WINDOWS IMAGE MANAGER

Image Display and Analysis Program for

Microsoft® Windows™

with Special Features for Satellite Images

USER'S MANUAL

Version 12

October 2023

WIM is a software tool for satellite image analysis under Microsoft Windows. WIM Automation Module (WAM) are the library, applications and a set of tools for automating WIM tasks. WimLE was a limited version of WIM that is no longer supported.

For the latest version and other news please check the WIM World Wide Web home page

https://www.wimsoft.com

For information or orders send E-mail inquiries to:

wim@wimsoft.com

Software Warranty

The WIM software is licensed solely on an as is basis, i.e., the entire risk as to its quality and performance is with the purchaser. In no event will Mati Kahru be liable for direct, indirect, incidental, or consequential damages resulting from any defect in the software.

Contents

1 Introduction

WIM is a general-purpose image visualization and analysis program for the Microsoft® Windows™ operating systems with special features for analyzing satellite images.

WIM has been designed to work with digital images. In the WIM parlor an image is simply a two-dimensional array (matrix) of numbers representing the values of its elements (pixels). Digital images of this kind are produced by satellite sensors but also by medical imaging, computer models, etc.

There are many ways and formats to store digital images. The simplest is a sequence of numbers (e.g., row by row) representing the pixels which is often referred to as unformatted raster image. In this case the total number of pixels equals the number of rows times the number of columns, and the file size equals the number of pixels times the number of bytes per pixel. In the simplest case each pixel is represented by one byte. Complex file formats, e.g., HDF or netCDF, add a lot of additional information to the file, e.g., dimensions, color palettes, geometric projections, and other attributes. The image data may be internally compressed to reduce the file size.

The most prevalent file type for satellite data is currently netCDF. Before it was HDF4. While WIM supports many file formats such as **netCDF**, **HDF4**, **HDF5** and others, **HDF4** is the main format within WIM.

Other formats that WIM supports are the following:

The most primitive image formats are unformatted raster images of unsigned byte (Byte or UINT8), one bit, unsigned and signed integer (2 bytes, Int16 and UINT16), long integer (4 bytes, Int32 and UINT32) or floating point (4 bytes, Float32) pixels. A special type of unformatted raster images corresponding to the North-East coastline of the US (NEC) in the Lambert Conic projection.

ASCII files;

Multi-band images of band-sequential, line-interleaved and pixel-interleaved types;

CoastWatch (NOAA/NESDIS) formats of compressed or uncompressed images that include ancillary information and may include overlays of coastlines and other features;

Compressed (run-length encoded) images;

Erdas/LAN format images;

NOAA/AVHRR Level-1B format files with multiple bands (images), see http://www.saa.noaa.gov/.

HDF (Hierarchical Data Format, both versions 4 and 5) with particularly the Scientific Data Sets (SDS) are used for storing data from SeaWiFS, MODIS, SSM/I, OCTS, MOS and other satellite sensors. HDF4 files can also be created from the Terascan™ TDF file format. Starting in 2015 NASA Ocean Color Processing Group as well as most other data sources have switched from HDF4 to **netCDF** (version 4) as the standard file format. WIM is capable to read and write netCDF files but the recommended file format for saving data in WIM is still HDF4..

Unexpectedly, older versions (2 and 3) of **netCDF** files can be read in WIM as HDF files while starting with version 4 they can be read only as netCDF.

Some satellite datasets, e.g. from Aquarius and most recently from the Japanese Aerospace Exploration Agency's (JAXA) SGLI sensor are in **HDF5**.

WIM makes a distinction between raster images and bitmap type objects (e.g. bitmaps, GIF, PNG, JPEG, and other similar file types). Whereas in raster images the emphasis is on the digital value of a pixel - a number does not have a color - the purpose of a bitmap object is its visual the appearance with colors. There are many programs that deal with bitmaps (MS *Paint* has been a perennial Windows companion) WIM is **not** a program for reading and visualizing bitmaps. WIM performs operations on images represented by numbers and not on bitmaps. WIM transforms digital raster images to bitmaps and can save as BMP, GIF, JPEG, PNG, TIFF and other bitmap type files, but not *vice versa*.

2 Hardware and Software Requirements

WIM is designed for the current Windows™ operating systems. Support for the older Windows versions (e.g., Win95, Win98, XP, etc.) is being gradually dropped and the latest versions are preferred. WIM depends on many system components provided by Windows. Support for 32-bit Windows has been dropped. We therefore strongly recommend to use the latest version of **Windows** and the 64-bit version of it.

Please note that in order to install WIM the user installing WIM needs to have the ***Administrator*** permissions.

As other Windows™ applications, WIM uses the display and printer drivers that have been installed in Windows™. It is therefore relatively independent of the actual hardware, assuming that it is compatible with Microsoft® Windows.

Depending on your display facilities (graphics adapter, monitor and the active display driver), the number of colors and the pixel resolution are different. You need **at least 256-color** graphics mode (8-bit color) to use WIM while the higher bit depths (16, 24 and 32 bits) should be used if possible. Most current graphics cards have 32-bit color capability. The preferred screen resolution depends on the size of your monitor. For viewing satellite images you should use at least the 1024 x 768 resolution.

3 Installing WIM

To install WIM/WAM you need to run the setup file which has a slightly different name corresponding to the output. The setup file for WIM+WAM in 64-bit Windows is named *wam64.msi*; the same for 32-bit Windows was *wam32.msi*. The 32-bit versions have been phased out. In the setup dialog please follow the default (*Typical*) options. If you have an older copy of WIM/WAM installed, you need to **uninstall** the older copy before installing the newer version.

The setup program makes a new directory for Wimsoft (the default is *C:\Program Files (x86)\Wimsoft*), expands and copies the required files.

Note that to install WIM/WAM you need to have ***Administrator*** permissions.

The setup process creates some separate folders under the *Wimsoft* folder such as *Maps*, *Examples, LUT* that include:

- sample images, a sample lookup table, etc. - they help you get familiar with WIM functions and can be deleted to save disk space.
- Terascan™ map projection files for North America.
- map files for generating overlays of coastlines, country boundaries and rivers for any image with a projection information.
- The WIM DVD/USB has folders *Images* that includes sample satellite images and a *Sat* folder with large sets of satellite data that are used in a course. The automated installation procedure will not install the *Images* or *Sat* folders. The *Images* and *Sat* folders are available online at respectively https://drive.google.com/open?id=0BwHWkbiHZx-0Rzc5YnVMalBMZ1U and https://drive.google.com/open?id=0BwHWkbiHZx-0Sno4d3V2djZEbkk.

When you run WIM for the first time you are requested to type in your license number that you received with WIM. If you change the hardware configuration of your computer or re-install Windows™ you need to enter the license number again.
If you installed WIM with the evaluation license, then you have to receive and enter the final license number in order to continue using WIM beyond the evaluation period.

4 Getting Started

Here we learn how to use the most basic operations in WIM. More detailed instructions are give in the "*Practical exercises with WIM and WAM*" and other exercises under the "Manual" section at https://www.wimsoft.com/manual.htm or in the *Course* folder (in the *Wimsoft* folder).

- You can load an image with the standard *File-Open…* dialog (an icon on the toolbar) but a much faster way is to find the necessary image files with the *Windows Explorer* and load them by just clicking (or double-clicking, depending on your Windows setup) on the filename. To be able to do that you need to associate WIM with the specific file extension. Click (or double-click) on any HDF image file (e.g. *composite.hdf* in *Examples* or *Images\hdf\SeaWiFS\baja_2000_april*) and select Wim.exe (typically in *C:\Program Files\WimSoft*) as the program to open it with. Then select "*read as HDF*" for that file type in WIM. You have to repeat that for HDF files with a different extension (e.g. SeaWiFS global Level-3 files like *S20000612000091.L3m_MO_CHLO* in *Images\hdf\SeaWiFS\Level3*).

- If you load images with the *Open* icon on the toolbar (menu *File – Open…*) you need to pay attention to the type of images that you are trying to read. WIM can read and write many types of images, but you have to select the correct **Files of type** in the dialog. The next time you are trying to read or write an image the type used last time is already selected as default for you. In each folder you will be shown only the files that have the appropriate extension for the selected file type. You can see all the files by typing *.* in the *File name* edit box. If your files are not shown then you can add your extensions to the default extensions in *View - Settings -* Extensions

- To look up pixel values hold down the **right button** of your mouse and move the mouse on the image. In the top part of the image window frame you will see values in the format X;Y (Lon; Lat)=PV (SV) where X and Y are image coordinates (X=distance in pixels from the leftmost column, Y=distance in pixels from the top row), Lon and Lat are the corresponding longitude and latitude (note the order!), the pixel value (PV) and the scaled value (SV in geophysical units). If the image has no projection, then the corresponding Lat and Lon are not shown.

- Select an area of interest (e.g., a rectangle) by moving the mouse with the **left button** down. Assuming that you successfully opened *composite.hdf*, select an area of interest and apply the *Histogram* function (*Histogram* icon on the toolbar or menu *Examine – Histogram - Calculate*). Many functions (Histogram, Statistics, Profiles, etc.) operate on the selected area of interest (either a rectangle or a line).

- To create a coastline overlay for an image select menu *Geo – Get Map Overlay*. Then pick a map file with global coastlines, e.g., ***coast_inter.b*** or *coast_low.b*. All the map files are located in the *Maps* folder (e.g. in

C:\Program Files\WimSoft\Maps or in *C:\Program Files\ WimSoft\ Examples*). WIM memorizes the locations and the parameters so that when you do it the next time you will be directed to that folder at once). When creating an overlay make sure that the background pixel value is 0 and the foreground pixel value is either 255 (white) or 1 (black).

- To overlay coastlines or other features on an image, click on the image in WIM, then on the *Overlay* icon on the toolbar (or *Multi –Overlay Image* in the menu). Make sure the image number to be overlaid is correct. The current image is assumed when looking up values or performing operations (e.g., *Histogram*, *Statistics*). You can make an image current by either **left clicking** on it or **double clicking** on it in the List of Images box.

- The basic information about an image is shown in the List of Images box but more detailed image attributes are shown when clicking on the *Attributes* icon on the toolbar (*View –Attributes* in the menu).

- Scaling and other settings of an image are shown in the in the *Current Settings* box (*Settings* icon on the toolbar or *View – Settings* in the menu). For example, check that the scaling for the *chlor_a* image in file *composite.hdf* is *Logarithmic* with *Slope* = 0.015 and *Intercept* = -2.0 (equivalent to the *Chl* scaling) and the projection is *Linear*.

- You can read multiple images at once by clicking on the *Open* icon on the toolbar (menu *File – Open…*) and selecting multiple files. For example, you can read all the HDF files from *Images\hdf\SeaWiFS\baja_2000_april* in one step.

- Each file can contain multiple images of its own. For example, you can load 3 SeaWiFS Level-1A images from *sw_l1a_mapped_byte.hdf* in folder *Images\hdf\SeaWiFS* and then try *Examine* - RGB_Image and choose the 412, 555 and 670 nm bands as the red, green and blue components, respectively. You can generate almost unlimited color combinations by adjusting the Min (down) and Max (up) scrollbars for each component.

5 WIM Basics

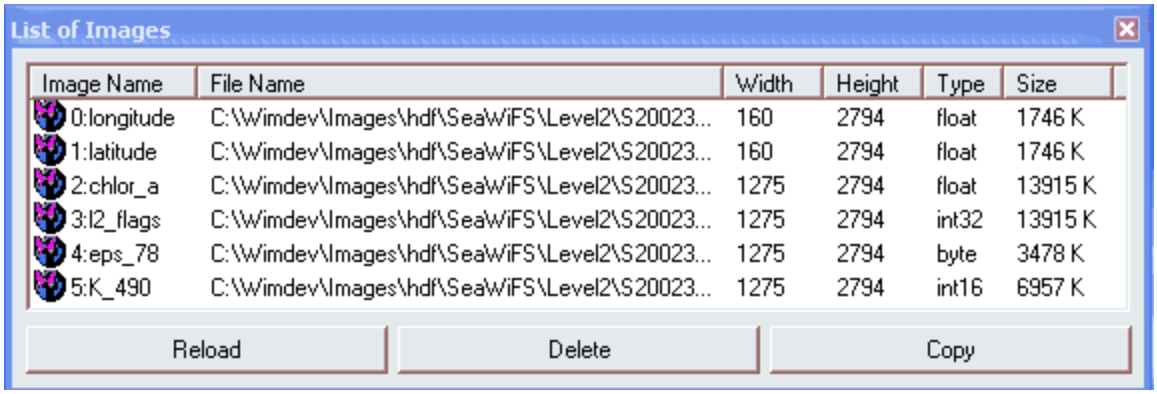
The basic data unit for WIM is a digital raster image. An unformatted raster image, shortly, an image, is simply a sequence of numbers representing picture elements (= pixels). The number of pixel values has to be the number of rows times the number of columns. The sequence of pixel values starts from the top left corner and continues row-wise to the bottom right corner. A pixel value is often represented by one unsigned byte that can have values from 0 to 255. WIM can also handle images with pixel values of 1 bit, 2-byte signed or unsigned integers, 4-byte floats and integers.

Another type of raster object is called bitmap. While an image is primarily a sequence of its pixel values, a bitmap file follows a strict structure and includes all the information needed to display the visual color image. When an intelligent program reads a bitmap, it can immediately recognize it as a bitmap, and has all the information to build the respective rendering on the screen. In contrast, when a program reads an image file, it needs additional information (e.g. the image dimensions, colors corresponding to pixel values, etc.) to build a picture. That additional information is often stored in the image file together with the image data. In the “old days” this additional information was often put into fixed “headers” while current formats such as HDF can store almost unlimited number of different attributes. WIM uses headers when reading specialized image formats, e.g. NOAA/NESDIS CoastWatch, Erdas Lan, and it can be set to skip a predefined number of header bytes. When reading plain (unstructured) raster image files WIM can make use of a small complementary info-file that provides basic additional data describing the image (size, value scaling of pixel values, geometric projection parameters). If no info-file is found, WIM assumes the current or default values. In order to read image formats containing an attached header (other than CoastWatch and Erdas/Lan) it is possible to skip the header bytes from the beginning of the file. A few programs (e.g. CCAR navigate), instead of appending an image to its header, overwrite the beginning pixel values in the image with the header data.

An image (i.e. a sequence of bytes representing the pixel values) may reside both in a disk file or in a memory buffer (image buffer). In order to display an image from a file, WIM allocates a memory buffer for it, reads the pixel values from the file into the buffer, converts the image buffer to a bitmap and then uses Windows™ functions to display the bitmap on the screen. Both the image buffer and the corresponding bitmap occupy parts of the computer's RAM. Up to 64 image buffers can reside in memory at the same time. While reading images from disk files or generating new images itself, WIM grabs additional memory from the computer's RAM for the image buffers and the corresponding bitmaps. When RAM becomes scarce for loading new, big images, the user should free up RAM by deleting other image buffers from memory.

The names of all loaded images with their dimensions (***Width*** = *DX* and ***Height*** = *DY*), ***Type*** and ***Size*** are listed in the ***List of Images*** dialog box.

The following example shows 6 loaded images of 4 different types: *byte*, 2-byte integer (*int16*), 4-byte integer (*int32*) and *float* (also 4 bytes per pixel). These images are all read from the same SeaWiFS Level-2 file in HDF format.



The ***List of Images*** can also be used for several operations on images in memory.

To switch between different images double-clicking on the selected image.

Reload reloads the bitmap from the image buffer and recreates its bitmap. **You may sometimes need to run Reload on a selected image buffer when the image seems to have a problem with its colors.**

Delete deletes selected image(s) from memory.

Copy makes an identical copy of the selected image to a new image buffer. You can also ***Rename*** an image name by simply editing the name.

By selecting an image and then selecting another image while holding down the Shift key a whole range of image buffers can be selected for deleting from memory. Multiple image buffers can be individually selected by selecting them while holding down the Control key. Renaming an image may be useful if you want to save an image buffer in a file under a different name and then want to save the corresponding info file (*File* - Save Info).

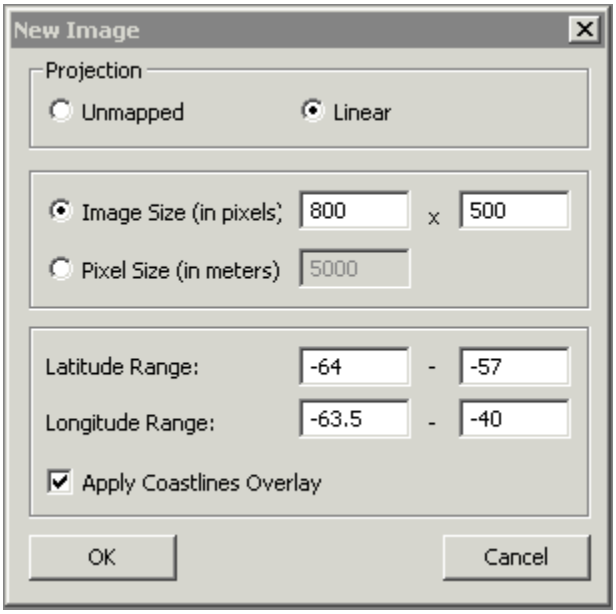
6 Menu System

The main menu system provides the following choices: *File, Edit, View, Examine, Geo, Transf, Multi*, *Segm, Window, Help*. Note: Some of the menu options are not available for WimLE and WimCW versions. The most frequently used functions have shortcuts on the Toolbar:

6.1 File - File Operations

New

Opens a new, "empty" (all pixel values set to zero) image buffer named <New Image> with one byte per pixel and either with selected width (*DX*) and height (*DY*) in pixels or the approximate pixel size in meters. The WIM *Linear* projection can be set with a specified *Latitude* and *Longitude* range. Coastlines can be automatically created and overlaid with a coastlines file specified in *Settings – Misc*.



Filter to Disk

see *Transf* - Filter to Disk.

Open...

Opens a file on the disk, reads one or more images to the corresponding image buffers and displays them on the screen. More complex image formats (CoastWatch, HDF) usually include all the necessary information to read and decode the image into geo-physical values. Simple raster images

with no additional information may need additional information (e.g. the size of the image) that can be read from an optional info file.

