INSTALLATION GUIDE

**GENERAL PREREQUISITES**

The required libs and dlls are included in the GitHub repository. The setup instructions for load\_extension is documented within setup.py if desired to re-create, otherwise the .pyd is located in “Python Extension\build\lib.win-amd64-2.7” (copy and paste it into Anaconda site-packages). Refer to this guide for a fresh install of all dependencies if needed. Be sure to have Microsoft Visual Studio installed along with the latest Python 2.7 (the Anaconda package is recommended). The neural network library being used is Lasagne alongside the Theano library. To get these packages, conda install mingw, libpython, theano, and lasagne.

**GDAL**

1. Install gdal 2.0.1, core components and the include/lib folders then run installer. I would keep these all in a single folder such as C:\GDAL. 2.0.1

Source found here:<https://trac.osgeo.org/gdal/wiki/DownloadSource>

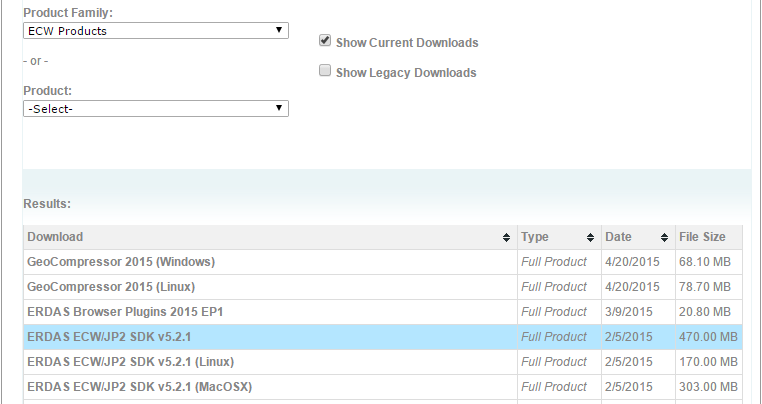
Core components and folders found here:<http://www.gisinternals.com/query.html?content=filelist&file=release-1800-x64-gdal-1-11-3-mapserver-6-4-2.zip>

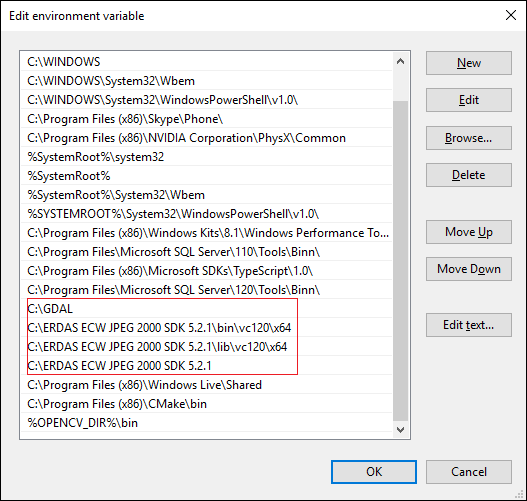
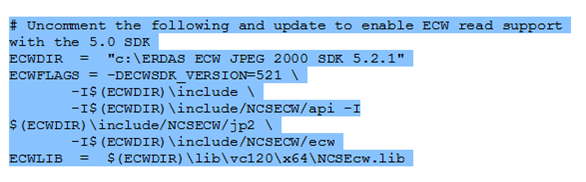
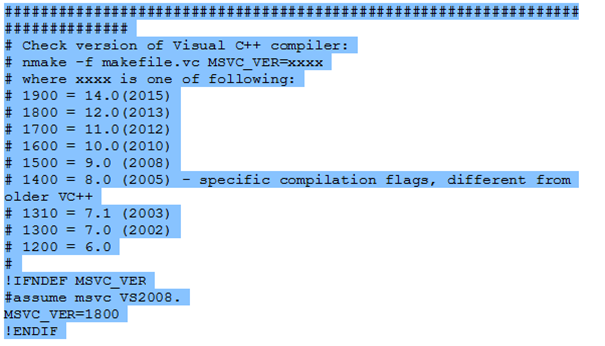
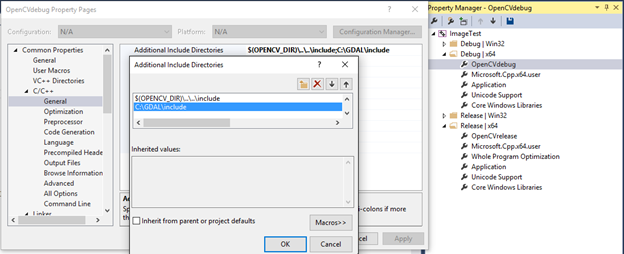
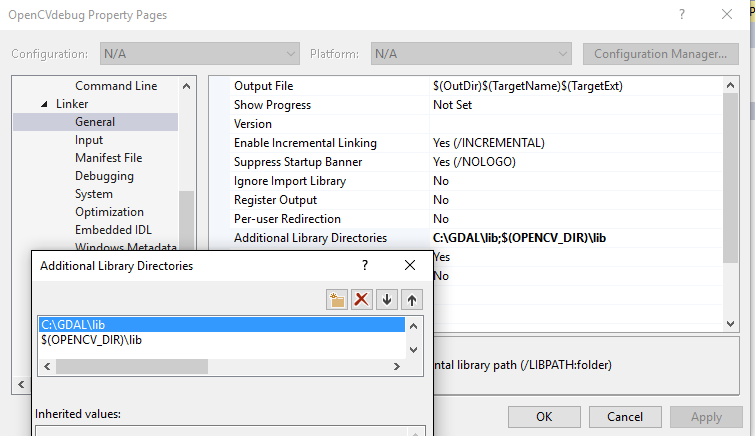


