first, followed by 2 as the second and followed by 4 as the third.
-c or –-compression-type	Takes an argument that describes the compression type. Currently this argument is only valid for tiff output types. The argument value can be jpeg, packbits, deflate or zip. Example Use: -c jpeg or --compression-type jpeg
--entry	Takes an argument that specifies which entry to extract. Tis argument should only be used for multi entry input sources. Sample Use: --entry 0

-l or –start-line	You can specify an offset in the line direction that specifies where to start the extraction. By default this is 0. Sample Use: --start-line 100 Will start at line 100
-L or –end-line	You can specify an offset in the line direction on where to stop the extraction. By default this is max lines Sample Use: --end-line 100 Will stop at line 100
-s or –start-sample	You can specify the offset in the sample direction on where to start the extraction. By default this is 0 Sample use: --start-sample 100 Will start at sample 100
-p or –end-sample	You can specify the offset in the sample direction on where to stop the extraction. By default this is max sample. Sample use: --end-sample 100 Will stop at sample 100
--create-overview	Specifies to create an overview for the output file being generated.
-q or --compression-quality	Specifies the compression quality. This is currently only valid for jpeg output type
-r or --res-level	Specifies the resolution level to extract from the input source. By default this is R0 or full resolution. Resolutions are numbered from 0 to (Max-1).
-t or –-create-thumbnail	Takes an argument specifying the dimension of the thumbnail. Note: This just grabs the closest r-level that fits that dimension. Sample Use: --create-thumbnail 512 Finds the closest r-level with closest dimensions to 512.
-w or –-tile-width	Defines the tile width for the handlers that support tiled output. For example, tiff_tiled_band_separate.

Examples:

1. icp jpeg foo.tif myfile.jpg
   Will copy foo.tif to myfile.jpg where the output type is specified as jpeg
2. icp –bands "1,2" tiff_tiled foo.tif output.tif
   Will copy bands 1 and 2 from foo.tif and put them into output.tif

At the time of this document the standard OSSIM formats supported are:

tiff_strip
tiff_strip_band_separate
tiff_tiled
tiff_tiled_band_separate
jpeg
general_raster_bip
general_raster_bil
general_raster_bsq
general_raster_bip_envi
general_raster_bil_envi
general_raster_bsq_envi
nitf_block_band_separate
nitf_block_band_sequential

If the GDAL plugin is supported it will depend on the GDAL build on what output formats are enabled. For my machine I have:

gdal_VRT
gdal_GTiff
gdal_NITF
gdal_HFA
gdal_ELAS
gdal_AAIGrid
gdal_DTED
gdal_PNG
gdal_JPEG
gdal_MEM
gdal_GIF
gdal_XPM
gdal_BMP

gdal_PCIDSK
gdal_PCRaster
gdal_ILWIS
gdal_PNM
gdal_ENVI
gdal_EHdr
gdal_PAux
gdal_MFF
gdal_MFF2
gdal_BT
gdal_IDA
gdal_FIT
gdal_RMF
gdal_USGSDEM

The plugin output types are all prefixed with the gdal_.

10.2 orthoigen

orthoigen application is a powerful product generation tool. Orthoigen generates an igen spec internally and calls the same code the igen application uses. Command line options are explained followed by before and after screen shot samples.

Usage: orthoigen [options] <input files> <output_file>

Note: The last file on the command line is assumed to be the output filename. The output writer type is determined by that extension unless a writer-template is given.

orthoigen is also an MPI aware applications. If MPI support was included during the build stage then you can use MPI for the processing. For example: If you have LAM/MPI and OSSIM was compiled with MPI turned on then you could execute:

mpirun –np <number_processors> orthoigen [options] <input files> <output_file>

and would use <number_processors> to perform the operation.