Before allocating a new memory buffer for the image, WIM first checks if there is an info file in the same directory for the selected image file. The info file is an ASCII text file with the same name as the image file but with an extension **.inf* containing *DX, DY,* and optionally value scaling and parameters for geo-registration (i.e. converting between image *X/Y* and Earth longitude/latitude coordinate systems). **Note**: If an info file exists and contains data, the number of pixels in a line (*DX*) and the number of lines (*DY*) are retrieved from it, disregarding the current values (see *View - Settings - General -* Image Size). With no info file in the same directory, the current image dimensions are assumed for the new image. Before actually loading the image, the user can change the image dimensions. The new image buffer of *DY* times *DX* bytes is allocated and the number of buffers in use is increased by one. The default image size during start-up is 512 x 512. You can change it with *Edit - Size* or *View - Settings - General -* Image Size. If your images are of variable size, it is convenient to save the info files for each of them (*File -* Save Info). Info files can also be created manually (e.g. with an editor) by writing the respective *DX* and *DY* into them. If you have specified a header length different from 0 (*View - Settings - Misc – Image Header*), the specified number of bytes is skipped from the beginning of the image file. In Windows™ bitmap rows must contain a multiple of four bytes. If *DX* is not a multiple of four, the bitmap rows are padded on the right to insure this. It is therefore a good practice to use a multiple of 4 for *DX*

The following file types can be selected:

Image

Reads unformatted raster image with one unsigned byte per pixel. Before actually reading the image buffer WIM checks if the buffer contains certain header information corresponding to the CCAR navigate software (Baldwin and Emery, 1993; Emery, 1995). If that is true then the user may either accept these parameters (image dimensions and projection parameters) or cancel and discard the header information.

NEC Image

A special version of the raster image with one unsigned byte per pixel corresponds to the North-East coastline (NEC) of the United States. These images are always 1024 x 1024 bytes and the sequence of bytes starts from the bottom row in contrast to a standard Image that starts from the top pixel row. The files containing images in NEC format should have the extension *.nec*. NEC images are assumed to be in the Lambert Conic projection.

ASCII

Reads an ASCII text file of pixel values into a Byte image with each pixel value (from 0 to 255) represented by up to 3 characters and separated by one or more separators. Separators can be a space, a tab character, a comma or a semicolon. Multiple consecutive separators are treated as a single separator. Negative values are not allowed (are set to 0) and values larger than 255 are set to 255. The following is an example of an image of 4 rows and 4 columns (an underscore stands for any separator):

```
255_200_180_111
100_90_8_122
8_100_100_234
199_212_231_123
```

Tab or comma separated (CSV) text files can be exported from many programs (e.g. Excel).
The *File* - *Save as ASCII* function in WIM also saves Byte images (space separated) in this format but ONLY if you set the *Settings* - *Value scaling* to *Pixel Value*. If *Value Scaling* is different from *Pixel Value* then the saved text file will probably have floating point numbers as text and be readable with the WIM function *Read ASCII Float* but not with *Read ASCII* as the text will have floating point numbers and not integers.
If the image is too small for viewing, you can do multiple expansions of it with *Transf - Zoom*.

ASCII Float

Is similar to Read ASCII but reads a text file of pixel values into a *Float32* (4 bytes per pixel) image buffer. The pixel values may be either integers or float numbers without strict restrictions on their length. As with Read ASCII the separators can be either a space, a tab character, a comma or a semicolon and multiple consecutive separators are treated as one. Before reading the file you can also set the color scaling parameters (see *View* - *Settings* - *General* - *Value Scaling*). Examples of small 4 x 4 pixel ASCII "images" in space and comma separated formats are given below.

```
255 200 180 111          0.167,0.176,0.22.0.24
100 90 8 122             0.10,0.133,0.188,0.211
8 100 100 234            0.166,0.0628,0.550,0.5
199 212 231 123          0.12,0.21,0.1666,0.18
```

Band Sequential

Line Interleaved

Pixel Interleaved

Reads files that contain several bands (channels) of image data either in band sequential (BSQ), band interleaved by line (BIL) or band interleaved by pixel (BIP) formats. BSQ is just a sequence of several images concatenated one after another. In the BIL format a line (row) of one band follows the corresponding line (row) of another band. In the BIP format a pixel of one band is followed by a corresponding pixel of another band.

You can specify any of the bands to be read, or the whole set, if you specify band 0. Either one or several new image buffers of *DY* times *DX* bytes are allocated. A special file type (Erdas/Lan) is included for reading ERDAS BIL or BIP files but the same can be accomplished by setting the header length to 128 with *View* - *Settings* - *General* – *Header Length*. A well-known image transformation program *Alchemy* produces BIP files with the BIF option (-B)

when the source image has more than 8-bits per pixel. In case where the different bands can be used individually, it might be more efficient to organize the data in the BSQ format. This would allow accessing the bands with the simple *File* - Open *Image* method (possibly skipping the previous band(s) as a "header").

CoastWatch

Reads a NOAA/NESDIS image file format (see http://www.nodc.noaa.gov/NCAAS/ncaas-home.html) that contains compressed or uncompressed image with ancillary information and potentially a graphic overlay with coastlines and latitude-longitude grid. Depending on the source of the CoastWatch file, the graphics overlay buffer may be missing or useless. The image dimensions as well as other ancillary data are automatically read from the file header. Although the CoastWatch image dimensions are usually 512 x 512 pixels, other sizes (e.g. 1302 x 1401 for the Northeast US full resolution image) can be handled as well. As the CoastWatch pixel values are 11-bit values, they have to be transformed. Four options are presented: *Truncate*, *Albedo*, *Temperature* and *2-byte/pixel buffer*. You should normally use either *Temperature* (for thermal bands and SST) or *Albedo* (for visible bands). *Albedo* and *Temperature* transform the values into albedo percentages or temperatures, respectively. When *Temperature* scaling is selected the resulting image is scaled as X10 (i.e. temperature in C multiplied by 10) or as SST Pathfinder. *Truncate* with Shift 0 picks the least significant 8 bits. *Truncate* with Shift 1 discards the least significant bit (equivalent to dividing by 2) and then uses the next 8 bits, *Truncate* with Shift 2 discards 2 least significant bits and so on. You have to choose a range between the lowest and highest value and the values in this range will be mapped to the pixel range of 0-255. The 2-byte option reads the image values directly into a 2-byte buffer without discarding any bits, and the graphics overlay into a 1-byte/pixel buffer. The graphics overlay usually contains the coordinate grid and coastlines. You can overlay it on the image with *Multi* - Overlay Image. If the match between the image and the overlay is not good, you can use *Multi* - Shift Image to shift the image relative to the overlay.

When reading a CoastWatch image, WIM first reads the header and displays the header information. Among others, the projection parameters and the latitude and longitude range are read from the header. Both Mercator and Linear projections are implemented (right clicking or *Examine* - Peeker shows the pixel coordinates). You can check the *Linear* projection's latitude-longitude coefficients with *View* - *Settings* - Projection or with *View* - Attributes.

If the longitude/latitude grid has been written into the image data by changing the pixel values to zero, you can fill it in with *Segm* - Fill Holes.

Compressed (RLE)

Reads a run-length-encodes (RLE) 1-byte-per-pixel image format. Correct image dimensions (*DX* and *DY*) have to be used or the RLE-decoding may not work. A sample RLE-encoded image (*s_califm.rle*) contains the latitude/longitude grid of the *s_calif1.img* and *s_calif_2.img* images in *Conic* projection. RLE-encoding is very efficient for storing images with large constant areas, e.g. overlays with latitude/longitude grids and coastlines.

You can produce RLE-compressed files with *File - Save As - .Compressed (RLE).*

Erdas/Lan

Reads a special file format introduced by a popular image-processing program (by Erdas, Inc.). The 128-byte header contains information on the number of bands (band interleaved by line, BIL), the pixel size (4, 8 or 16 bits) and image dimensions. The first 4 bytes of the header are assumed to be *HEAD*). At present only 8-bit-pixel images can be read. The extension of Erdas/Lan files is usually **.lan*. The user is given the option to select the set of bands and the subset of the *X* and *Y* ranges to be read. Subsetting is sometimes required if the Erdas/Lan file has been produced on a Unix workstation and is too big to be handled by the PC. The names of the image buffers will have the number of the band appended. Please note that the bands are numbered from *1* to *N* but the lines and pixels are numbered from *0* to *M-1* where *M* is the *X* or *Y* range of the image.

Float

Reads a plain (unformatted) raster image with each pixel represented by a 4-byte float number in the "small-endian" format (cf. the next section).

For displaying a float image the values are scaled linearly into 256 bins between *Min value* and *Max value* (see *View - Settings - General* – Color Scaling. An easy way to find the minimum and maximum values and stretch the color-coding between them is to use *View* - LUT Stretch. The minimum and maximum values can then be viewed in the *View - Settings - General - Scaling for Integer/Float.*
Some image operations are either not available or not convenient to use for float image buffers. In that case it is recommended to convert the float image to Byte with *Transf* - Convert and a suitable scaling type.

Float Bigendian

Like the previous option, Float Bigendian reads a Float32 raster image but in the "big-endian" format. The term "big-endian" or "small-endian" refers to whether the most significant bytes of a 4-byte float number are in the end or in the beginning. The bigendian format files are usually generated by Sun SPARC, SGI and others systems and are a popular format in the Unix world.

GOES-SST

A special version of the raster (byte) image with the upper left corner at (180W,60N) and the lower right corner at (30W, 45S). These images are always 3000 columns by 2100 rows and in a simple Linear projection (gc coefficients are respectively –180, 0.05, 60, -0.05). The scaling of the SST is linear with slope=0.15, Intercept=-3.15. When read as GOES-SST WIM assigns the correct scaling and projection. These files can also be read as generic raster images but then the user has to manually set the size, scaling and projection.

HDF file

Hierarchical Data Format (HDF, version 4) is a versatile data file format developed by the National Center for Supercomputing Applications at the

University of Illinois, http://hdf.ncsa.uiuc.edu and adopted by NASA as the standard data exchange format for various data-sets in their Earth Observing System (EOS) software.
HDF can store an almost unlimited number of data objects in a single file. In the context of image analysis the basic object types are: raster images (8-bit and 24-bit) with the associated color lookup palettes (for a raster-8 image) and scientific data sets (SDS), i.e. multidimensional numerical arrays. A multidimensional array in the SDS can contain 8-, 16-, or 32-bit signed or unsigned integers or 32- or 64-bit floating-point numbers (int8, uint8, int16, uint16, int32, uint32, float32, float64). Ancillary information and non-image data of the HDF file (attributes) can be examine with *View* - Attributes.

When reading an HDF file WIM tries to find information on value scaling (see *View - Settings - General* - Value Scaling*)* and projection. If there is a palette in the HDF file, the current lookup table is replaced with the data from the new palette and the palette flag is set to *Custom* (see *View - Settings - General* - Palette*)*. The last palette read replaces the previous *Custom* lookup table.
Due to the increasing popularity of HDF in storing and transferring satellite image data, WIM includes many features for handling the special features of SSM/I, SeaWiFS, MOS, OCTS images and sea-surface temperature data in various formats. Some of these features are described in later sections of this document.

WIM can also save images with the accompanying auxiliary information in HDF files (File - Save as … HDF) that makes HDF the preferred file format when working with WIM. In addition to images WIM can also store locations of Vector Objects (such as points, transects and rectangles but not text labels) in a HDF file.

HDF5 file

The newer version of Hierarchical Data Format (HDF, version 6) has not been very popular for satellite datasets but has been recently adopted for the new JAXA SGLI sensor (see under JAXA SGLI).

NOAA Level-1B (L1B) file – obsolete!

NOAA Level-1B format is used by NOAA to store the Advanced Very High Resolution Radiometer (AVHRR) sensor data from NOAA's Polar Orbiting Environmental Satellites (POES). Current AVHRR sensors have 5 channels (bands) with bands 1-2 in the visible-near infrared and bands 3-5 in thermal infrared. NOAA Level-1B is a very complex format with a large number of modifications due to different sensors and options. The basic types of Level-1B files are the GAC (Global Area Coverage) and the LAC/HRPT versions. The GAC data is produced by sampling the AVHRR 1 km data at 4 km reduced resolution. The LAC (Local Area Coverage) Level-1B data are 1 km resolution data that are recorded on-board the satellite and dumped at the receiving stations at a later time. The HRPT Level-1B data are 1 km direct readout data obtained as the satellite passes over receiving stations. The Level-1B data normally include all 5 channels at 10 bit precision, with time tags, Earth location, and calibration information. It is also possible to unpack the 10-bit data into 16 or 8 bits.

The WIM adaptation of the Level-1B format is based on the Land Analysis System (LAS)/AVHRR Data Acquisition and Processing System(ADAPS) of the US Geological Survey (see http://edcwww.cr.usgs.gov/programs/sddm/lasdist/info/index.html).
WIM includes functions for reading the Level-1B files. Level-1B data can be ordered from the NOAA/NESDIS Satellite Active Archive (http://www.saa.noaa.gov) and from other satellite receiving stations. In spite of being a standard format, many varieties exist due to differences between sensors and file creation procedures. Not all versions have been tested with WIM. The default options of 5 bands at 10-bit sampling are supported. Earth-locating and calibration of the Level-1B data from different AVHRR sensors depends on a large number coefficients included in a number of tables. These tables are included in a directory ADAPSTABLES that the user should copy manually from the WIM CD to your hard disk (the default target location being *C:\Program Files\WimSoft\ADAPSTABLES*). Before these tables become available to WIM the user has to create an environmental variable ADPAPSTABLES and set it to the location of the directory (e.g. to *C:\Program Files\WimSoft\ADAPSTABLES*). On Windows 2000/XP systems environmental variables can be set by right-clicking on *My Computer* and selecting *Properties/Advanced/Environment Variables/New*. Please note that the tables must be in the Unix text mode, i.e. line ends with a single LF character and not with a CR character. Updated tables can be downloaded from ftp://edcftp.cr.usgs.gov/orders/g1k/. Please download the tables as **binary** and not as ASCII text.
AVHRR channels 1-2 are calibrated into percent reflectance (values from 0 to 100) and channels 3-5 into temperature in degrees C. In case of errors (a common case being that a required table is missing or not accessible from the ADAPSTABLES directory) a log is shown. The full log of processing each Level-1B file can be viewed by using the Attribute icon on the toolbar or menu *View* – Attributes).
By right-clicking on the image the X, Y and the Longitude and Latitude of each pixel are shown along with the pixel value and geophysical value (percent reflectance or temperature in degrees C). Coastlines and other overlies can be created by using *Geo* - Get Map Overlay . Be warned that this operation is very slow for Level-1B images. The generated overlies can be inserted into the image by using *Multi* – Overlay Image. Sometimes the geo-location has a significant error that can be manually corrected with Multi - Shift Image.

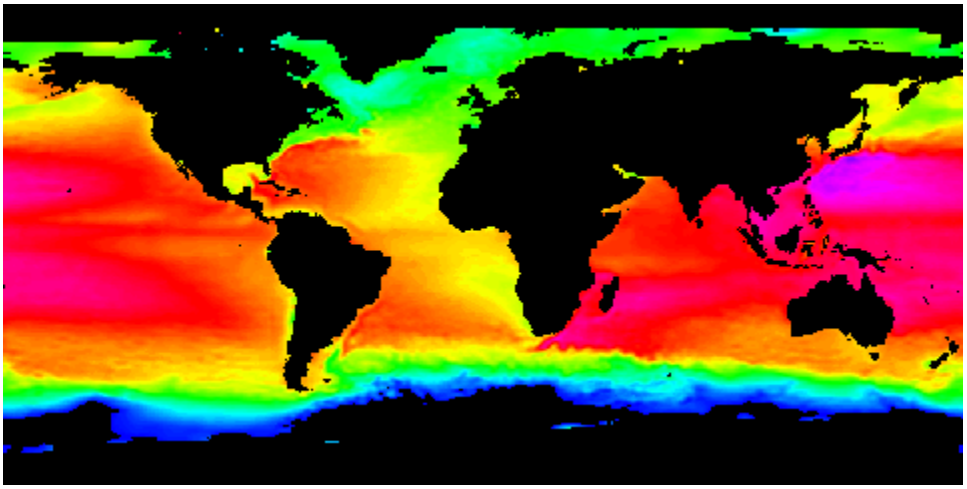
Lat, Lon, Value ASCII

Loads individual triplets of *Latitude*, *Longitude* and *Value* from a text file, creates a *Float32* image in *Global Equal Angle* projection. The size of the image is taken from *Settings – Misc - Default Navoceano Size*. Each triplet in the input file is assumed to be on a separate line. The sequence can be either *Longitude, Latitude, Value* or *Latitude, Longitude, Value* depending on the "*Lat first*" checkbox in *View – Settings – Misc.* Each line may have more data after the *Value* – only the first columns are used and the rest are ignored. For example, the input file may have lines like

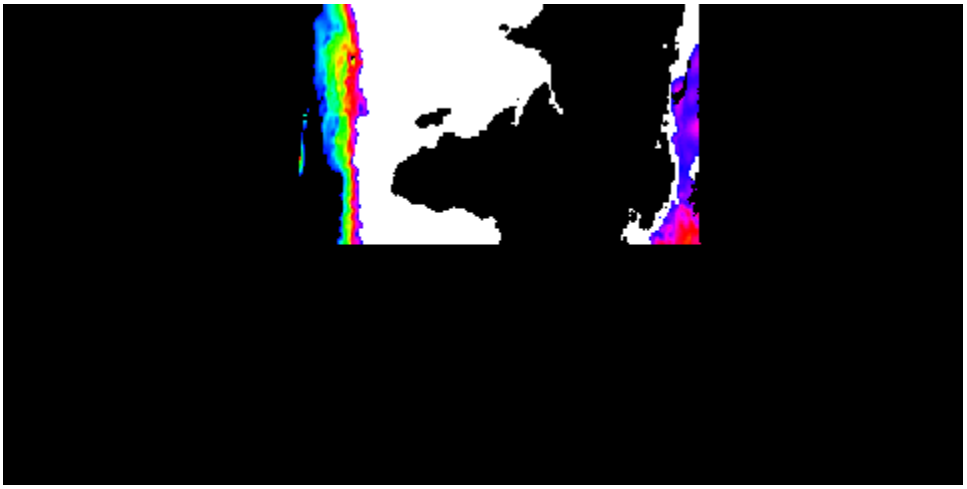
```
180.2500  -78.2500    0.0094
180.2500  -77.7500    0.0087
```

```
180.2500  -77.2500    0.0108
180.2500  -76.7500    0.0155
180.2500  -76.2500    0.0180
.....
```

The resulting image may look like this:



However, if the *Latitude/Longitude* sequence is chosen incorrectly the resulting image may look like this:



The *Latitude* is assumed to range from –90 to 90 and *Longitude* from –180 to 180. If *Longitude* is over 180 then it will be normalized to the range –180 to 180. It is natural that depending on the specified size and the number of specified pixels the resulting image may be sparse, i.e. have empty pixels between pixels with value.