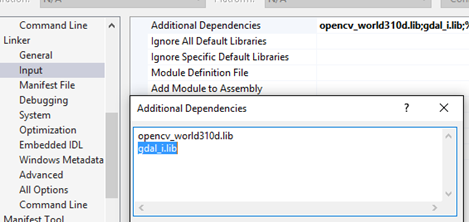


**ERDAS ECW SDK 5.2.1**

1. Install ERDAS ECW SDK 5.2.1. (<http://download.intergraph.com/?ProductName=ECW%20Plugins>)



1. Run ECWJP2SDKSetup\_5.2.1 after extracted. Leave it as Desktop Read-Only Redistributable, click next, then agree, and then selected a destination folder, preferably like C:\ERDAS.
2. Go into your system’s environment variables (just search for environment variables in windows start) and add these paths.
3. Next navigate to the gdal-2.0.1 folder. For me the path was C:\GDAL\gdal-2.0.1. There should be a file called nmake.opt. Find this block of code and make the edits shown below (open nmake.opt with wordpad and follow commented directions).
4. ECWDIR is the directory to the ERDAS 5.2.1 SDK (change directory name, if you named it differently than I did). ECWFLAGS is the version, change to 521. Change ECWLIB so that it uses vs120 and x64 support.
5. Near the top of the file you should see this, change MSVC\_VER= to your version (“MSVC 12” would be the value “1800”). Save and exit.
6. Next open your VS2013 x64 Native Tools Command Prompt (search for the Microsoft visual studio app, right click it and click “go to file location”, and it’s located in “visual studio tools”). Cd to where the nmake.opt file is located (C:\GDAL\gdal-2.0.1), and do “nmake /f makefile.vc”
7. After that’s done unpacking do “nmake /f makefile.vc install”
8. After that, lastly do “nmake /f makefile.vc devinstall”
9. Now open Visual Studio and create a new project with the following property sheets (you can just edit the OpenCV ones).
10. Under C/C++ General: Additional Include Dir, add path C:\GDAL\include Under Linker General add C:\GDAL\lib folder in the Additional Library Directories

Under Linker Input add gdal\_i.lib as an additional dependency

1. Grab a Jp2 image and try to load it with Tsosie’s code now. It should be in the email he sent earlier. Once you have a jp2 image, change the path of this code to where you have your .jp2 saved.

Up to this point you should have that warmerda folder in the C drive.

Change the line of code near the end to write the down-sampled image to this folder.



You may need to save and reopen VS before it can compile. Then run the program.

1. Depending on how big the jp2 is, it should take a little bit. Navigate to the warmerda path the new jp2 image is in (C:\warmerda\bld\bin). Open the x64 native tools command prompt again and cd to this location. Now try ”gdal\_translate downsampled.jp2 output.tiff”.

It should spit out the downsampled jp2 image as a tiff image now named “output.tiff”