Options	Description
--annotate	Takes an annotation keywordlist.
--chain-template	Specify a chain template for each image to process
--combiner-type	Specify the type of combiner to use. The type can be ossim class names. For example: To get a full listing run the factory_dump with factory_dump –t ossimImageCombiner type. Here is a list of combiners tested: ossimBlendMosaic ossimMaxMosaic ossimImageMosaic ossimClosestToCenterCombiner ossimBandMergeSource ossimFeatherMosaic
--cut-center-ll	Takes two arguments in degrees specify the latitude followed by the longitude of the cut center. For example: --cut-center-ll 28.5 –118 will specify the cut center at location <28.5, -118>.
--cut-radius-meters	Specify the cut distance in meters. This will take the center cut point and produce a bounding box surrounding the center point with the specified distance.
--enable-entry-decoding	A filename can be separated by a | and a number (NO space). Example: - ./a.toc|0 will do entry 0 of a.toc file and on unix you might want to use a \ since the separator is a pipe sign. Example: ./a.toc|0 will do entry 0 of a.toc
--geo	Will set the view to a geographic projection with origin at 0,0. The GSD = to the input.
--geo-scaled	Takes a latitude as an argument for purpose of scaling. Specifies that no spec file was defined. Defaults to a scaled geographic image chain with GSD = to the input.
--utm	Specifies a utm projection.
--hist-stretch	Specify in normalized percent the low clip and then the high clip value. Note: You must have ran create_histo on all images before applying this operation.
--input-proj	The output projection will default to the input projections.

--meters	Change the GSD to the passed in meters per pixel. So if you want 50 meters per pixel GSD then make this –meters 50.
--resample-type	Specify what type of resampling you would like. Values can be: - nearest - bilinear - cubic - gaussian - hanning - hamming - lanczos - mitchell - catrom - blackman - sinc - qessel - quadratic - hermite - bspline
--slave-buffers	Specifies number of tiles to buffer when doing MPI applications. When passing the tile from the slave node back to master we can continue processing the next tile if the buffers is larger than 1. For performance you can play with this value. If not present then by default it will be 2 buffers per node.
--tiling-template	Specify an external file that contains tiling information. A template can be passed to orthoigen that enables tiling support. Tiling can be done in pixels, meters, decimal degrees. For meters and decimal degrees tiling it will depend on the view projection. If you are geographic then use decimal degrees else use meters.
--view-template	Is an OSSIM keywordlist the describes the view projection and parameters.
--writer-template	By default a writer is determined based on the extension of the output file. A writer keywordlist can be used a template to orthoigen if you need more settings than the default action taken based on the file extension. **Note: this didn't make it into 1.6.4 release but is available from current CVS.**
-t or –thumbnail	Will manipulate the view so the output product corresponds to the thumbnail dimension. for example: --thumbnail 512 will produce a 512x512 (close to it) thumbnail image of the product being generated.

For all examples we will use the following overlapping input images.

Figure 100 This will correspond to filename tile1.jpg and is a 256x256 UTM input file

Figure 101 This will correspond to filename tile2.jpg and is a 256x256 UTM image. This has been brightened to show visual differences in the algorithms.

One of the images has been brightened to show visual differences of certain operations. These images are also located under the test_data/orthoigen directory on the test_data distribution.

10.2.1Reprojection

A view must be specified for the orthoigen application to execute properly. First operation will reproject the UTM tile1.jpg image to a geographic output tif

```
orthoigen --geo tile1.jpg output.tif
```

Figure 102 Geographic projected output. Tile1.jpg is a UTM 5 meter product and is reprojected to a 5-meter geographic projection. See the difference between Figure 100.

Two other quick options are –-utm and –-input-proj. The –-utm will reproject to utm and the –-input-proj will default the output view to the input images projections. For more complex view definitions you can use the view template.

View templates are generally reserved for scripting where other applications drive the execution of the orthoigen application. A detailed keyword list for the view template is beyond the scope of this document. The easiest way to generate a keywordlist for the view is to use the imagelinker GUI and save out an igen spec file and cut the product view template out of the spec file and save it to a template file that can be used in orthoigen. For example, I opened up tile1.jpg into imagelinker and changed to a California state plane projection using the edit view. Next, do a file/save as and do a spec file only save of the image. Open the spec file and cut and paste the product information into a file called view.kwl. You should get something that looks like the following:

```
product.projection.central_meridian:  -119.000000000000000
product.projection.datum:  NAR-C
product.projection.ellipse_code:  RF
product.projection.ellipse_name:  GRS 80
product.projection.false_easting:  2000000.000000000000000
product.projection.false_northing:  500000.000000000000000
product.projection.major_axis:  6378137.000000000000000
product.projection.meters_per_pixel_x:  5.000000000000000
product.projection.meters_per_pixel_y:  5.000000000000000
```

```
product.projection.minor_axis:  6356752.314100000075996
product.projection.origin_latitude:  35.333333333333336
product.projection.pcs_code:  26944
product.projection.std_parallel_1:  36.000000000000000
product.projection.std_parallel_2:  37.250000000000000
product.projection.tie_point_easting:  2000000.000000000000000
product.projection.tie_point_northing:  500000.000000000000000
product.projection.type:  ossimLambertConformalConicProjection
```

Cutting the view out and paste it into a file called view.kwl. Now run the orthoigen with the view template:

```
orthoigen –-view-template view.kwl input/tile1.jpg lambert.jpg
```

Produces an image called lambert.jpg

Figure 103 Using the view template we produced a lambert projected image. See the difference between Figure 100.