NAVOCEANO L2 (*.def) – obsolete!

Sea Surface Temperature (SST) datasets derived from the NOAA (National Oceanic and Atmospheric Administration)-Polar Orbiting Advanced Very

High Resolution Radiometer (AVHRR) using the multichannel sea surface temperature algorithm (MCSST) are being generated by the US Naval Oceanographic Office (NAVOCEANO) and distributed at a near real-time rate by NASA's Physical Oceanography Distributed Active Archive Center (PO.DAAC.

The individual Level-2 NAVOCEANO format datasets (*.def) are not images but Latitude-Longitude dataset derived from the GAC (Global Area Coverage) data of individual orbits. These datasets are converted to images in the Global Equal Angle projection by WIM. When reading a single Level-2 NAVOCEANO dataset WIM creates a single global image. The default image size is 720 x 360 pixels but this can be changed in the *Settings* - *Misc* – *Default Navoceano Size*. As individual datasets provide a relatively narrow swath of data along the cloud-free portion of the satellite track, these datasets are usually composited over many passes in order to get good global coverage. When selecting multiple Level-2 NAVOCEANO images simultaneously WIM creates 2 images: a composited global image of average SST and another image with the number of pixels used in the composite. This operation is analogous to Multi – Composite. It may be tricky initially to select multiple images with WIM. In order to do that you should highlight one image (e.g. by right-clicking on it) and then selecting a range of images with *Shift-left-click* or selecting individual images with *Control-left-click*. After a set of images has been highlighted, click on the *Open* button of the dialog. Attributes of the composited image can be seen by using View – Attributes.

NAVOCEANO L3 (*.dat) – obsolete!

Starting from January, 2002 JPL PO.DAAC has made available gridded 18-km MCSST Level 3 (NAVOCEANO) images (their product #144) as weekly composites. As explained above you can easily composite NAVOCEANO data with WIM but the composites provided by PO.DAAC save time and provide additional products. The format that these Level-3 NAVOCEANO images are distributed is Band–Sequential binary. It is a simple format with 5 images appended to each other. Each image has 2048 columns and 1024 rows and uses 1 byte per pixel. The bands are Sea Surface Temperature, the number of points per bin, Sea Surface Temperature Anomaly, Interpolated Sea Surface Temperature, and Interpolated Sea Surface Temperature Anomaly. The files are separated by satellite (NOAA-16 of NOAA-14) and by pass (daytime or nighttime). The value scaling used for SST is *SST-Pathfinder* (i.e. a *Linear* scaling) with a *slope* of 0.15 *intercept* of –3.0. The *intercept* for the SST anomaly is –20.0 while the *slope* is the same. Starting with version 5.45 WIM has is a special file type for reading NAVOCEANO Level-3 datasets while previous versions of WIM can read these files as NAVOCEANO L3 or as generic band-sequential binary images. When read as NAVOCEANO L3 file type WIM automatically sets the scaling and projection parameters. When read as generic band-sequential images you have to manually sets Value scaling for the SST and SST anomaly images to SST-Pathfinder, projection to *Global Equal Angle*. As the sequence of image data is different from the WIM convention, *Transf – Mirror - Horiz. Axis* has to be applied to prevent the images being upside down.

netCDF (*.nc)

NetCDF is currently the most popular format for storing satellite data. It is very similar and shared many common parts with both HDF4 and HDF5.

Unsigned Int

Int -> Byte

Read a plain (unformatted) raster image with each pixel represented by a 2-byte unsigned integer. While ***Int->Byte*** converts it immediately to a 1-byte-per-pixel buffer, ***Unsigned Int*** retains 2 bytes per pixel in the allocated image buffer.

An unsigned integer of 2 bytes (16 bits) may have values from 0 to 65535. Depending on the convention either byte can be the most significant byte (MSB). Before loading the image from the file you have to specify if the MSB is the second (default) or first byte. Before actually reading the image buffer WIM checks if the buffer contains certain header information corresponding to the CCAR navigate software (Baldwin and Emery, 1993; Emery, 1995). If that is true then the user may either accept these parameters (image dimensions and projection parameters) or cancel and discard the header information.

For displaying an image of unsigned integer values (i.e. from 0 to 65535) the pixel values are scaled linearly into 256 bins between *Min value* and *Max value* (see *View - Settings – General –* Color Scaling). An easy way to find the minimum and maximum values and stretch the color coding between them is to use *View* - LUT Stretch.

Some image operations available for 1 byte-per-pixel image buffers are not implemented for 2 and 4 byte buffers. The operations not supported are grayed in the menus. You can convert between image buffers of different pixel size with *Transf* - Convert. The options in *Transf - 2 Byte To 1* are similar to the options when the 2-byte buffer is read and immediately converted to 1-byte buffer.
When converting the range of 0 to 65535 different values of an unsigned integer to the range of 0 to 255 possible for an unsigned byte, a transformation of values is needed. The representation of an unsigned integer depends whether the most significant byte (MSB) is the first or second. The pixel value is formed as 256 * MSB + LSB (least significant byte). Besides the byte order, you can choose many ways how to make the reduction from 16 bits to 8 bits. The easiest would be to shift the bits to the right. Shifting to the right by 1 is equal to division by 2, shifting by 2 corresponds to division by 4, etc. In essence, if you select shift 1, you will loose the least significant bit but retain the original values between 2 and 511 transformed to the range of 0-255. If you make a shift of 2, you will retain the original values between 4 and 1023. In WIM the bit-shift operation is coupled with magnitude test. If after the operation the value is still larger than 255, it will be set to 255. By specifying a negative shift number you simply select the case when the LSB is first, but with the same absolute shift to the right. Specifying 0-shift produces the second byte value only if the first byte is zero (i.e. the integer is less or equal to 255), and 255 if the first byte is different from zero. For the corresponding operation of 0-shift assuming that the LSB

is first, you have to select -16. By selecting shift 8 (or -8), you will extract the first (second) byte for the new pixel value. By selecting shift 17 (-17), the square root will be taken from the integer value to compress it into a byte, assuming that MSB is the first (second).
Special options are available for the case when the 2-byte value specifies temperature in ºK multiplied by 10. By selecting 18 (or -18 when MSB is second) the value is converted to ºC multiplied by 10 (i.e. 2731.5 is subtracted). By selecting 19 (or -19) the value is converted to a coded temperature (see *Transf* - Convert) that keeps temperature values between -3 and +27 ºC. In both cases temperatures outside the possible range get respectively the minimum or maximum value.
A related operation *File* - Open...2-byte Integer reads a similar image file but keeps it as a 2-byte buffer instead of converting into a 1-byte buffer.

Overlay – obsolete!

Reads a plain (unformatted) raster image with each pixel represented by a single bit (i.e. with possible values either 0 or 1). The bits have to be grouped into 8-bit bytes. The 1-bit-per-pixel image (here called overlay) is converted to a normal 1-byte-per-pixel image buffer.

The pixels with their corresponding bit = 1 will make a byte value 1, while zero bit will produce a byte value of 0. If, instead of the value 1, you would like the pixels to have values 255, you can use *Transf* - Binarize - *1* to make the conversion. You can then overlay the overlay image to your background image by using *Multi* – Overlay Image. NB! When an image was compressed to an overlay file, consecutive 8 bytes were compressed into one byte. If the number of pixels in lines was not divisible by 8, the image size had to be increased. It is therefore recommended to make overlays only from images with a multiple of 8 pixels per line. A sample overlay image (*est256.ovl*) corresponds to the sample image *est256.img*. It has to be noted that while 1-bit-per-pixel overlays provide 8 times compression for binarized images, RLE-encoding (see *File - Open...* Compressed (RLE*)* and *File* – Save As... Compressed (RLE) may give even better compression rates while keeping all the 8 bits per pixel.

Subset – obsolete!

Reads a subset from a 1-byte-per-pixel image file. Sometimes images (and the corresponding image files) are very big and even if you have a lot of RAM in your PC, the Windows™ functions that allocate RAM for WIM do not return enough RAM to accommodate both the image buffer and the bitmap. In these cases you can read only a part of the image and still be able to analyze and view it. You can select the image area that you want to extract and select which pixels you want to pick, i.e. every Nth pixel in *X* and every Mth pixel in *Y* direction. Please note that both pixels in a line as well as lines in an image start from 0. E.g. If your “big” image is 4096 x 2048 in size and you want to subsample the entire image by picking every 4th pixel in both X and Y (that produces a subsampled image of 1024 x 512) the (default) values of the ending range are 4095 and 2047, respectively. Similarly, if you would like to subsample the upper-left corner of 1024 by 1024, your ending range is 1023 and 1023, respectively. Due to a requirement of the MS Windows memory management, it is recommended that the width (*X)* of the resulting

image is a multiple of 4. If your final *X* dimension is not a multiple of 4, WIM will increase it, or if it reaches the right bound of the image, decrease it.
This operation is commonly used with images from the Landsat Thematic Mapper (TM). A "full scene" of TM data has 7 bands of 7942 times 6800 pixels. With each full image occupying about 50 MB it is often not possible to view the whole scene or make a RGB composite from a selection of 3 bands on an average or low-end PC. A convenient approach is to first read the individual images at lower resolution, e.g. every 10th column and every 10th row, make the RGB composites of those, and then read sub-areas of the scene at full resolution. A set of individual bands can be conveniently saved in one file with *File - Save As* - Erdas/Lan.

Copy Image

Makes a copy of the current image buffer into a new image buffer under the same name.

Cut Image

Cuts the specified rectangle of the current image into a new image buffer that is then displayed. You will see the dimensions of the selected area continuously as you drag the mouse. If no rectangle has been specified by mouse dragging, it prompts for the upper-left and lower-right corner coordinates. The coordinates are bound between 0 (the minimum *X* and *Y* at the top-left corner) and the size of the current image. Due to a requirement of the MS Windows memory management, it is recommended that the width (*X)* of the selected rectangular area is a multiple of 4. If your final *X* dimension is not a multiple of 4, WIM will increase it, or if it reaches the right bound of the image, decrease it. If the current image has geo-coefficients different from zero, they are transformed for the new image named <CutImg>. If the original image is in Mercator or other geographic projection, the <CutImg> will be in a recalculated Linear projection.

Save As...

Writes the current image buffer to a file. You can select between a number of file types.

Image

Saves a plain (unformatted) binary file. The size of the file will be the number of rows times the number of columns times the number of bytes per pixel. The number of bytes per pixel in the saved image file will equal the number of bytes per pixel of the image buffer, i.e. 1, 2 or 4. The sequence of the pixels saved is from the top-left corner row-wise to the bottom-right corner.
In case of a 2-byte-per-pixel image buffer, the least significant byte is the first and the most significant byte the second. Due to different floating-point formats used on different computer platforms, float images saved with WIM may not be directly readable on other platforms without conversion.

2-Byte Int

Writes the current image buffer (either 1 or 2 bytes per pixel) to a 2-byte-per-pixel file. For each pixel the first byte is the least significant (LSB) and the second byte the most significant (MSB) byte. If the current image buffer has 1 byte per pixel, a zero byte will be added as the most significant byte to each pixel.

Compressed (RLE) – obsolete!

Saves the current 1-byte-per-pixel image with RLE (run-length) encoding compression into a file. RLE-compression is not very effective for fine-grained images. However, for images with large areas of constant values (e.g. due to saturation by clouds, errors, etc.) it can save a considerable amount of disk space for storing large images. RLE-encoding is loss-less, i.e. after decoding the saved compressed file you restore exactly the same image. You can easily check it by saving an image as *Compressed (RLE)*, reading it back with *File - Open... Compressed (RLE),* and calculating the difference (*Multi - Difference*). You should get a black image consisting of plain zeros. Check the maximum difference with *View - Set Colors* as the Difference operation automatically includes *View - Lut Stretch.* Both the *Start* and *End* colors should be zero that prove that the maximum difference between the corresponding pixel values is zero.

Google Earth KML (*.kml)

Google Earth KMZ (*.kmz)

Writes the current image in the format compatible for viewing in Google Earth (http://www.google.com/earth). In the KML format the actual image (in the PNG format) is saved separately from the KML text file whereas in the KMZ format both the compressed image (in the PNG format) the KML file are embedded into a single KMZ file. You can also view the current image directly in Google Earth with Geo – View in Google Earth ( icon in the toolbar) or in Google Maps with Geo – View in Google Maps ( icon in the toolbar).

Erdas/Lan – obsolete!

Writes a set of 1-byte-per-pixel image buffers to a file in the ERDAS LAN format. (ERDAS is a popular image-processing program by Erdas, Inc.). The ERDAS LAN format has a 128-byte header followed by image data in band-interleaved-by-line (BIL) format. The first 4 bytes of the header are *HEAD*.

HDF (SDS)

Writes a set of image buffers as Scientific Data Sets to a file in the HDF format (see File - Open... HDF file .) Note the *Save as compressed* option in *Settings - HDF Options*. With compression the file size can be reduced many times while the file extension and usage remain the same.
Auxiliary data in the form of global or local attributes (see *View* – Attributes) are also saved into the new HDF file. This allows to save, e.g. the scaling settings and projection information when saving into a new file. You have to select one image buffers to be saved into the HDF file. Please note that the image data is internally compressed if the checkbox "*Save as compressed*" is checked in *Settings - HDF Options*. Compression is handled internally by the

software when reading the HDF file. For typical images with large areas of empty or uniform data the compression ratio is often 10-15 times and savings in disk space are drastic. For images with variable pixel values the compression is less than two times.
When creating new images, the attributes are generally copied from the source image to the new image. However, sometimes some attributes need to be changes, e.g. those related to Value Scaling if the scaling was changed, with the *Transf – Convert* operation. Starting with WIM 5.37 this has been fixed so that the appropriate new scaling attributes (*Scaling* type=*Linear* or *Logarithmic*, *Slope*, *Intercept*) are saved with the new image.
The Color Scaling attributes (*WIM Color Range*: *Min*, *Max*) are also saved.
Another problem is that when you manually set the projection type and then save the image, WIM will not save the newly set projection type. The idea is that WIM saves all the HDF attributes but manually setting projection does not change the attributes. The solution is to manually create a projection attribute that will be saved and retrieved when the file is read the next time. For example, when setting the projection manually to Global Equal Angle you need to create a new attribute "*Map Projection*" and set it to "*Global Equal Angle*".

HDF with Lat/Lon arrays

Writes an image into HDF file (see previous section) together with separate *Latitude* and *Longitude* arrays. Note: if your image has a projection (i.e. is mapped) then you DO NOT NEED to save in this format. Creating and storing Float32 arrays of latitude and Longitude just makes the file many times bigger and wastes both time and space. For each pixel a corresponding *Latitude* and *Longitude* values are recorded. Having separate *Latitude* and *Longitude* arrays for every pixel is the most versatile form of projection, however, it is also the **slowest** to use in generating coastlines or other geo-referenced objects. Therefore, if possible, saving data in this file type **should be avoided**. A single set of *Latitude* and *Longitude* arrays is saved when more than one image is saved to the file. All the images to be saved into a file should be of the same size. Note the *Save as compressed* option in *Settings - HDF Options*. With compression the file size can be reduced many times.

Overlay

Creates a 1-bit-per-pixel overlay from the current 1-byte-per-pixel image buffer and writes it to a binary file. The pixels different from zero will get a bit value of 1, while zero pixels remain zero. NB! In order to do the 8 byte to 1 byte compression, the number of pixels in a row has to be a multiple of 8. It is recommended to make overlays only from images with 8N pixels per line. It has to be noted that while 1-bit-per-pixel overlays provide 8 times compression for binarized images, RLE-encoding (see *File* - Save As - Compressed (RLE)) gives even better compression rates while keeping all the 8 bits per pixel.

Monochrome Bitmap

16 Color Bitmap

256 Color Bitmap

24 Bit Bitmap

Saves the bitmap from the current image to a bitmap file with a specified number of colors (type of bitmap): 2, 16, 256, or 16.7 million (24 bits). It is usually sufficient to have 256 colors in the bitmaps as 24-bit bitmaps take about 3 times the space of 256-color bitmaps.

ASCII

Writes pixel values of the currently selected area (rectangle or line) or the whole image as ASCII numbers in a file. The format of the recorded ASCII numbers depends on the *Value Scaling* option (*see View - Settings - General* - Value Scaling), i.e. it is a 3-character integer separated with a space for no value scaling and a floating point number for the scaled pixel values. This function is useful for transporting data to other software packages, e.g. for additional statistics, plotting, etc. As images usually contain a large number of pixels, ASCII files of integer or float numbers may become quite large (approximately 4 times of the image file size). If *Line* is currently selected Area of Interest (*see View - Settings - General* - Area of Interest), the pixel values along the specified line are recorded but not the *X* and *Y* coordinates of the line (cf. Line Save).

Lat, Lon, Value ASCII (*.dat)

Writes the image pixels as a series of triplets of *Latitude, Longitude, Value* in ASCII text characters. If an area of interest (Rectangle) is selected then only pixels in that area of interest are saved. You can choose whether to save the *Latitude* first, followed by *Longitude* and the *Value* or *Longitude* first, followed by *Latitude* and the *Value* depending on the "*Lat first*" checkbox in *View – Settings - Misc – Lat, Lon, Value Format*. The default formatting string is specified in *View – Settings - Misc - Lat, Lon, Value Format* as *%6.4f %6.4f %6.4f\n* and can be edited to suit your data. The meaning of this string is to use 6 characters, including 4 characters after the decimal point for each of the three values. The "\n" character means end of line and a tab character could be inserted as "\t". Be careful when editing the string as WIM may become unstable with certain settings. If the image has no projection information then the image coordinates (*X* and *Y*) will be used instead of the *Longitude* and *Latitude*. If values from more than one image are saved then the last number format is used for all of them. This file format was introduced for easy transfer of image data from WIM to various GIS packages. Be warned that this format uses a lot of disk space. For example, if a BYTE image would be save in the *Lat, Lon, Value* format using the default formatting string the size of the file would increase 21 times as each BYTE will explode to 21 bytes (6 for each of the three values plus 3 separators).
The same function is used for a rectangular area of interest (the Rectangle object) when selecting a *Rectangle* in the *Vector Objects* table and clicking on the Save *Lat, Lon, Value* button. Using *Geo–Get Vector Objects* – *Rectangle* makes it convenient to fix the corner points of one or more

rectangular areas of interest and save the values of these rectangles as *Latitude, Longitude, Value* triplets from a series of images.

GIF

saves the current image (bitmap) as GIF.

JPEG

saves the current image (bitmap) as JPEG.

PNG

saves the current image (bitmap) as PNG (Portable Network Graphics) file.

TIFF

Saves the current image buffer with the current color lookup table in a TIFF (Tag Image File Format) file that is portable between different hardware platforms and operating systems. Only 1-byte-per-pixel image buffers are supported by transforming to TIFF 6.0 Class P format (palette color images). If the image is in the *Global Equal Angle* projection then GeoTIFF tags will be added to the saved TIFF file according to the GeoTIFF format that is a common format in the Geographic Information Systems (GIS). Adding GeoTIFF tags preserves the geo-registration of the image when read with GeoTIFF-compatible software. GeoTIFF tags for other projections may be added in the next versions of WIM.

Close

Closes the current image buffer (i.e. frees up the allocated memory) and discards the corresponding bitmap from the screen. The same can be completed by closing the image window or by selecting the image from the List of Images box and choosing Delete.

Lookup Table

Allows to read (load) and write (save) lookup table (LUT) files. LUT is used to associate certain colors with certain pixel values.

Load LUT...

Loads a custom look-up table (LUT) for color coding from a file and sets the palette flag to *Custom* (see *View - Settings - General -* Palette). The default custom LUT (until another custom LUT is loaded) is the grayscale. The LUT file should be an ASCII file with the following structure:

```
  0   r0   g0   b0
  1   r1   g1   b1
  2   r2   g2   b2
.................
.................
255 r255 g255 b255
```

Each line should end with a Carriage Return (CR) and Line Feed (LF) character which is standard in DOS/Windows text files. In UNIX ASCII text files end with a single LF character. Here the first column is just the sequence number for your convenience and is not used by WIM. The next columns specify, respectively, the amounts of red, green and blue (numbers from 0 to 255) in the consecutive color. In case of a grayscale LUT, the red, green and blue components are equal. 256 rows are expected. If less than 255 color triplets are read from the file, the missing values are assumed to be white (255 255 255). In order to use the specified LUT in unmodified form, set the *Start* and *End* pixel values to 0 and 255, respectively, with *View - Set Colors...*.

You can create a sample LUT file corresponding to the default *Rainbow* palette by using *File* - Save LUT... (you may want to name it *rainbow.lut*). The LUT in *spectrum.lut* is another smoothly changing color palette. You can check how the red-green-blue components look like with Edit – LUT Edit. A popular LUT file for chlorophyll and other images (*chl.lut*) is included. Another sample LUT that is included with WIM is the one used by the IGBP Global Land Cover Characterization program. It has specific colors for 17 different land cover types but the rest of the LUT (colors 18 to 256) are unspecified and filled up by white (255-255-255) for WIM. See http://edcdaac.usgs.gov/glcc/images/gif/igbp_lgnd.jpg for more information. You can create your own LUT files by modifying the current LUT and then saving with *File - Save LUT*).

Load LUT raw...

Loads a custom look-up table (LUT) for color coding from a file and sets the palette flag to *Custom* (see *View - Settings - General* - Palette). The default custom LUT (until another custom LUT is loaded) is the grayscale. The raw LUT file is a binary file of 768 bytes organized in the following order: 256 red values (8-bit integers), 256 green values, and 256 blue values. The raw LUT format is the same as the raw palette format of the Hierarchical Data Format (HDF) for 8-bit raster data. A sample raw LUT file included with WIM is *spectrum.raw* (provided with the HDF code and utilities) which is another continuous spectrum lookup table. A popular LUT file for chlorophyll and other images in the “raw” format (Chl.raw) is included. You can create your own raw LUT files by modifying the current LUT and then saving with *File - Save LUT raw*).

Save LUT...

Saves the current look-up table (LUT) to a LUT file. The LUT file is an ASCII file with the structure given under *File* - Load LUT. You can interactively edit the current LUT with *Edit* - LUT Edit.

Save LUT raw...

Saves the current look-up table (LUT) to a raw LUT file. The raw LUT file is a binary file of 3 times 256 bytes (consecutive 256 bytes for the Red, Green,

and Blue components, respectively). The raw LUT format is the same as the raw palette format of HDF (for 8-bit raster data). You can interactively edit the current LUT with Edit - LUT Edit.

Save Info – obsolete!

Saves the current image size and optionally geographic projection parameters to an info file corresponding to the current image. The info file is **not** used for image file formats that have all the necessary attributes in the image file itself (CoastWatch, HDF). If the info file exists, its contents will be overwritten; otherwise a new file will be created. The name of the info file is formed from the current image name with the extension of **.inf*. Please observe that during image operations, image buffers in memory get different names; some of these are not acceptable as DOS/Windows filenames. An error message will appear if you try to save a corresponding info file. If you want to save the info file for one of these buffers, first rename the buffer or write the image buffer to a file (*File* - Save_As.), reload that file (*File - Open...*) and then save the info file.
The simplest format of an info file is the following:

```
Xsize ysize
```

In case of *Linear* (rectangular) latitude/longitude projection, the first line has it's geo-conversion coefficients:

```
Xsize ysize lon_top_left dlon lat_top_left lat
```

e.g., the US West coast CZCS image (*calchl81.img*) has in its info-file:

```
512  512  -140.0  0.07  55.0  -0.07
Pigment
```

The top-left corner has coordinates 140 W (negative !) and 55 N. The pixel increments of longitude and latitude are 0.07 degree/pixel. Please note the that longitude parameters precede the latitude parameters.

On the next line the current pixel value scaling will be saved (see *(View - Settings -* Value Scaling*): x10*, *x100*, *Pigment*, ...

In case of projections other than the linear, the name of the projection should be indicated on the following line. Only the first letter of the projections name (either capital or small) is used. The available projections are: *Mercator*, *Polar Stereographic*, *Transverse Mercator*, *Conic*, *Albers equal area*, *Equidistant Cylindrical, n3a, n3b, s3a, s3b*.
The last 4 are actually specific forms of the polar stereographic projection used for the SSM/I products (see Chapter 8).
The next line should have the latitude and longitude of the center of the image and the latitude range in degrees (decimals).
Two reference parameters are needed for the *Mercator, Polar Stereo* and *TV Mercator* (Transverse Mercator) projections and four parameters for the *Conic*, *Albers* and *Equidistant cylindrical* projections. They should be on the line following the line with the center latitude and longitude.

A sample info file for an image in the *Mercator* projection and no *Value Scaling (Pixel value*) looks like that:

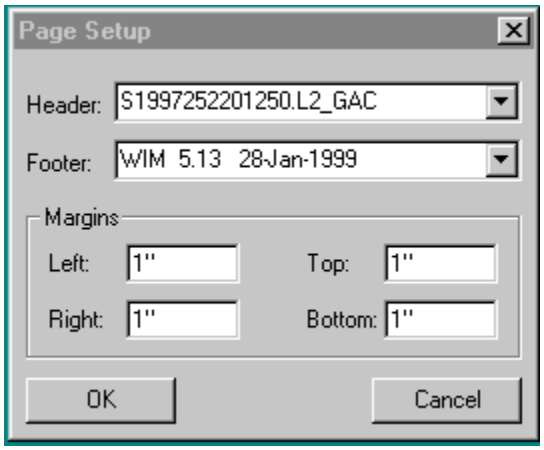
```
256 256  -119.248047  0.009727 34.853882 -0.008081
Value
Mercator
34.0  -118.000000 2.500000
0.000000 0.000000 0.0 0.0
```

Sample info-files for images in the Conic projection are given by *s_calif1.inf* and *s_calif2.inf*.
The geometric projections follow the information by the US Geological Survey (Snyder, 1982) and are compatible with the CCAR navigate program (Baldwin and Emery, 1993; Emery, 1995).

It has to be noted that with the increasing popularity of HDF file format the need for info-files has greatly diminished and these functions are not developed further.

Page Setup

Allows to adjust the size of the image to be printed as well as to add a header and footer to the printout.



Print

Prints the current image (bitmap) to one of the installed printers. See also Page Setup and Print Preview.

Print Preview

Shows the print preview of the current image.

Print Setup

Invokes the Windows™ printer setup dialog box.

Exit

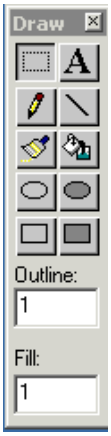
Exits WIM.

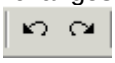
6.2 Edit - Edit Operations

Copy

Copies the currently selected image (bitmap) area or the whole current bitmap (if no area is selected) to the Clipboard. (Clipboard is a common memory pool in Windows™ used for transferring data between applications). Other Windows™ programs (e.g. Microsoft® Paint) can then paste the copied area from the Clipboard.

Draw



Allows to edit the image buffer with a versatile set of tools familiar from popular drawing programs. In contrast to the drawing programs which can only edit bitmaps, the *Edit - Draw* functions actually edit the image buffer and then update the bitmap for viewing. The set of tools includes the familiar tools like **text** (the letter "A") for annotations, **pencil** for free-form line drawing, **line** for straight lines, **brush** for painting, **paint-can** for filling with color, open and filled **ellipse**, open and filled **rectangle**. You can specify separate numerical values for the outline and for the filled area. The rectangular tool in the top left corner **disables** the drawing action and gives the left mouse button back to specifying the *Area of Interest* (either a rectangle or a line - *see View - Settings - General -* Area of Interest*).* Drawing is limited to Byte and Integer image buffers (the values will be wrong for Float image buffers). Starting from version 6.41 you can **undo** the changes and moving back and forth up to 16 steps back using the following  icons on the Toolbar.

If the background image is a RGB image then you can edit the image with a set of 3 values (for each of the Red, Green and Blue components). You can convert an image to RGB image with Transf – Convert to 24bbp (RGB).The color selection is done with the dialogs shown below.

LUT Edit

Performs interactive editing of the Red, Green and Blue components of the LUT (Lookup table) in memory. The default LUT is the grayscale palette. Optionally allows to load a LUT file from disk or to save the current LUT into a file. The Palette type is automatically changed to *Custom*. Each of the Red, Green and Blue LUT components can be modified either by the scrollbars or by mouse movements. The resulting bitmap can be viewed by selecting *Preview*. To test the operation, check out *spectrum.lut* after loading it with *File - Lookup Table -* Load LUT.

Scale to Clipboard

Copies the current color table to the Clipboard as a bitmap. You can then either save it as a bitmap or paste it directly into another program for combining your image with the corresponding color scale. If *Value Scaling* options have been selected (*View - Settings - General -* Value Scaling), real geophysical values are shown (e.g. the *CZCS-Pigment* concentration in mg/m^3 or temperature in C º), otherwise they are the pixel values. It is essentially the same operation as *Examine* - Color - Table plus the transfer of the result to the Clipboard.

6.3 View - View Options

Toolbar

Show or hide the toolbar.

Status Bar

Show or hide the status bar.

Image List

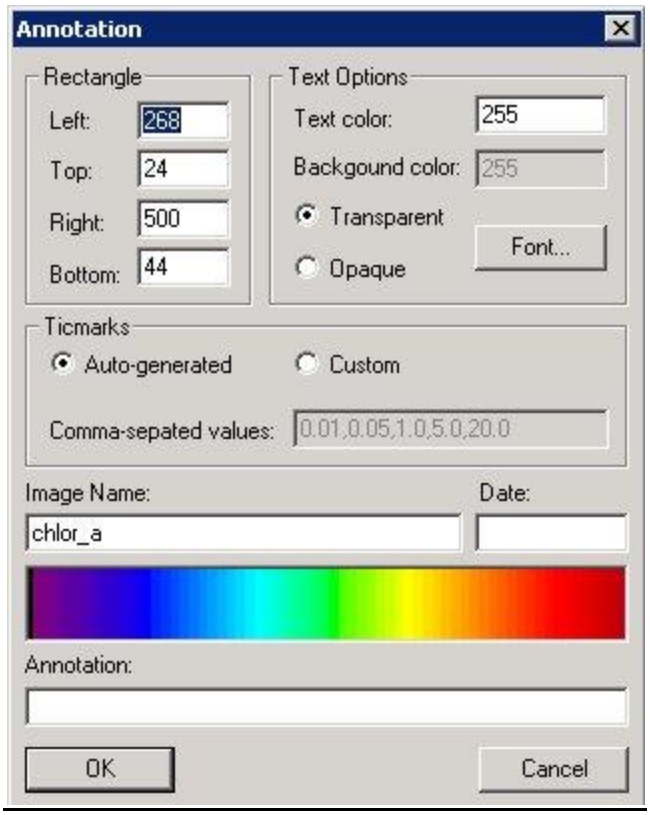
Show or hide the List of Images dialog box.

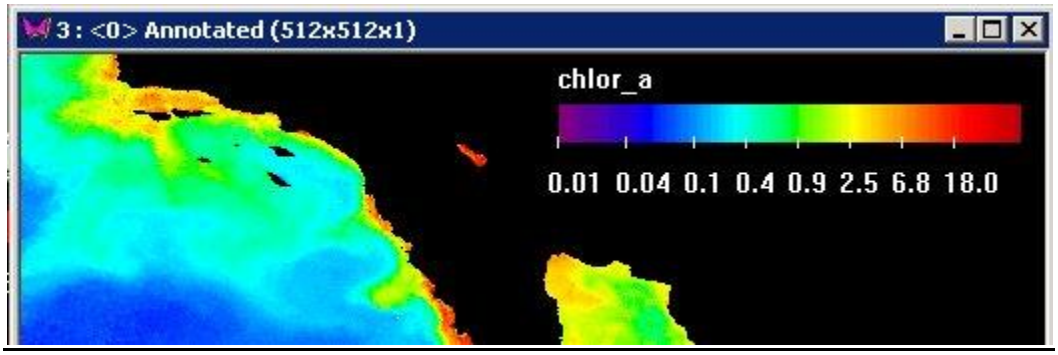
Zoom

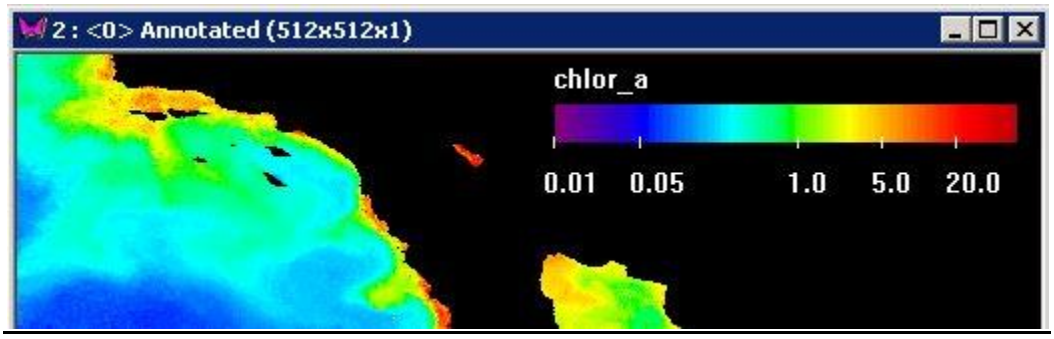
Allows to zoom the image bitmap at up to 64X magnification. This zoom operation acts at the bitmap level whereas the Transf - Zoom actually produces a zoomed image buffer.

Annotate

Writes a horizontal color scale at selected location into the image buffer, annotates with image name, title, etc. The values on the color scale depend on the current (*View - Settings - General)* Value Scaling. Please note that the easiest way of selecting a place for the annotation is by **mouse**: just **select a rectangular area with the left button** where you want the color bar to be, then select *View-Annotate* and the color bar will be written into the selected area. After you find a suitable size and location, you can put the annotation into **exactly the same location** on other images by **NOT selecting an area with the mouse** before using *View-Annotate* from the menu. The last location of the color scale is always saved for future calls to the function. This allows to apply consistent annotation to a series of images. A copy of the image is automatically opened with the annotation. You can easily make another try to find the most appealing size and location for the annotation.
A related operation adds a simple vertical color scale can be added to the left side of the image with Add Color Scale.
The tic-marks are either generated automatically or can be specified one by one in the text field *Comma-separated values*. In the examples below the auto-generated set of tic-marks are: 0.01, 0.04, 0.1, 0.4, 0.9, 2.5, 6.8, 18.0. By specifying tic-marks 0.01, 0.05, 1.0, 5.0 and 20.0 only those will be used in the annotation.


Annotation
Rectangle
Left: 268
Top: 24
Right: 500
Bottom: 44
Text Options
Text color: 255
Backgound color: 255
Transparent
Opaque
Font...
Ticmarks
Auto-generated
Custom
Comma-sepated values: 0.01,0.05,1.0,5.0,20.0
Image Name:
chlor_a
Date:
Annotation:
OK
Cancel


3 : <0> Annotated (512x512x1)
chlor_a
0.01 0.04 0.1 0.4 0.9 2.5 6.8 18.0


2 : <0> Annotated (512x512x1)
chlor_a
0.01 0.05 1.0 5.0 20.0

Attributes

Allows to view, edit, delete, add and copy to another image buffer of ancillary information describing the current image (e.g. scaling parameters) and global attributes describing the set of images (e.g. the satellite pass, date, receiving station, processing version). Attributes are commonly used in HDF. When creating your own HDF files with *File* - Save as .. HDF it is very convenient to add additional information with this function. These attributes are retrieved at a later time when reading the HDF file. Global attributes are common to all images in the same HDF file. Local attributes describe individual images. To access the local attributes of a particular image, click on the image and the select *View - Attributes*.