10.2.2Mosaicing

Orthoigen can be used for mosaicing images together. Note the layer ordering is as they appear on the command line. The first image is the top layer and the last image is the bottom layer. Running the following orthoigen will do a default A over B style mosaic.

orthoigen –t 256 -–utm tile1.jpg tile2.jpg output.tif

Figure 104 utm projected, 256x256 thumbnail.

Another variation that allows one to change the mosaic type is to specify another combiner on the command line using the —-combiner-type options.

```
orthoigen –-combiner-type ossimClosestToCenterCombiner –t 256 -–utm
tile1.jpg tile2.jpg output.tif

orthoigen --combiner-type ossimFeatherMosaic –t 256 -–utm tile1.jpg
tile2.jpg output.tif
```

Figure 105 utm projected, 256x256 thumbnail doing a closest to center combiner (left). A feather combiner (right).

There are several other combiner types not shown:

ossimBlendMosaic Blends all layers into a single average value. Areas that have only one image are just copied.

ossimMaxMosaic Takes the maximum value of all layers.

ossimColorNormalizedFusion Takes 2 input images where one is a color image and the other is a pan/grey scale image. Creates a pan sharpened product.

ossimLocalCorrelationFusion Takes 2 inputs where one is a color image and the other is a pan/grey scale image. It's an adaptive fusion that uses the local correlation between the two images to sharpen the color image. **Note:** This currently should only be used in the Imagelinker GUI since I do not automatically determine the scale difference between the pan/grey scale image and color image. These parameters will need to be hand tweaked in the editor.

ossimSFIMFusion Acronym stands for **S**moothing **F**ilter-based **I**ntesity **M**odulation and takes 2 inputs where one is a color image and the other is a pan/grey scale image. **Note:** This currently should only be used in the Imagelinker GUI since I do not automatically determine the scale difference between the pan/grey scale image and color image. These parameters will need to be hand tweaked in the editor. This algorithm appears to work the best at preserving spectral information.

ossimBandMergeSource Takes all input files and merges the bands together and produces a merged output. So if you have 3 one band input images then you will have a single 3 band output image.

Since OSSIM is factory driven, for a complete list you can execute the application

factory_dump –t ossimImageCombiner

and it will list all ossimImageCombiner type objects.

10.2.3Histogram

Histogram operations can be applied to any image that has histograms built. Histograms can be built by using the create_histo application. Currently we have only exposed a linear stretch given a penetration percent. The clip percents are normalized between 0 and 1.

```
orthoigen --hist-stretch  .1 .1 --utm  tile1.jpg histogram.tif
```

Penetration is given as the first number corresponding to the left side and the second number cooresponding to the right side.

Figure 106 Histogram stretched utm image. 10% (.1) penetration on both the left and right side.

10.2.4Cut and Resample

The filter kernel can be changed by setting the filter type using the —resample-type. We will zoom tile1.jpg to .5 meter. The original is a 5 meter image and then cut it by 50 meters. Here we show 3 command line executions with nearest, cubic and sinc respectively.

```
orthoigen --resample-type nearest --utm --cut-center-ll 37.92626
-122.37761 --cut-radius-meters 50 --meters .5 input/tile1.jpg
nerest.jpg

orthoigen --resample-type cubic --utm --cut-center-ll 37.92626
-122.37761 --cut-radius-meters 50 --meters .5 input/tile1.jpg cubic.jpg

orthoigen --resample-type sinc --utm --cut-center-ll 37.92626
-122.37761 --cut-radius-meters 50 --meters .5 input/tile1.jpg sinc.jpg
```

Figure 107 Shows a cut and resample differences. These are tiles cut at lat 37.92626 and lon –122.37761 with cut radius of 50 meters and gsd resolution of .5 meter. The original was a 5 meter image giving a 10x zoom. Left, is nearest neighbor resampling, middle is cubic and the right side is sinc. See options table for other resampler types