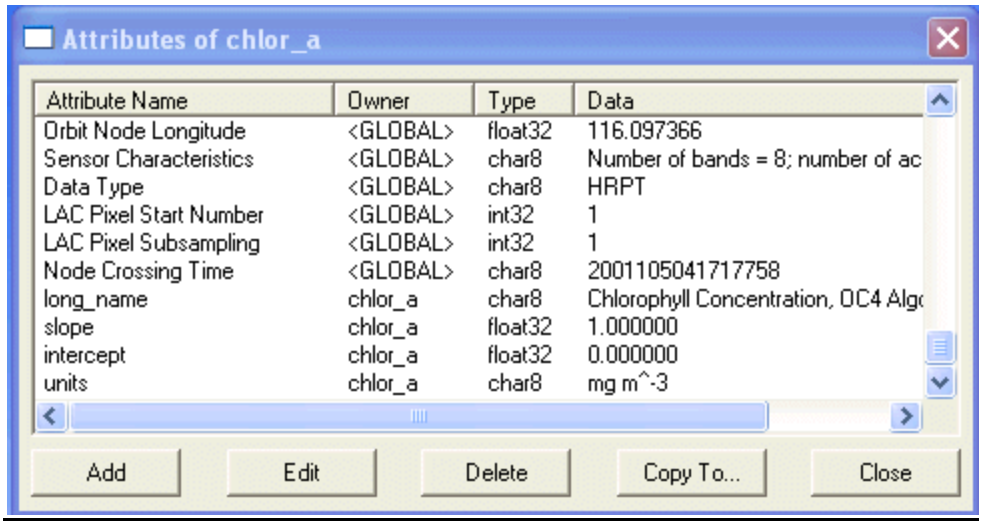
For NAVOCEANO images the attributes are generated by WIM from the ancillary data in the NAVOCEANO datasets.

For CoastWatch files the Attributes function is replaced with showing the CoastWatch header data that has similar information but is not modifiable by the user . Projection parameters and a few other characteristics of the image (NOAA satellite number, year, month, day, hour, minute) are also read from the CCAR image header (Baldwin and Emery, 1993; Emery, 1995). Correct time of the image is needed for making the solar correction (*Transf* - Solar Correction) and calculating the turbidity index with Multi - Turbidity *(ch1, ch2)*. When saving a CoastWatch image as HDF, the CoastWatch header data is automatically converted to HDF attributes; these attributes (e.g. the geo-referencing attributes specifying the projection) are then used by WIM when the HDF file is read.

The “Copy TO…” function makes it possible to selectively copy attributes from one image buffer to another. This useful, for example, when remapping an image to the projection of another image and creating a new remapped image. The problem of which attributes to get from the source image and which attributes from the target projection image is not easily solved automatically.

For Level 1B files the Attributes function is replaced with the diagnostic log of processing of the Level-1B file.

The following example shows the global and local attributes of a SeaWiFS Level-2 image.



Settings

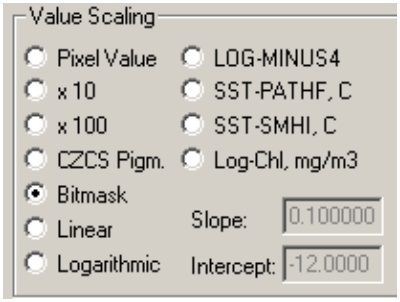
Allows to view and alter the current settings. As this function is used very often, it has a shortcut to it on the toolbar.
It has the following sections (tabs): *General, Extensions, Projection, HDF Options, Misc*.

General:

Image Size: ***Width*** (*DX*), ***Height*** *(DY)* and type, i.e. the number of bytes per pixel and either float or integer type;

2-byte and 4-byte per pixel images can be either ***Signed*** or ***Unsigned***;

Color Scaling: ***Min*** *value* and ***Max*** *value*.
Up to 256 different colors are used by WIM to display images. *Color Scaling* sets the *Min and Max* values of a range that is linearly stretched into 256 color bins. Unsigned 1-byte-per-pixel images can have only 256 different values; therefore this function is not needed for them. It is needed for buffers of 2 or 4-byte integer and float numbers. After you set the *Min* and *Max* values, the subsequent color stretching (*View – Set Colors* or the Toolbar color icon) only applies to the range between those *Min* and *Max* values. These *Min* and *Max* values are saved as *WIM Color Range* attributes with Save as .. HDF.



Value Scaling:

Real life image values (geophysical values) are seldom confined to integers provided by the 1, 2 or 4 byte per pixel integers, therefore, value scaling is used to convert geophysical values into pixel values. WIM has both predefined and generic scaling options. Generic scaling can be either *Linear* or *Logarithmic*. The *Base* in the *Logarithmic* scaling is always assumed to be 10.0.

Both *Linear* and *Logarithmic* make use of the *Slope* and *Intercept* values.

These scaling type (*Linear* or *Logarithmic*), *Slope* and *Intercept* values are saved as attributes with Save as .. HDF.

A simple predefined scaling option is *x 10* which simply means that the pixel values have been derived by multiplying the geophysical values with 10 (and rounding to the nearest integer). E.g., a temperature range 0.0 to 25.5 C is naturally converted to unsigned byte values by multiplying by 10 (*x 10* value scaling). In that case a pixel value of 250 would correspond to temperature of 25.0. However, in order to compress more information to an 8-bit pixel value and still use the simple format of 1 byte images, more sophisticated coding can be used. WIM uses 2 predefined nonlinear (Logarithmic) value scalings: *CZCS-Pigment* and *Chlorophyll* are both suitable for phytoplankton pigment concentrations that often have log-normal distribution.

A scaling of *Pixel Value* means no value scaling.

x 10 means that the geophysical values have been multiplied by a factor of 10 to get pixel values.

x 10 means that the geophysical values have been multiplied by a factor of 100 to get pixel values.

SST-PATHF, C This is the sea-surface temperature (SST) coding used in the NOAA/NASA Pathfinder Sea Surface Temperature (MCSST) data by the Physical Oceanography Distributed Active Archive Center at the Jet Propulsion Laboratory (http://podaac.jpl.nasa.gov/sst/). SST is calculated according to the following equation:

```
SST (°C) = 0.15 * Pix_Value -3.0.
```

SST-SMHI, C This is a Sea-Surface Temperature coding used by the Swedish Meteorological and Hydrological Institute (http://www.smhi.se). The pixel values are assumed to be representing temperature in ºC with the following transformation:

```
Temp = ((255 - Pix_Value) * 30) / 255 - 3
Pix_Value = 255 - (Temp + 3) * 255 / 30
```

This allows having values between -3 and +30 °C and at the same time reverses the order of pixel values, i.e. if cold was associated with low values before, it has high values after the transformation. The idea to reverse the

order of pixel values is to have cold temperatures white and warm temperatures dark using a conventional grayscale coding.

CZCS Pigm. The pixel values are assumed to represent phytoplankton pigment (Chlorophyll-a plus phaeopigments) concentration (mg/m^3):

```
Pigment = 10**(0.012 x Pix_Value - 1.4)
Pix_Value = log10 (Pigment) + 1.4) / 0.012
```

This coding was often used with the *Coastal Zone Color Scanner* data. The pixel values in the sample image *calchl81.img* are scaled as pigment concentrations. You can view them (and use in various operations) when selecting *CZCS Pigm.* for that image. Note that the CZCS-compatible pigment image of the SeaWiFS Level 2 product is actually a 2-byte-per-pixel buffer and uses Linear *Value Scaling* with Slope = 0.001 and Intercept = 32.0.

Log-Chl The pixel values are assumed to represent phytoplankton chlorophyll-a concentration (mg/m^3):

```
chlor-a = 10**(0.015 x Pix_Value - 2.0)
```

With 1-byte-per-pixel pixel the approximate range covered is 0.01-67 mg/m^3. This is a special case of the generic Logarithmic scaling. You can also set the scaling manually by selecting *Logarithmic* with the respective *Slope* and *Intercept*.

LOG-MINUS4 This is similar to *Log-Chl* but is used for geophysical values that are much too low for the *Log-Chl* scaling. :

```
val = 10**(0.015 x Pix_Value - 4.0)
```

With 1-byte-per-pixel pixel the approximate range covered is 0.0001-0.668.This is a special case of the generic Logarithmic scaling.

With a generic ***Logarithmic*** *Value Scaling* the following power equation is used to calculate geophysical values from the scaled pixel values:

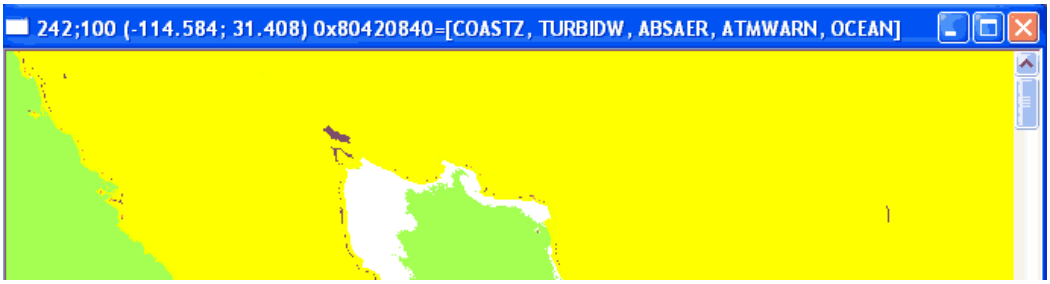
```
Real_Value = 10**(Slope x Pix_Value + Intercept)
```

Obviously, *Linear* scaling with *Slope* = 0.1 and *Intercept* = 0.0 corresponds to the predefined *x 10* value scaling. Likewise, the *Pigm., mg/m3* and *Log-Chl* are special cases of the generic *Logarithmic* scaling.

While dragging the mouse with the right button down you can normally see the pixel values. If various *Value Scaling* options have been selected, you will also see the calculated values according to the scaling algorithm (see also *Transf - Code/Decode*). The pixel values used in several *Examine* operations (*Examine - Color -* Table*, Examine –* Peeker*, Examine -* Profiles*, Examine -* Statistics*)* as well as the values saved to a *file (Examine -* Line Save , *Examine -* Point Save, *File -* Save As - ASCII) will be converted to a floating point number depending on the *Value Scaling* option.

When performing operations with multiple image buffers (e.g. *Multi* - Add 2 Images, *Multi* - Composite, *Multi* - Difference, *Multi* - Divide w. Image, *Multi* - Subtract Image), the decoding to a floating point value is performed before the operation. The *Value Scaling* of the resulting image buffer is set equal to the *Value Scaling* of the current image.

Bitmask is a specific form of scaling where each bit is representing certain preset flags or conditions. It is used, e.g. in the SeaWiFS and MODIS Level-2 data files. Values of the bitmask flags can be viewed on the source image by right-clicking on the image: the flags (bits) that are ON are shown in the window header. The names of the individual bit flags are taken from the attributes. For example, in case of SeaWiFS Level-2 flags attribute f01_flag is equal to ATMFAIL (atmospheric correction failure), attribute f02_name is LAND, attribute f03_name is BADANC (bad ancillary data), etc. The example below shows that a particular selected pixel has set flags COASTZ (coastal zone), TURBIDW (turbid water), ABSAER (absorbing aerosols), ATMWARN (atmospheric correction warning) and OCEAN. In fact, any BYTE, Int16 or Int32 image can be interpreted as a bitmask. If the name of a particular bit flag is not found it is designated as "unnamed". A special operation (*Transf – Bitmask*) exists for Bitmask images that will select areas where specific bits are either ON or OFF.



Area of Interest:
Line or *Rectangle*.
The *Area of Interest* feature toggles the type of selectable area between *Rectangle* and *Line*. If *Line* is the current type of area of interest, you can drag a straight line across the image in any direction. Histogram, Profiles and Statistics are then performed on the selected line. If *Rectangle* is selected, the corresponding operations are performed on the selected rectangle.

Palette:
Rainbow, Grayscale, Custom palette type.
Palette is essentially a lookup table that matches a pixel value to a triplet of values for the Red, Green and Blue components that are commonly used in computer color generation. In 256 and higher color modes you can select between different palettes. In 16-color mode (not common these days!) you can normally use only the system palette. The default palette (before loading anything else) is the built-in *Rainbow* palette. The *Grayscale* palette can be easily generated by the software. All other palettes (*Custom*) have to be read from a file, e.g. using the command *File* - Load_LUT… The *Rainbow* palette has been adapted from Prentice (1987) with slight modifications. It starts and ends with blue and shows a subjectively continuous color transformation.

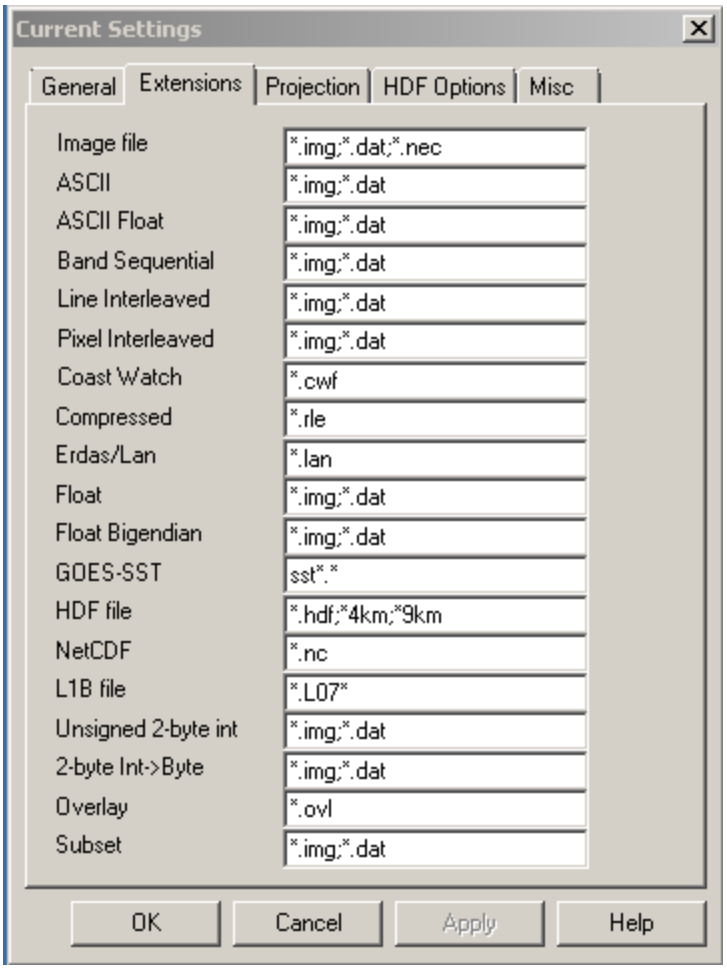
You can set up WIM so that another default palette is read from a specified file on disk whenever WIM starts. Use the *Settings - Misc – Use Default LUT* option for that. By default, WIM reads the default palette from a file *chl.lut* in the Wimsoft folder. This palette is good for many types of images and not just chlorophyll. More palettes are included with the WIM package. For example, two-tone (red and blue) palettes *anomaly.lut* and *anomaly7.lut* are useful for showing anomalies.
Each image has its own palette unless it's the built-in *Rainbow* type. When an image or a set of images is saved to a HDF file then each image retains its individual palette as an HDF attribute. When reading an HDF file saved with WIM the user can choose whether to use the current default palette (set in *Settings – Misc - Default LUT* and with the *Settings – HDF Options - Override LUT in HDF* checked) or the palette from the HDF file (unchecked *Override LUT in HDF*). *Settings - General* also shows the origin of the Custom palette (obtained either from a palette file or from an image).

Confirm Settings:
If the *Confirm Settings* checkbox is checked, WIM displays the *View - Settings* dialog box each time when reading an image without a corresponding info-file; if disabled, the dialog box is not shown. When checked, WIM prompts if projection parameters are to be taken from a CCAR image header (Baldwin and Emery, 1993; Emery, 1995), otherwise they are accepted automatically if found.

Restore Defaults:
Clicking on this button deletes the WIM registry settings, e.g. the WIM file extension settings, window locations, the locations of files used in the past, parameter values used with various operations, etc. A typical case when you need this is if you accidentally associate a file extension with a wrong file type (after double-clicking on a file and selecting WIM as the application to open that file).

Extensions:

Has default image file extensions for all image file types. You can change them to suit your file naming convention. WIM always remembers the last file type that it loaded, and suggest that as the default file type for a next load. The initial extensions are **.img, *.dat* for most image file types, **.ovl* for overlays, **.rle* for compressed files, **.hdf* for HDF files, etc. If modified, the new default extensions will be saved (in registry) for later WIM sessions. Typical extension settings are shown below:



Projection:

Satellite images of the Earth can be in various geometric projections. WIM has options for a number of projections.
Projection type is one of *WIM native projections*:

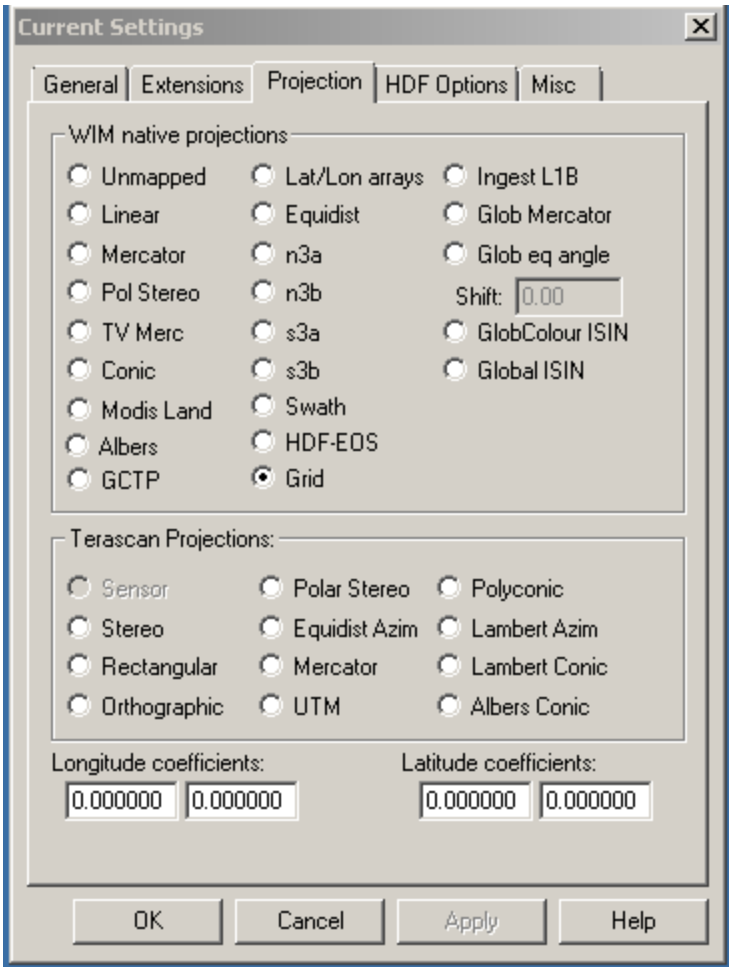
Unmapped, *Linear*, *Mercator*, *Polar Stereo*, *Transverse Mercator*; *Conic, Albers, Equidistant Cylindrical, n3a, n3b, s3a, s3b; Swath, HDF-EOS, Ingest L1B, Glob Mercator, Global equal angle with a Longitude shift, Lat/Lon arrays.*

or one of *Terascan* projections*:*

Stereo, Rectangular, Orthographic, Polar Stereo, Equidistal Azimuthal, Mercator, Universal Transverse Mercator (UTM), Polyconic, Lambert Azimuthal, Lambert Conic, Albers Conic.

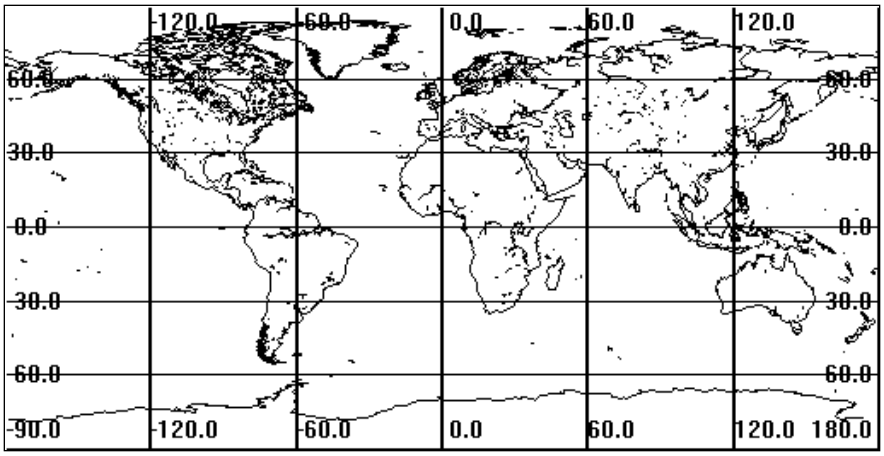
The GCTP family of mapping functions includes most common projections includes as a separate dynamic link library (gctp.dll) and derives from the

General Cartographic Transformation Package (GCTP) from the USGS National Mapping Division.

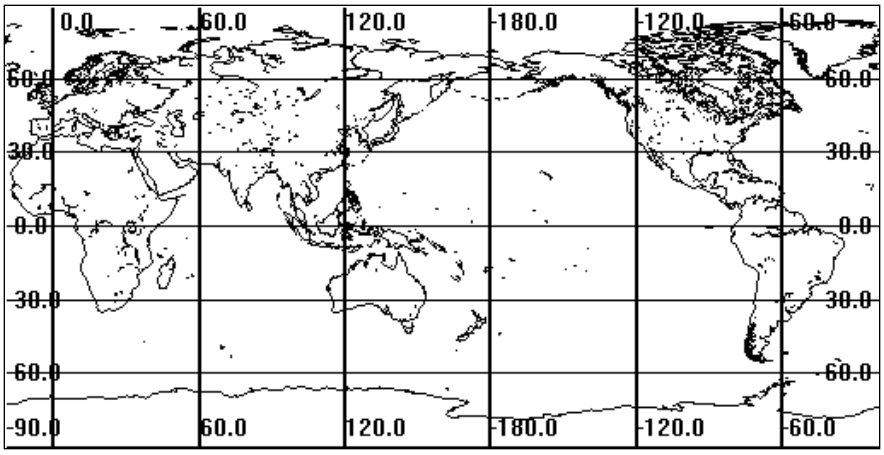


The *Global equal angle* projection is popular for Level-3 global images and is also known as the Plate-Carrée grid. It is a grid and can be used in WIM without any additional parameters. The only issue is the meridian value corresponding to the left side of the image. The default value is assumed to be –180 (e.g. SeaWiFS Level-3 and Pathfinder Sea Surface Temperature images) and the *Shift* value is equal to 0. For the MCSST and GLI Level-3 images the *Shift* is automatically set at 180 and the left side of the image corresponds to the zero meridian. For the OCTS Level-3 global images produced by NASDA you have to manually set *Shift* = -160 whereas the OCST Level-3 images produced by NASA no shift is necessary (i.e. it is 0). See the following figure on the influence of the *Shift* parameter.

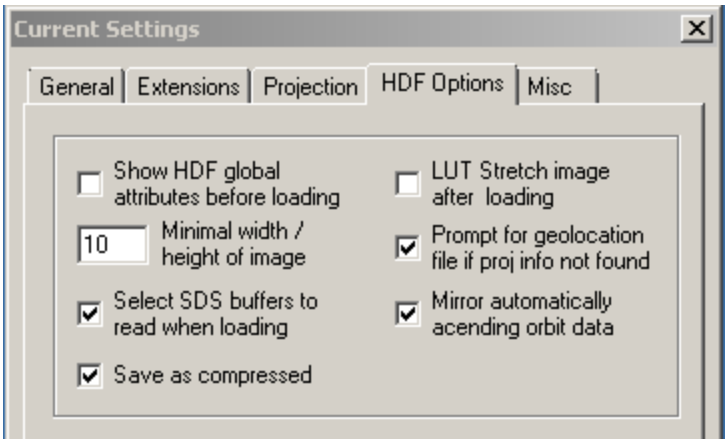
Shift = 0 =>



Shift = -160 =>



When saving the file as HDF file the Shift value will be recorded and will be set when the file is read again with WIM. It is obvious that the *Shift* value of –160 is convenient when looking at imagery covering both sides of the international dateline, e.g. the Pacific Ocean. With Shift = 0 the area of interest will be in two separate parts (see the figures at the top). Note that Level-3 in netCDF formats are provided in *Grid* projection which is in practice equivalent to the *Global equal angle* projection if using a constant longitude and latitude steps.



The projections *n3a, n3b, s3a, s3b* are specific implementations of the *Polar Stereographic* projection used with the SSM/I products delivered by NSIDC (see SSMI/Products*).*

The Swath projection is specific to the SeaWiFS Level-1A and Level-2 products and shows the image in the satellite view projection. SeaWiFS global mapped images are in the Global equal angle projection.

CoastWatch images can only be in Linear or Mercator projections. The simplest conversion between image coordinates and an area of Earth is accomplished by the *Linear* projection which is acceptable for small areas.

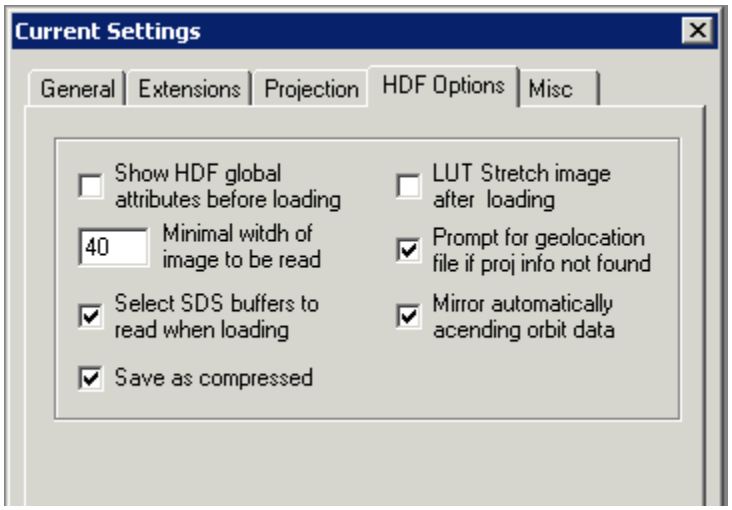
In case of a Linear projection the coordinate conversion between the video (screen) and geographic (longitude and latitude) is performed with two pairs of coefficients (*A, B* and *C, D*). The top left corner of any image is assumed to have video (screen) coordinates of X = 0; Y = 0 with X increasing to the right and Y increasing to the bottom. Longitude is assumed negative on the Western Hemisphere and latitude is assumed to be negative on the Southern Hemisphere.

```
Lon = A + B * X;    Lat = C + D * Y
```

where *X* and *Y* are the screen coordinates (pixel and line, respectively). The default coefficient values are zero. If at least one of them is different from zero, you will see the calculated longitude and latitude of the current pixel when you drag with the right button of the mouse.
See section 5 (Useful hints) how to find the linear geo-conversion coefficients for your image. Normally the coefficients are read from an info file (**.inf*) but you can change the coefficients manually in the *Settings - Projection* dialog box.

In case of geographic projections other than *Linear*, up to 4 extra reference variables are used. If any of the reference variables is different from zero, they are shown near the bottom of the *View - Settings* - Projection dialog box. *ref1* is a reference latitude, *ref2* is a reference longitude, *ref3* is another latitude and *ref4* is another longitude. Some projections do not require all the reference variables. The reference variables are read from the info-file (*see File* - Save Info). For more on projections, see *Geo* - Remap Proj.

HDF Options:



Show HDF global attributes before loading - if selected, shows *View* – *Attributes* before loading the images.

LUT Stretch HDF Images after loading *checkbox* - if selected, automatic LUT stretching between the minimum and maximum pixel values is performed after loading HDF SDS images (see *View* - LUT Stretch).

Minimal width of image to be read - WIM will discard arrays narrower than the specified number of pixels as images. HDF files often contain various arrays of numbers that are not supposed to be images but to provide other auxiliary information, e.g. calibration coefficients. These arrays are usually narrow, e.g. a certain number of calibration coefficients per scan line, and can be discarded from being displayed by WIM.

Prompt for geo-location file if proj info not found - if selected and WIM cannot recognize geo-location (projection) of the image – prompts for a separate geo-location file name.

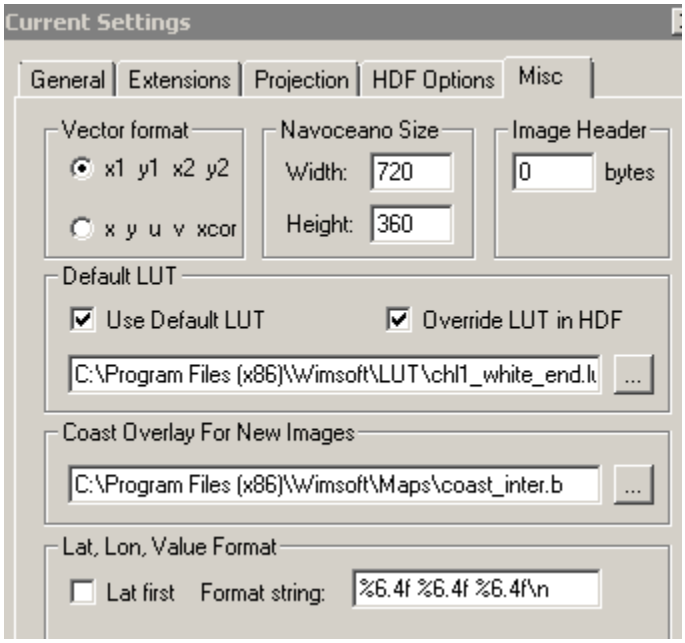
Select SDS buffers to read when loading - when selected, this option gives you a list of all image buffers (Scientific Data Sets) that can be selectively read into RAM. When not selected, all image buffers will be read into RAM.

Mirror automatically ascending orbit data - when this option is NOT selected, ascending orbit data will be upside down. When selected and the image is detected as ascending orbit, it is automatically mirrored.

Save as compressed - if selected, the image data inside the saved HDF file is compressed (with the (with the Lempel/Ziv-77 dictionary coder) that can reduce the file size by many times. For typical images with large areas of uniform pixel values (e.g., due to no data or clouds) the compression ratio is

10-15 times. For images with highly variable pixel values the compression ratio is less than two.

Miscellaneous (Misc):



Vector Format - *x1 y1 x2 y2* or *x y u v xcorr*.
Vector Format toggles between 2 different formats how vectors are represented in a file. *x1 y1 x2 y2* is the default and *x y u v xcorr* is an adaptation to the format produced by the *Multi* - Motion Detect function (see *File* - Get Vectors).

Default Navoceano Size:
The default Width (720 pixels) and Height (360 pixels) can be changed by the user. The image width and height specified here are also use when opening files in the *Lat, Lon, Value ASCII* format.

Image Header;
In order to skip a certain number of bytes from the beginning of a plain raster image file (e.g. those containing a header), you can specify the header length in bytes that will be skipped when reading image files.

Default LUT:

Use Default LUT – this allows to use a predefined palette file that will be read and used as the default palette for all images (instead of the *Rainbow* palette). Other palette files (included with WIM are *chl.lut*, anomaly.lut, and *spectrum.lut*) can be used instead. You can create your palette with Edit – LUT Edit, save it with *File - Save LUT* and then make it your Default palette.

Override LUT in HDF – this allows to ignore the palette in a HDF file and use the current default palette. Starting with WIM 6.28 an individual palette is saved with each image as an HDF attribute. Sometimes you may want o use a consistent palette for all images. This option allows to choose whether to use the palette from the HDF file or the default palette.

Coast Overlay for Default Images – this coastlines file is used in the *File – New* function to create and overlay coastlines. The default is *C:\Program Files (x86)\Wimsoft\maps\coast_inter.b*

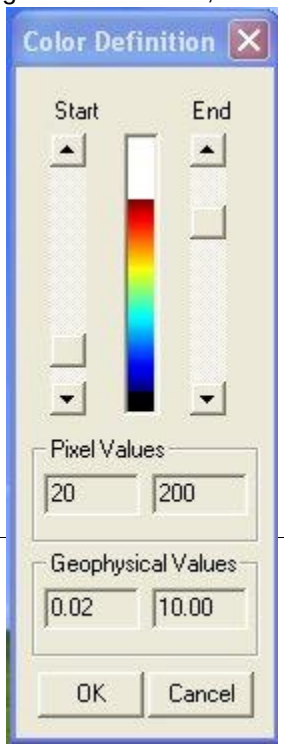
Lat, Lon, Value Format:

Lat first – this checkbox allows to choose if *Latitude* or *Longitude* comes first in triplets of *Latitude, Longitude, Value* in the format of *.pnt* or *.csv* files in *Geo-Get Vector Objects* and in saving files as *Lat, Lon, Value ASCII (*.dat)*.

Format string – this edit box allows to select the exact format of saving triplets of *Latitude, Longitude, Value*. The default formatting string is *%6.4f %6.4f %6.4f\n* and means that 6 characters, including 4 characters after the decimal point, are used for each of the three values. The "\n" character means *end of line* and a tab character could be inserted as "\t". Be careful when editing the string as WIM may become unstable with certain settings.

Set Colors...

Allows you to specify the coloring sequence (look-up table = LUT) by the *Start* pixel value and the *End* pixel value. Both parameters range from 0 to 255. If *Start* is smaller than *End*, all pixels less and equal to *Start* will get the first color in the palette (usually black), and all pixels greater and equal to *End* get the last color of the palette (usually white). The pixels with values between *Start* and *End* will be colored by a linear sequence of the available colors. Pixel values different from 0 but smaller than *Start* will get the second color in the LUT. Respectively, if *End* is less than 255, pixel values less than 255 but larger than *End* will get the second color from the end. If *Start* is greater than *End*, the opposite color sequence will be assigned. If you are running more than one color-intensive application simultaneously, Windows may run out of available colors and has to replace colors on inactive windows by temporally matching them to the colors of active windows. In 15/16- or 24-bit color modes the selection of colors is practically unlimited.



In case of a Byte image the values below the Start and End pixel values show the respective geophysical values. For example, in case of a Byte image with the logarithmic Chlorophyll scaling (*Log-Chl*) the Start and End pixel values of 20 and 200 correspond to Chl concentrations of 0.02 and 10.0 mg m^{-3} (see example on the left).

In case of 2 or 4 bytes per pixel (integer or float) the pixel values are first scaled with the Min and Max values from *View - Settings - General* –Color-Scaling to be between 0 and 255. After that the color coding is similar to the 1-byte-per-pixel images.

The *Start* and *End* values are saved as *WIM Color Stretch* attributes with Save as .. HDF.

Vector Objects

Shows the table of Vector Objects. The Vector Objects table consists of a number of the following vector objects: *Points*, *Transects*, *Rectangles*, *Labels, Drifter Tracks* and *Vectors* and the following buttons: *Add, Edit, Delete, Close, Save, Save Lat,Lon,Value, Show Profiles, Statistics*.

The ***Add*** button allows a manual creation of a vector object. Usually vector objects (except the *Label*) are imported with the *Geo - Get Vector Objects* function from simple ASCII text files or with the *Geo - Read Vector Objects from HDF* from *HDF* files (recorded previously with the ***Save*** button in this function). An ASCII text file with the longitudes and latitudes can be created with an editor (such as *Notepad*), exported from a spreadsheet program (like MS *Excel*) or created by the WIM function *Geo-Distance* or with *Examine-Point Save*. An example of a *Vector Objects* table is shown below.

Each image has its own set of vector objects. When another image is selected, the list of vector objects for that image is shown. An individual object can be selected by clicking on it after which it starts blinking on the image. This feature makes it easy to visualize and locate multiple objects on the image.

With the ***Edit*** button you can manually edit the selected vector object.

The ***Delete*** button deletes the selected vector object.

The ***Save*** button saves the selected object(s) in HDF file that can be read with the *Geo - Read Vector Objects from HDF* function.