10.2.5Tiling

Orthoigen supports geographic, meters, and pixel tiling options. The tiling specification is currently through an external keyword list and is loaded using the –-tiling-template command option. The format of the template is:

```
igen.tiling.type: ossimTiling
igen.tiling.tiling_distance: 10000 10000
igen.tiling.tiling_distance_type: meters
igen.tiling.delta: 100 100
igen.tiling.delta_type: per_pixel
igen.tiling.tile_name_mask: tile_%r%_%c%
igen.tiling.padding_size_in_pixels: 0 0
```

where:

i**gen.tiling.tiling_distance** specifies the unit of the distance type

igen.tiling.tiling_distance_type can be pixel, meters, degrees, minutes, or seconds

igen.tiling.delta specifies the delta distance. If delta_type is per_pixel then it is in the unit of the distance_type. If the delta_type is total pixels hen the delta is the total number of pixels within the tiling_distance.

igen.tiling.delta_type can be total_pixels or per_pixel

igen.tiling.tile_name_mask specifes a mask for the filename. **NOTE:** this is only used in igen application and not orthoigen. This mask is actually specified

on the command line as the output image if –-tiling-template option was specified. The mask variables are:

%r% all occurances are replaced by the row number

%c% all occurrences are replaced by the column number

%i% all occurrences are replaced by the tile id

%or% all occurrences are replaced by the origin of the row

%oc% all occurrences are replaced by the origin of the col

%SRTM% does SRTM tiling scheme. Nice to use when building a database of SRTM elevation from some other elevation format and projection.

igen.tiling.padding_size_in_pixels Specifies a padding to use. It will extend the borders out by the specified number of pixels in the horizontal and vertical direction.

Pixel tiling is used most commonly when you want to guarantee some power of two set of images or the single image is too large and needs to be split apart. We will tile up tile1.jpg image in **Figure 100** to a 64x64 set of tiles. First define the template tiling keyword and assume the name is pixel_tiling.kwl:

```
igen.tiling.type: ossimTiling
igen.tiling.tiling_distance: 64 64
igen.tiling.tiling_distance_type: pixels
igen.tiling.delta: 1 1
igen.tiling.delta_type: per_pixel
igen.tiling.padding_size_in_pixels: 0 0
```

Now execute:

```
orthoigen --utm --tiling-template pixel_tiling.kwl input/tile1.jpg
pixel_%r%_%c%.jpg
```

NOTE: we put the file name mask as the last argument. Make sure that the name generated is unique. Supplying only a column id or just row id will generate same named files but putting both a row and column will produce a unique name.

The above command line produces a 4x4 grid of tiles with a total of sixteen tiles named pixel_0_0.jpg, pixel_0_1.jpg, pixel_0_2.jpg, … etc. Only the first row is shown:

Figure 108 These are the fist row of tiles produced from the orthoigen pixel tiling example.

The mask could also have been pixel_%i%.jpg and would have generated pixel_0.jpg, pixel_1.jpg, …,pixel_15.jpg

Let's specify the tiling in meters and put it in a file called it meter_tiling.kwl. I will create a tiling for the 5-meter UTM file in **Figure 100**.

```
igen.tiling.type: ossimTiling
igen.tiling.tiling_distance:  320 320
igen.tiling.tiling_distance_type: meters
igen.tiling.delta: 5 5
igen.tiling.delta_type: per_pixel
igen.tiling.padding_size_in_pixels: 0 0
```

This will generate a tile that is 320 meters wide by 320 meters high and will generate the tile based on a 5 meters per_pixel setting. This should give something along the lines of a 64x64 pixel tile where the gsd is 5-meters. Now execute:

```
orthoigen --utm --tiling-template meter_tiling.kwl input/tile1.jpg
meter_%r%_%c%.jpg
```

Currently the tiles are snapped to an even tiling_distance. You might get some tiles that have a sliver of valid information while the rest of it is black or null. Here is the output of the first row.

Figure 109 These are the fist row of tiles produced from the orthoigen meter tiling example.