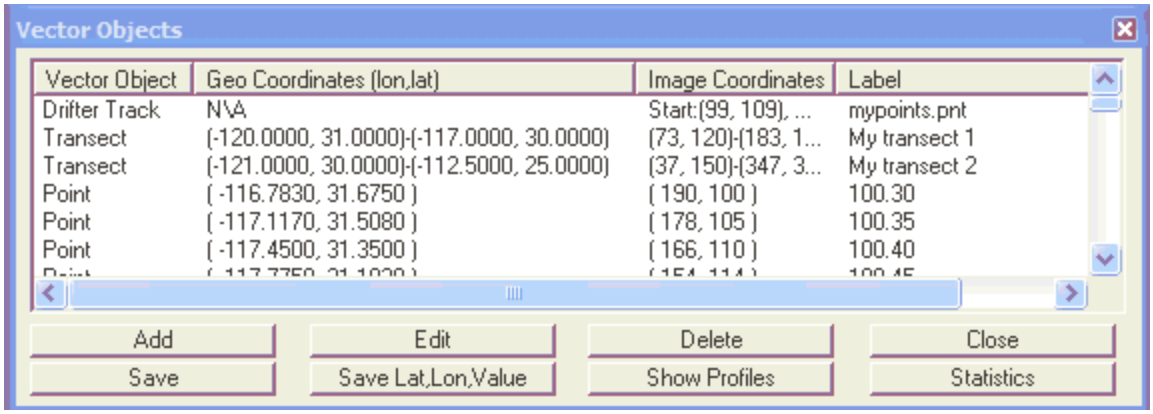
The ***Save Lat,Lon,Value*** button saves the selected object(s) in a text file.

The ***Show Profiles*** button shows the profile for a selected transect object.

The ***Statistics*** button shows statistics for each of the applicable object. The statistics for a Point is calculated for the window of 3 by 3 pixels centered at the point (pixel).

If you want to copy the set of vector objects from one image to all the other images in memory, select the source image (by clicking on it) and apply *Geo - Unify Vector Objects*. Now all the images in memory will have the same vector objects. This operation is used when you want the same vector operation to be performed on a set of images but don't want to read the same set of vector objects for each individual image. For example, you may want to pick the same transect on a series of images, extract it, get a series of these transects into MS *Excel* for plotting. You can read the transect file once, transfer the vector objects to other images and save each transect with the *Save Lat,Lon,Value* button to an ASCII text file. You can then read that text file into MS *Excel*.

When selecting a *Rectangle* object all the pixel values within the rectangle can be saved with the *Save Lat,Lon,Value* button. The result is like the menu function *File – Save as Lat, Lon, Value ASCII*. The formatting of this operation can be modified in *View – Settings – Misc – Lat, Lon, Value Format*.



Drifter Track is a Vector object that is similar to a sequence of *Point* objects but is treated as one set. The file formats of the *Drifter Track* is similar to the Point files: either the Point (*.pnt* or *.csv*) or NOAA drifter format (*.DAT). Each vector object needs to be in a separate file.

Text *Label* is a special type of vector object. It's primary purpose is to provide flexible annotation of the images. Using the *Add* function in the *Vector Objects* table it is possible to write text labels anywhere in the image so that these labels can be edited or removed without affecting the image. A related function *Edit-Draw-Text* is also available.

LUT Median

Stretches a linear color look-up table in the middle of the histogram leaving out the bottom and top distribution tails. This operation is useful when you want to exclude a few very low and very high outliers that would otherwise make the LUT Stretch operation insensitive to most of the pixels in the middle of the histogram. It is also useful if you want to focus the look-up table on a certain area - make a histogram of the area and run *LUT Median* with 0 exclusion. You must run *Examine* - Histogram - Calculate before using this

routine. Remember that the last histogram calculated is assumed when performing this operation.

LUT Stretch

In case of a 1-byte-per-pixel image finds the minimum (*Start*) and maximum (*End*) pixel values on the current image and makes a new linearly stretched color look-up table between them. This function has another useful application: finding the maximum and minimum values of a 1-byte-per-pixel image. Use *View* - Set Colors after it to check the minimum (*Start*) and maximum (*End*) values of the image.

In case of a 2-or 4-byte-per-pixel image (unsigned integer or float, respectively) the operation finds the minimum and maximum values and then scales the pixel values between them by setting the *Min value* and *Max value* (see *View* - *Settings* - *General* - Color Scaling). The *Start* and *End* values of color stretching are not changed. Note that the SeaWiFS Level-2 Chlorophyll-a and CZCS-pigment products are signed 2-byte integers and their typical *Min* and *Max* suitable typical for color coding are around -32000 and -29000, corresponding to concentration range 0–3 mg m^{-3}, respectively.

Loop Images

Plays a “movie loop” of the images in memory. It switches sequentially from the first image to the last and starts all over again. Select a suitable delay period (1...60 s) between switching to another image. This operation is useful for detecting changes between different images. Select it again to stop the loop.

Add Color Scale

Adds a vertical color scale to the left side of the image. A more versatile color scale (horizontal) can be added with View – Annotate.

6.4 Examine - Examine Operations

Color

Consists of the following options:

Table

Shows the current color table with the corresponding pixel values. If a specific *Value Scaling* option has been selected (*View - Settings - General - Value Scaling*), the displayed values are the calculated geophysical values (e.g., pigment concentration in mg/m^3 or temperatures in ºC), otherwise they are the pixel values.

Scan

Scans (animates) through the whole color palette. It shifts the colors in the 256-color palette one step every predefined number of milliseconds (10-1000) between the currently selected color range (*Start* and *End*). To end the color animation, select any menu. This operation is not available for 16-color (standard VGA) and high-color (more than 256 colors) modes.

Random

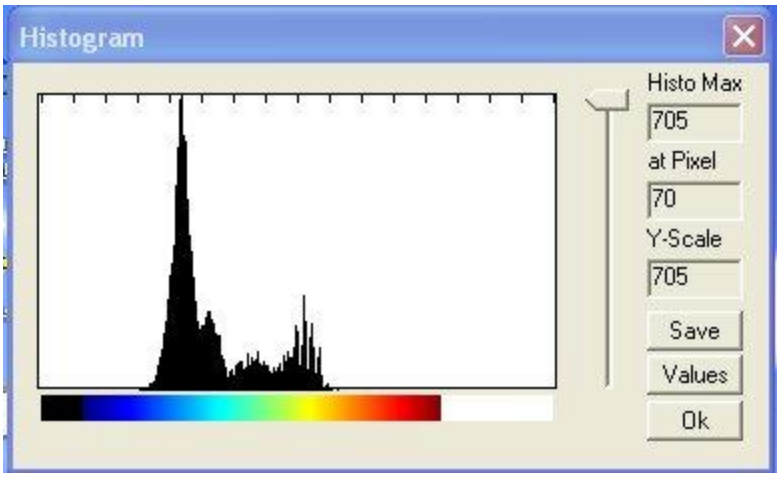
As Color Scan but generates random color palettes. Not available for other than 256-color modes. To end the color animation, select any menu.

Contour Lines

Creates a new image buffer with contour lines in a specified pixel value for a selected range of pixel values. To get a smoother-looking picture, the image is usually smoothed with a median filter before applying the contour-finding algorithm. The bigger kernel size of the filter produces stronger smoothing. The filtering option for 2- and 4-byte-per-pixel image buffers is available starting with WIM version 6.12. If *Copy source* is selected the image itself is copied to the new image buffer; otherwise, a blank image buffer is used to draw the contour lines. Having the contours on a blank image gives you more options, e.g. you may want to manually edit the contour lines with *Edit – Draw* before you overlay the contours on the image with *Multi – Overlay Image*. You have to select the *Start*, *End* and *Step* values for which the contour lines are drawn into the image buffer. The *Start*, *End* and *Step* contour values refer to image values after *Value Scaling*. If you want to select the contour lines according to pixel values (i.e. without *Value Scaling*), set the *Value Scaling* equal to *Pixel Value* in *View - Settings*. You can also select the pixel value with which the contour lines are drawn. The pixel value of 255 results in *white* contour lines for most color palettes (LUTs) with Byte images but not with 2- and 4-byte-per-pixel images. The background pixel value of the new image (if created) will always be 0.

Histogram

Consists of several histogram operations:



Calculate

Calculates histogram for the selected rectangle or line, or for the whole image, and shows it.
In case of 2 or 4 bytes per pixel (integer or float) the pixel values are first scaled with the Min and Max values from *View - Settings - General* –Color-Scaling to be between 0 and 255. The following histogram calculation is then similar to the 1-byte-per-pixel images.

Show

Shows the previously calculated histogram. Optionally switches to *Histogram - Save* or *Histogram - Values*. The vertical scale of the plot can be changed with the scrollbar. Values of the histogram maximum, the pixel value with highest frequency (maximum histogram) and the vertical scale are shown.

Save

Saves the current histogram in an ASCII file.

Thresh

Shows the pixel values of the distribution tails below which and above which the specified percentage of values are. E.g. by specifying 10% you will see the corresponding pixel values above which the top 10% and, respectively, the bottom 10% of the values are.

Values

Shows the numerical values of the previously calculated histogram in a table with eight values in a row.

Line Save

Writes the *X* and *Y* coordinates and the pixel values of the currently specified line (excluding the last pixel) as ASCII numbers into a file (each triplet on a new line). This function is only accessible if *Line* is selected by *View - Settings - General* - Area of Interest. A similar routine *File* - Save As - ASCII only records the pixel values without the coordinates. If image projection is set (for Linear projection any of the geo-conversion coefficients have to be

different from 0, see *View - Settings* - Projection), the *x* and *y* coordinates recorded in the file are the Longitude and Latitude, otherwise they are the screen (video) coordinates. If *Value Scaling*: *x 10, Pigment, Scaled* or *SST-…* has been selected in *View - Settings - General - Value Scaling*, the recorded values are saved in their respective units, otherwise they are the pixel values.

Peeker

Allows you to view (peek) the pixel values in a 5- by-5 block centered at the currently selected pixel (top left corner of the selected rectangle or the starting point of the line). The 5-by-5-pixel window is adjusted so that it would not get outside the image limits. If any of the longitude/latitude coefficients (*View - Settings* - Projection) is different from zero, the longitude and latitude values of the center pixel are also shown. The values displayed for each pixel depend on the *Value Scaling* parameter chosen for the current image (see *View - Settings - General - Value Scaling*). By pressing and holding down the right button of the mouse, it is possible to see the values continuously updated while you move the mouse around the image. Note: You can do this if you start dragging from the Peeker window and move over to the image window without releasing the right button. If you start on the image window you will get the normal current pixel position (*X, Y*) and value on the window top frame. You can even use more than one *Peeker* window by opening a new one from the menu while keeping the previous one. However, do not start too many *Peekers* as you may finally crash the program.
In case of a float (4 bytes per pixel) image buffer only 3 pixel values are shown per line.
View - Zoom can also be used for detailed examination of a small image area.

Point Save

Allows you to save pixel coordinates and values to a file. Clicking with the Right button of the mouse on an image pixel normally shows the pixel coordinates and value on the window's top frame. If *Point Save* has been selected, the coordinates and values are also recorded in an ASCII file each on a new line. If you select an existing file, it's contents will be lost before new data is recorded. The selected points are shown on the screen by a black line connecting them. Up to 256 points can be shown on screen, saving to the disk file can go on until there is space on the disk. Do not forget to deselect the Point Save option when you don't need it any more by selecting the menu option again. If the image projection is set (see *View - Settings* - Projection), the x and y coordinates recorded in the file are the Longitude and Latitude, otherwise they are the screen (video) coordinates. If *Value Scaling* has been selected (*View - Settings - General – Value Scaling*), the recorded values are in their respective units, otherwise they are the pixel values.

Profiles

Plots the currently selected line or rectangular area. In case of a line, the beginning of the line is positioned on the left and the end of the line on the right. In case of a rectangle, you may get either plots of rows (horizontal lines) or columns (vertical lines). If the height of the area is greater than the width, columns are plotted, otherwise the rows are plotted. In the case of the column plot, the left side on the plot corresponds to the top of the rectangular area. If *Value Scaling* has been selected (*View - Settings - General - Value Scaling*), the plotted values are in their respective units, otherwise they are pixel values. Optionally allows to save the values on the selected line or rectangle as ASCII integers in a file.

RGB Image

Uses three image bands for the Red, Green and Blue components, respectively, to construct a "true-color" image. This is a special type of Int32 image in memory. When right-clicking on the RGB image the three components (Red-Green-Blue) are shown. You need to stretch the Red, Green and Blue bands separately in order to create the desired result. The *Low* and *High* scrollbar levels for each band correspond, respectively, to the *Min* and *Max* values of the component pixel values between which the color scale is linearly stretched. They are similar to the *Start* and *End* values in *View* – *Set Colors*… The selected three sets of *Min* and *Max* values are saved in the computers registry and are used as default values next time this operation is used.
In order to be close to a "true-color" image the bands should roughly correspond to the red, green and blue spectral bands. For example, with SeaWiFS images the "true-color" images are usually constructed from bands 670 nm (Red), 555 nm (Green) and 412 nm (Blue). A sample image file *sw_l1a_mapped_byte.hdf* in the *Images\SeaWiFS\baja_2000_april* folder of the WIM CD (or online) can be used for testing this operation. Different features, such as land, turbid water, clear water, aircraft contrails, clouds can be separated by stretching the *Min* and *Max* values for the R, G, B components.

When applying this operation to 2-bpp and 4-bpp images the *Current Settings* – *Color Scaling* *Min* and *Max* need to be suitably set. You can try to use *View* – *LUT Stretch* on each band before applying this function.
High-color graphics modes (i.e., 15-16 or higher) are required for this function to work properly. The RGB image can also be saved as HDF. In this form the HDF image can be read again as a RGB (single Int32) image.

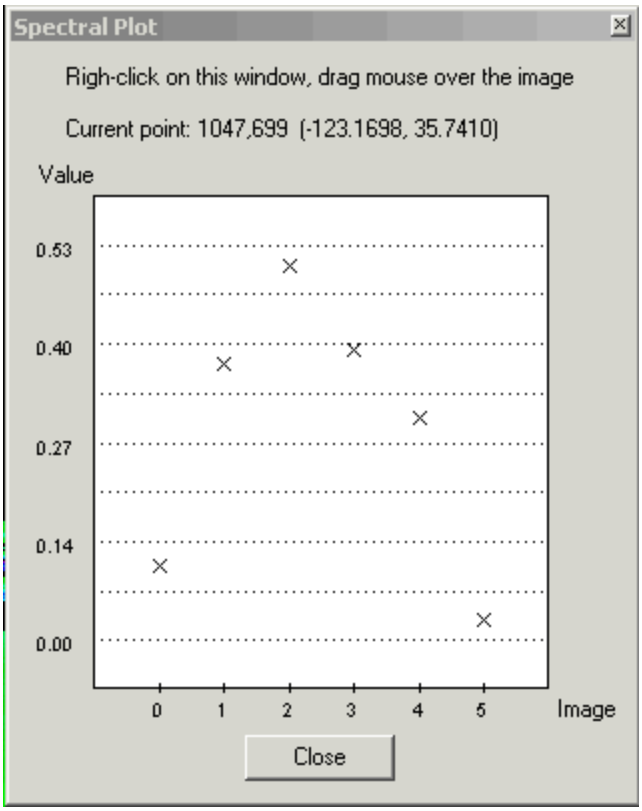
Finding the best color combination by shifting the *Low* and *High* scrollbars can be very slow for large images. You can first find the best color combination on a small rectangle. Just select a rectangle and select *Examine-RGB Image* from the menu. The RGB image is created quickly in the small, selected rectangle. You can now adjust the range values of the components. After clicking OK the selected range values will be propagated to the whole image. This is a very useful trick when working with large images.

Another useful tip concerns making coastlines and other overlays. A typical coastline overlay has pixel value 255 and that produces a blue coastline on a RGB image (think of it as the first = Blue component of a RGB image). To create a white overlay, you need to create a RGB image from the single overlay. A special function *Transf-Convert to 24bpp (RGB)* does exactly that. A more difficult ("manual") way of doing that is to select the overlay image and then *Examine-RGB Image*. Make sure that all 3 component images point to the sequence number of the overlay image. That makes all 3 components 255 and that produces a white RGB image. Similarly, you can create overlays of almost any color. For example, to create red coastlines, pick the coastlines image for all the components but disable the green and blue components by making their *Low* and *High* scrollbar levels equal to 255. Now you can overlay the coastlines with desired colors on the target RGB image with *Multi – Overlay Image*.

The *Transf-Convert to 24bpp (RGB)* function is also useful for combining typical WIM images with RGB images. For example, you may want to combine a chlorophyll image for the ocean with the true-color image for land. You can do that by creating a mask for land (and for ocean), creating separate RGB images for land and ocean, masking respectively the land or the ocean part with *Multi-Mask w. Image* and then the combining (with *Multi-Overlay Image*) the masked RGB images.

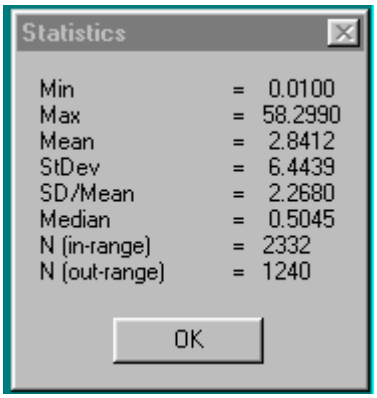
Spectral Plot

Interactively plots pixel values from multiple images versus image number. If the images represent multiple (co-located) spectral bands, then the plot is a true "spectral plot". You should right-click on the plot area and drag the mouse pointer over the image. For each pixel the plot shows the values of the corresponding pixel as a function of the image number of each of the loaded images. The pixel location (x; y and Longitude; Latitude) is also shown. The images can represent anything and do not have to be spectral bands. For example, if you load images of Chl-a and SST then the "spectral plot" would show the corresponding values of Chl-a and SST for a selected pixel. When six bands of normalized water-leaving radiances (nLw_412, nLw_443, nLw_488, nLw_531, nLw_551, nLw_667) are read from a MODIS Level-2 file you can have something like the plot below. Note that the spectral shape shows a typical spectrum of water-leaving radiance. The minima in both the short and long wavelength parts are due to strong absorption by CDOM, phytoplankton and water.



Statistics

Calculates a number of statistical variables for the currently selected rectangle, transect or a selected area of the current image. Statistics selected Transects or Rectangles read from a Track file (.trk) can be calculated with *Geo-Get Vector Objects*.