Let's specify the tiling in seconds and put it in a file called seconds_tiling.kwl. I will create a tiling for the 5-meter UTM file in **Figure 100**. Now since we are a geographic tiled output we will need to convert the input image into a geographic projected image using the –-geo option for the output view. The tiling keywords are:

```
igen.tiling.type: ossimTiling
igen.tiling.tiling_distance:  10 10
igen.tiling.tiling_distance_type: seconds
igen.tiling.delta: .16
igen.tiling.delta_type: per_pixel
igen.tiling.padding_size_in_pixels: 0 0
```

This says to do a 10x10 seconds image where each pixel size is .16 seconds. Now execute the command line:

orthoigen --geo --tiling-template seconds_tiling.kwl input/tile1.jpg seconds_%r%_%c%.jpg

and we get tiles named seconds_0_0.jpg, seconds_0_1.jpg, … etc. Using **Figure 100** as the input we get:

Figure 110 These are the fist row of tiles produced from the orthoigen meter tiling example.

10.2.6Writer template

Writer templates are currently available with CVS builds since the version 1.6.4 release. If running from the CVS source code you can apply the writer template to specify specific output attributes and change to other writers. The easiest way to create a template is to use imagelinker to create an igen spec file much like was done for the view template. Open an image and then in the file/save as set your writer definitions and then output all your settings to the igen spec file. Edit the spec file and copy out the writer text located with object2. prefix value. Move this to a file called writer_template.kwl. Although the current orthoigen will take it as valid you really should strip the object2. prefix from the writer keywordlist. For example, I get the following from when extracting the writer from the igen spec file created from the imagelinker application:

```
object2.color_lut_flag:  0
object2.compression_quality:  75
object2.compression_type:  none
object2.create_envi_hdr:  0
object2.create_external_geometry:  0
object2.create_fgdc:  0
object2.create_histogram:  1
```

```
object2.create_image:  1
object2.create_jpeg_world_file:  0
object2.create_overview:  1
object2.create_readme:  0
object2.create_tiff_world_file:  0
object2.description:
object2.enabled:  1
object2.filename:  /data/ear1/output/docs/writer.tif
object2.id:  33
object2.image_type:  tiff_tiled_band_separate
object2.input_connection1:  9
object2.input_list_fixed:  1
object2.number_inputs:  1
object2.number_outputs:  0
object2.output_geotiff_flag:  1
object2.output_list_fixed:  0
object2.output_tile_size_x:  64
object2.output_tile_size_y:  64
object2.overview_compression_quality:  75
object2.overview_compression_type:  1
object2.pixel_type:  point
object2.projection.units:  meters
object2.type:  ossimTiffWriter
```

Now strip the object2. prefix and save in the writer_template.kwl:

```
color_lut_flag:  0
compression_quality:  75
compression_type:  none
create_envi_hdr:  0
create_external_geometry:  0
create_fgdc:  0
create_histogram:  1
create_image:  1
create_jpeg_world_file:  0
create_overview:  1
create_readme:  0
create_tiff_world_file:  0
description:
enabled:  1
filename:  /data/ear1/output/docs/writer.tif
id:  33
image_type:  tiff_tiled_band_separate
input_connection1:  9
input_list_fixed:  1
```

```
number_inputs:  1
number_outputs:  0
output_geotiff_flag:  1
output_list_fixed:  0
output_tile_size_x:  64
output_tile_size_y:  64
overview_compression_quality:  75
overview_compression_type:  1
pixel_type:  point
projection.units:  meters
type:  ossimTiffWriter
```

Now execute the template on the input image:

```
orthoigen –-utm –-writer-template writer_template.kwl tile1.jpg
output.tif
```

It will use the settings in the writer_template.kwl to instantiate the writer and output the product.

10.2 image_info

This application displays the general information about an image.

Usage: image_info [options] <input_file>

By default all information is output for the input file. If multiple entries exists then each is prefixed with image<number>., where <number> is the entry number.

Options	Description
--palette	If one exists, a color palette will be outputted.
-i	Only image information such as height, width, etc is outputted.
-o	Output the information to the specified file
-p	If the input image has any then output the projection information. Will also output corner coordinates
-s	Specify the datum you want the ground rect returned. Currently it is the datum of the projector.
-v	Overwrite output geometry file.

Examples:

1. image_info <file_name>
   This will output all information to standard out.
2. image_info –-palette <file_name>
   Will output a palette only if one exists.
3. image_info –i <file_name>
   Will output image information only
4. image_info –i –p <file_name>
   Will output image and projection information

10.3 img2rr

This is an overview builder application. It is important to build overviews for any image file to be processed at different resolutions. When executed on any image a multi resolution pyramid is created for the image as an external file with a .ovr extension. In the case of r multi entry it will ad _e<number>.ovr to each entry.

Options	Description
-a or –-include-fullres	Will copy the full res image into the overview file
-e or –-entry	If the input image is multi entry then do a specific entry.
--resample-type	Currently you can do nearest or box. Box is the default and is just a 2x2 average.
--compression_quality	Currently you can do nearest or box. Box is the default and is just a 2x2 average.
--compression-type	The compression type can be NONE, JPEG, PACKBITS, or DEFLATE. The default is none and no compression is used.

Examples:

1. img2rr foo.tif
   Will build all overviews for foo.tif and produce an output image called foo.ovr.
2. img2rr –a foo.tif
   Will copy the full resolution image into foo.ovr and also build overviews into the same file.

3. img2rr –-compression-quality 70 ––compression-type JPEG foo.tif. This will produce an overview file call foo.ovr and the different resolutions will be jpeg compressed at 70% quality.

10.4 create_histo

This application creates a histogram for the image and writes it to the sme filename but with extension .his.

```
Usage: create_histo [options] <image_file>
```

Options	Description
--entry	Takes an argument that specifies which entry to compute a min max for. Sample Use: create_histo –entry 0 foo.ntf Will do the first entry of input file and create an external histogram called foo_e0.his
--bins	Override for the number of bins. Really important for float data and higher bit depths. By default it will be 65536 number of bins if can't be determined.
--max	Override for the bin range. This sets the max value for the histogram.
--min	This sets the min value for the histogram.

Examples:

1. create_histo foo.tif
   Will create a histogram for the input image and call it foo.his.
2. create_histo –min 100 –max 1100 –bins 4000 foo.ntf
   Will create a histogram for values between 100 and 1100 and estimate it with 4000 bins.

10.5 cmm

Compute min/max pixel values from within a file. Important to run this on radiometries other than 8 bit. It will write an external OSSIM Metadata File (omd) file. So if you have an image foo.tif and you run cmm foo.tif then the file foo.omd is written/updated to the new values.

```
Usage: cmm <image_file>
```

Under unix type shells you should be able to wildcard it since they get expanded: cmm *.tif

Executing the command within a shell that supports wildcarding (like a bash shell) would compute min max for all files with a .tif extension.

This program will force a re-compute of Min/Max Pixel Values.

Options	Description
--min	Min pixel override. The min value is not computed from the image but uses this value as the min value for all bands
--max	Max pixel override. The max value is not computed from the image but uses this value as the max value for all bands
--null	Can override the null value.
-e or –entry	Takes an argument that specifies which entry to compute a min max for. Sample Use: cmm –entry 0 foo.ntf Will do the first entry of input file.
-l or –list-entries	Will list the id indices of each entry. Note: Since some handlers have multiple[le entries where some entries can't be rendered, either they aren't renderable or we just don't support it yet, you could get non sequential numbering. For example: you might have an image with 10 entries but only 3 of them are renderable and might have hypothetical id's 3, 7, 8.

Examples:

1. cmm foo.tif
   Min and max values are computed.
2. cmm –-min 1 –-max 200 foo.tif
   The file is not scanned since both min and max are overridden and will be
   Written to the foo.omd.

10.6 ogeom2ogeom

ogeom2ogeom is a geometry conversion application that can take any projection understood by OSSIM and produce a coarse grid or RPC projection. It is important to note that the application will currently overwrite the passed in geometry if it was originally an ossim geometry file with a .geom. extension.

Usage: ogeom2ogeom [options] <geometry|imagefile>

The argument to the applications can take either an image file and try to extract the geometry and convert it or an ossim geometry file. Whatever file is given it will produce that file with a changed extension to .geom. Some of the options are:

Options	Description
--cg	Create a coarse grid projection. This will estimate the input projection with a grid.
--disable-adjustments	Applies to –-cg option. This will disable the creation of adjustable parameters in the coarse grid. The coarse grid generates a layer for every adjustable parameter taking up a lot of space. This can decrease the amount of space used by the coarse grid.
--noelev	If the output is RPC then all heights are 0. For coarse grid it just turns off applying heights to the model.
--rect	Takes 4 arguments ulx uly width height. If you give the application a geometry file only, then you must specify the image coordinates for the geometry.
--rpc	Generate an rpc output model

--tolerance	Used as an error tolerance for the estimate. This is mainly used by the coarse grid for splitting to higher estimates if it doesn't fall within tolerance.

Examples:

1. ogeom2ogeom --cg <image>
   Will Try to get an input projection from the image and replace it with a coarse grid model.
2. ogeom2ogeom --cg –-noelev <image>
   Will create a coarse grid model and disable elevation. Note: Elevation is applied but after it is applied it is no longer affected by elevation.

10.7 applanix2ogeom

The application gives a quick way to generate OSSIM geometry files given the Applanix parameters. The applications can only support ECEF orientations where the platform positions are in ECEF WGS84 points and UTM Orientation with ORTHOMETRIC HEIGHTS and not ELLIPSOIDAL HEIGHTS. In CVS the Applanix model support ORTHMETRIC or ELLIPSOIDAL HEIGHTS where the ellipsoidal height must be above the WGS84 ellipsoid.

Usage: applanix2ogeom <camera_file> <exterior_orientation_file> <imageToProcess> <optional_output_directory>

Where:

- <camera_file> an ossim formatted camera file for applanix.
- <exterior_orientation_file> an Applanix file generated by their POSPAC software.
- <imageToProcess> Is an applanix image. Usually: <id>.extension. The <id> is used to look up its orientation in the exterior orientation file.
- <optional_output_directory> Output the result to an output directory.

If an <imageToProcess> is not given then all geometries from the <exterior_orientation file> will be generated with <id>.geom. as the filename.

We currently only support either UTM or Ecef specified exterior orientation file. The camera parameters must be specified in an OSSIM keywordlist. The Applanix generates a calibration report in the form of a pdf. Copy those parameters to the associated keywords. A sample OSSIM/Applanix camera file is provided:

sensor: sn0016
focal_length: 54.909
principal_point: -.032 .331

image_size: 4077.0 4092.0
pixel_size: 0.009
d0: 1 -0.02
d1: 2 -0.2
d2: 3 -0.66
d3: 4 -1.57
d4: 5 -3.05
d5: 6 -5.25
d6: 7 -8.3
d7: 8 -12.32
d8: 9 -17.44
d9: 10 -23.76
d10: 11 -31.39
d11: 12 -40.42
d12: 13 -50.93
d13: 14 -62.99
d14: 15 -76.64
d15: 16 -91.94
d16: 17 -108.89
d17: 18 -127.52
d18: 19 -147.8
d19: 20 -169.71
d20: 21 -193.19
d21: 22 -218.16
d22: 23 -244.53
d23: 24 -272.17
d24: 25 -300.94
d25: 26 -330.64
distortion_units: microns

where:

- **sensor** specifies the sensor id
- **focal_length** specifies the focal length in millimeters
- **principal_point** specified in millimeters
- **image_size** specified in pixels. Typically 4077x4092. The parameter is width followed by height separated by a space.
- **pixel_size** pixel size on the camera in millimeters
- **d0-d25** are distortion parameters. Is a tuple that is the distance from the center followed by the distortion
- **distortion_units** The units of the distortion. Usually they are given in microns. They will be converted internally to millimeters.

Examples:

1. applanix2ogeom camera_file exterior_orientation
   Will generate all geometries into the current directory with name <id>.geom.

2. applanix2ogeom camera_file exterior_orientation output_dir
   Will generate all geometries into the output_dir with name output_dir/<id>.geom.
3. applanix2ogeom camaera_file exterior_orientation imagefile
   Will generate one geometry file if the image is found in the exterior orientation file. The image is usually something of the form <id>.jpg or <id>.tif, .. etc. It first strips the extension and then uses that for the id to lookup into the exterior orientation file.
4. applanix2ogeom camaera_file exterior_orientation imagefile output_dir
   Will generate one geometry file and then output to the output_dir.

10.8 igen

igen is a powerful command line application that is entirely keywordlist driven. It takes a keywordlist as an argument and builds a product from the keywordlist specification. One way to use it is to setup a product in imagelnker and save out a spec file. This is typically done if you have a product that is very large and might take several hours or maybe a day to run. The spec file generated can be used to run the command line application igen that can be ran separate from the imagelinker GUI. For example, if you saved out the spec file from imagelinker to a file called igen.spec then you can do:

igen igen.spec

producing a product based on your setting from imagelinker. Adding

mpirun –np <np> igen igen.spec

will run the application on a cluster of machines for faster product generation. **Note:** I have not done windows builds of mpi.