When calculating statistics, it usually makes sense to exclude the pixel values represent no data or bad data. To do that you can exclude the lower and upper range of pixel values from the calculations as values such as 0 or 255 are often used for indicating out-of-range values or for graphic overlays. If the current image has its projection set, you can specify the rectangle used in the statistical calculations either in pixel or geographical coordinates by selecting between *X range/Lon range* and *Y range/Lat range*. E.g., if *X range* has been selected, the longitude range will be calculated from the *X*-coordinates whereas if the *Lon range* has been selected, the *X*-coordinates will be calculated from the *Lon range*. Please observe that in the video coordinates the *Y*-axis starts from the top (e.g. the top left corner has coordinates 0; 0) whereas in the geographical

coordinates latitude increases from bottom to top. If an area of interest on the image has been deselected (by clicking on the image), the *X/Lon* and *Y/Lat* selection made on the *Start Statistics Calculation* dialog box will be preserved and can be used for another image. To reset the area selection, run *Statistics* without a selected range.
The described way of calculating statistics can be used for any rectangular area or transect. For irregularly shaped areas there is another variant of the *Statistics* function that uses a second image (a Mask) to specify the areas of interest. Masked areas can be of any shape. To use that check the "*Use Mask*" check box and specify the mask image number and the mask number. The mask image uses pixel values different from 0 to specify pixels for which statistics is to be calculated. You can create a mask image in many ways. The most common approach is to create a coastlines image with *Geo-Get Map Overlay* and then use *Edit-Draw* to create non-zero areas. Up to 255 different masks can be created on a single image using mask values from 1 to 255. The only restriction is that the different masks cannot overlap. Please note that if you use pixel value 255 to create the coastlines then you should not try to use mask number 255 unless you really want to calculate statistics for the coastlines.

Time series

Makes a time series plot by extracting the currently selected line or rectangular area from a set of images currently in memory. The set of images must be of the same size and form a sequence. The vertical plot range is stretched between the respective minimum and maximum pixel value. The respective means are connected by a line whereas individual pixel values are shown by x. Optionally allows to save the pixel values in an ASCII file where the number of lines equals to the number of pixel values in the selected area of interest and the number of columns equals to the number of images selected. You can then import the saved file into another program for further analysis or plotting.

X-Y Scatter

Makes an X-Y scatter plot from the currently selected line or rectangular area of the current image versus respective pixel values of another image. The pixel values of the current image are assumed to be on the X-axis. The image number of the pixel values put on the Y-axis has to be entered. The plot ranges are initially between the respective minimum and maximum pixel values on both axes. Subsequent refinement of the *Min* and *Max* values on both axes allows filtering out pixels affected by clouds, missing data or examine the relationships for different clusters of pixels. The selected clusters of pixels can be saved in an ASCII 2-column file. This file can then be imported into another program (e.g. MS *Excel* spreadsheet) for further analysis or plotting.

6.5 Geo - Geo Operations

Bathy Image

Reads a bathymetry database file with ocean depth values and creates a *Int16* image of bathymetric depths (in meters). The current (source) image must have a defined projection and the created bathymetry image will have the same size and projection as the source image. You can easily create a depth image for any area of the world by first creating a new blank image in *Linear* projection (*File* – *New*) and then creating a corresponding bathymetry image. WIM includes two world bathymetry databases: a lower resolution (5 minute latitude-longitude grid, *world_bathy_5min.dat*) and a higher-resolution (2 minute latitude-longitude grid, *world_bathy_2min.dat*), both in the *Maps* folder. The bathymetry image is created from the data in these files using bilinear interpolation. After creating the bathymetry image you can create depth contours using the *Examine* – *Contour Lines* function. For visualization (e.g. for overlays) it is easier to use Byte (1bpp) images instead of Int16 (2-bpp) images. Therefore you may want to convert the Int16 depth image to a Byte image using some sort of scaling. You can use *Transf – Convert* and convert to Byte with Log scaling with a slope of 0.025 and intercept of 0. After that you can use *View – Set Colors* and stretch the colors from 207 (Start) to 0 (End). Please note that Start is higher than end. With most palettes that will create more intuitive colors. You can then overlay the depth contours (isobaths) on top of the scaled depth image with *Multi - Overlay Image* (remember to select the image first and then select the overlay image number !).

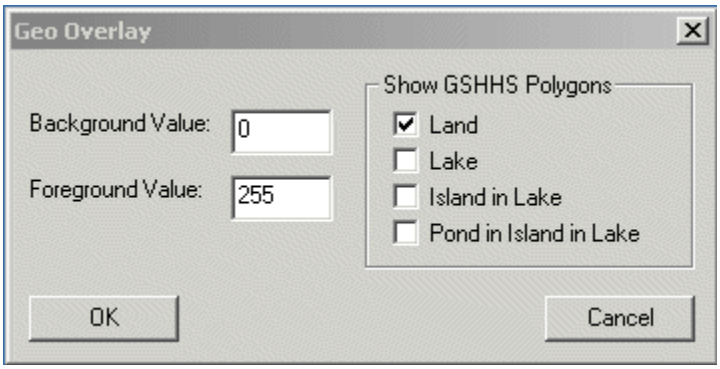
Get Bathy Overlay

Reads an ASCII text file database file with coordinates of a certain depth isobath and creates an overlay file with. FTP site. In order to create a bathymetry overlay the image has to have projection information. After creating the overlay you can merge it into the image by using *Multi* - Overlay Image (select the image first and then select the overlay image number!).
The bathymetry contours are read from a *.di6 file that has a sequence of Longitude and Latitude values of the contiguous line segments separated by an integer which states the number of records following that should be connected. An example of a *.di6 file:

```
        7
-70.241949 42.650208
-70.223761 42.663116
-70.214960 42.715920
-70.233148 42.735868
-70.260137 42.728827
-70.250749 42.656075
-70.241949 42.650208
        2
-70.233148 42.517025
-70.268937 42.524066
```

Get Map Overlay

Reads a database file with coordinates and creates an overlay file with the selected features. WIM includes "low-resolution" global databases for creating overlays of coastlines (*gshhs_crude.b*), country boundaries (*boundlow.dat*) and rivers (*riverlow.dat*). Corresponding high-resolution (1 km) files are provided in the *Maps* folder of the WIM CD. WIM includes functions for reading GSHHS (Global Self-consistent Hierarchical High-resolution Shorelines) shorelines format (Wessel and Smith, 1996). GSHHS files have extension **.b* and include datasets in 5 different resolutions (in order of increasing accuracy): *crude , low, inter, high, full*. The resolution of the *full* GSHHS dataset is better than 100 m. The sizes of GSHHS files change from 167 KB for the *crude* to over 87 MB for the *full* dataset. The installation of the GSHHS full dataset can be skipped in the Custom installation option. When creating the overlay the user can choose the level of shorelines to include, e.g. whether to include only the land-sea shorelines, and/or lakes (inland water-bodies) or higher-level shorelines like islands in lakes, etc. The time for scanning these GSHHS datasets for coastlines increases proportionally with the size of the dataset. The additional fine-scale details are not necessary and sometimes distracting when dealing with global or large-scale images. Therefore, the most appropriate map file depends on the resolution and scale of your image. Only the crude-resolution file (*coast_crude.b*) is included with the downloadable WimLE package.



In order to create a Map Overlay the image has to have projection information. After creating the map overlay you can merge it into the image using *Multi* – Overlay Image (select the image first and then click on the Overlay icon on toolbar). Both the background and foreground pixel values of the overlay images can be selected.
If the selected *Foreground Value* is <= 255 then the resulting image is a Byte image whereas if the *Foreground Value* is > 255 then the resulting image is a Int16 image. Please note that the maximum value of Int16 is 32767 and you probably don't want to select a bigger value for the Foreground as the value will be wrapped into a smaller value. **The background value should always be 0** if you want to overlay this image on top of another image. A longer way of making the coastlines to have pixel values higher than 255 is first to create a regular Byte image with the coastlines of pixel value 255, then use *Transf – Convert* to make it into an Int16 or a Float image and then *Transf – Linear Trans* to multiply the pixel values by, e.g. 255.

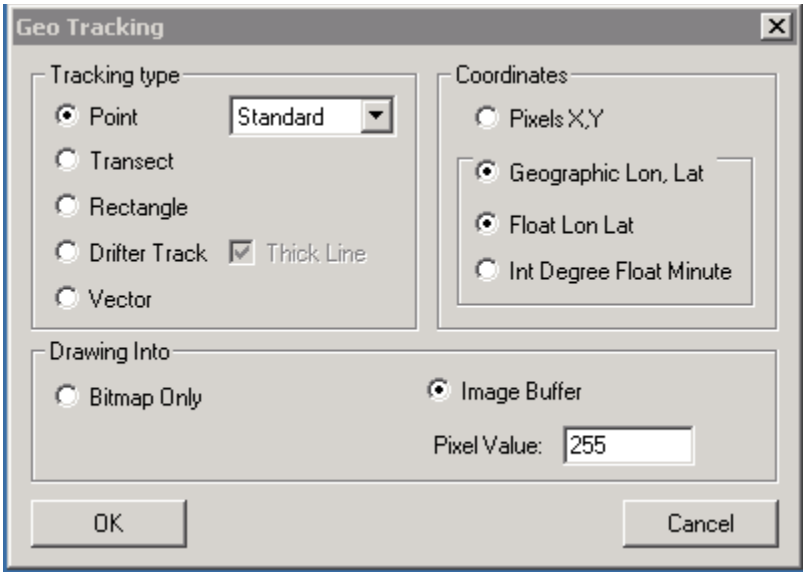
Get Vector Ojects

This function loads vector objects such as points, transects, rectangles, drifter tracks or vectors from a text file and displays them on top of an image. allows to pick values from an image or calculate statistics of a predefined list of points, transects, rectangular areas or drifter tracks.

Note: starting from version 6.46 WIM can read the ESRI shapefiles (http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf) as other vector objects. Currently the shape objects are written into the image or into the bitmap but they don't get into the Vector Objects table. A sample shapefile (*USstates.shp*) with US state boundaries is installed in the *Maps* folder.

For example, you can have the coordinates (longitudes and latitudes) of a series of points specified in a text file and after loading the points as vector objects you will see the points plotted on the image. After selecting a particular vector object, it starts blinking and you can get statistics of the image corresponding to the location of a vector object, e.g. 3 x 3 pixel window corresponding to a point object. A related function *Geo - Read Vector Objects from HDF* reads similar vector objects from a (binary) *HDF* file.

Each image has its set of *Vector Objects* that you can see in the *Vector Objects table* (select *View-Vector Objects* or click on the  icon on the toolbar). After completion the imported vector objects are shown with *View-Vector Objects*. By clicking on a vector object in the *Vector Objects* table the selected vector objects starts flickering by changing its color.



Drawing Into. The selected vector objects can be marked in the screen bitmap or actually recorded with a selected pixel value into the image buffer.

When marked only in the Bitmap they will disappear the next time the screen bitmap is rebuilt from the image buffer. When recorded in the image buffer the actual pixel values in the image buffer are changed to the selected pixel value. The changed image can then be saved. Note that when you change pixel values in the image and look up the pixel values, the pixel values that you select will not be the original ones but the values that you assigned yourself. Therefore, it does not make sense to do statistics over the pixels that you have assigned yourself.

When reading text files with vector objects, the first line in the list is usually regarded as a comment line and is skipped. The coordinates can be specified either in the video format (x = pixel column number, y = line number, counting from the upper left corner) or in the geo-referenced format (longitudes and latitudes). The geo-referenced coordinates can be in the *short* format (decimal *Longitude* and decimal *Latitude*) or in the *long* format (int *Lon-degree* float *Lon-minute* int *Lat-degree* float *Lat-minute*). Please note that the *Point* (*.csv* or *.pnt*) file format can be either *Lon, Lat, Value* or *Lat, Lon, Value*, i.e. starting with either *Latitude* or *Longitude* and is specified in the *Settings - Misc - Lat, Lon, Value format* options. Another special format for point data is the NOAA drifter file format (*.DAT, see *Drifter track* below) which is always in the "*Lat firs*t" format.

Point

You can select the symbol used to mark each point in the image. The default symbol ("Standard") is a filled circle. An extensible list of available symbols is stored in the file *PointTypes.xml* and new symbols can easily be created by editing that file. For example, a simple rectangle pointing up is defined by the following series of zeros and ones:

```
<name>Triangle</name>
<data>
  <row>0001000</row>
  <row>0010100</row>
  <row>0100010</row>
  <row>1111111</row>
</data>
```

The file *PointTypes.xml* must be in the same directory with the WIM executable (typically in *C:\Program Files\WimSoft*). Selecting different symbols for different data sets the user can distinguish

There are several file formats for Point objects:

```
Header
x1 y1
x2 y2
x3 y3
.....
```

... or

```
Header
lon1 lat1
lon2 lat2
....
lonn latn
```

Each point should be on a separate line (ending with a CR/LF sequence). The longitudes and latitudes can be specified either as a single floating-point value (in degrees) or as a degree value and a floating point minute value. Please keep in mind to use negative numbers for both western longitudes and southern latitudes. Please note that WIM uses Longitude before Latitude (to be compatible with the sequence of x, y) and not vice versa. Both

```
-118   30.0   30   0.0
```
and
```
-118.5    30.0
```
are valid lines specifying the same point of 30° 0.0' N; 118° 0.0' W.
It is very convenient to have other information following the coordinates on the same line – this information will be included in the output file. For example, you can use in situ values and additional information following the Lon-Lat information.

The sequence of picked pixel values is shown as a profile. The locations of the picked pixels are shown on the image bitmap as black or white rhomboids (centered at the pixel) or as connected lines (if *Line* has been selected as *Area of Interest* in *View - Settings - General*). You can store the picked points (*x*, *y*, pixel value, etc.) in a file. In case of the *Geographic Lon Lat* option the *Lon* and *Lat* will be stored as floating-point geographical coordinates. The pixel values are saved according to the *Value Scaling* option in *View - Settings - General.* E.g., if *Log-Chl* or any SST option is selected as *Value Scaling*, the pixel values are converted to mg/m^3 or °C, respectively. Each point is regarded as the center of a small 3 x 3 pixel area and the following information is appended to each line:

```
Value  N_Valid  N_Invalid   Min   Max   Mean   Median
```

Here *Value* is the pixel value at the specified point, *N_Valid* is the number of valid pixels in the 3 x 3 area, *N_Invalid* is the number of pixels outside the valid range, *Min*, *Max*, *Mean* and *Median* are statistics for the 3 x 3 area.

There is a sample point file *calsta81* in the *long* geo-referenced format corresponding to a US West coast CZCS image *calchl81.img*. The extensions for point files can be either *.csv* or *.pnt*. If a listed point happens to be outside of the image, the value and statistics are recorded as –99.999.

You can easily create a "point" file using any text editor or by *Examine - Point Save*. You can either pick individual isolated pixels or use free-hand drawing to pick pixels along a curve. The file saved with *Point Save* can be

read with *Geo–Get Vector Objects* - *Point,* depending on the selected geo-coefficient options.

Transect

is the option where the endpoints of straight lines are specified, respectively, in either video or geographic coordinates. (In the video coordinates 0; 0 is in the top left corner). The format is the following:

```
Header
x11 y11 x12 y12
x21 y21 x22 y22
....
xn1 yn1 xn2 yn2
```

Here x_{i1} and y_{i1} (*i = 1, 2*) are the start coordinates, and x_{i1} and y_{i2} are the end point coordinates of track. The transect (track) files are assumed to have extension **.trk.*

In case of the geo-referenced option, the longitudes and latitudes can either be specified in the *long* (degrees and minutes) or *short* (floating point degrees) format, e.g.,

```
Track-geo: CalCOFI lines 87 and 93
-123 35.6 29 50.9 -117 18.8 32 57.5  line 93
-123 45.3 31 19.8 -118 30.2 33 53.2  line 87
```

or

```
Track-geo, short form: CalCOFI lines 87 and 93
-123.593 29.848 -117.313 32.958
-123.755 31.330 -118.503 33.887
```

The examples are from sample files *calgeo.trk* and *calgeosh.trk,* respectively), corresponding to a sample image *calchl81.img*.

The same tracks in video format (from a sample file *calvideo.trk*) look like this:

```
Track-video: CalCOFI lines 87 and 93
234 359 324 315
232 338 307 302
```

If an image is loaded, the pixels are picked along the track, displayed on the screen as a profile, and optionally stored in an ASCII text file (with a format: *X*-coordinate, *Y*-coordinate, Pixel value). The *Value Scaling* settings in *View - Settings – General* are used to convert the pixel values to geophysical values.

If no image is in memory, a clean image buffer with the current size is created and the track-points are assigned the value of 255. You can then

store the track as an image *(File - Save As - Image...*) or overlay file (*File - Save As - Overlay*..), or use it as an overlay for another image. In order to overlay the track on another image, first switch to the image and then use *Multi* – *Overlay Image*.

Note: Keep in mind that WIM usually shows longitude (corresponding to X-coordinate) before latitude (corresponding to Y-coordinate) and not *vice versa*. Also be careful to use negative numbers for both western longitudes (e.g., `-118 30.0` or `-118.5)` and southern latitudes. The total number of tracks in one file is limited to 100. If any of the track-points appears to be outside of the image, the respective *X* or *Y* value is assumed to be either 0 (if negative) or the maximum *X* or *Y* value.

Note: The first line is always regarded as a header (comment) and skipped.

Please note that pixels are picked along the track from the beginning to end and then plotted from left to right. If your track goes from east to west, you will see a reversed graph on the screen.

Rectangle

In the Rectangle option rectangular areas are specified by their upper-left and lower-right corners in either video or geo-referenced coordinates. The file formats are exactly the same as in the Transect (= track, .trk) option. For each specified rectangle the *Examine - Statistics* function is performed.

If no image is in memory, a clean (zero values) image buffer with the current size is created and the points along the perimeters of the rectangles are assigned the value of 255. You can then store it in various image formats (e.g. *File - Save As - Image...* or overlay file with *File - Save As - Overlay*..), or use it as an overlay for another image. In order to overlay the rectangles on another image, first switch to the image and then use *Multi* – *Overlay Image*.

Drifter track

This is a special version of the *Point* object and is like a set of points treated together. An arrow showing the direction of the *drifter track* is shown. The file format is the same as that of the *Point* vector objects. Additionally, the NOAA drifter data format (*.DAT) can be read directly. These files are assumed to have either *.DAT or *.txt extension and always have the latitude value preceding the longitude value regardless of the setting of the *View* – *Settings - Misc - Lat first* checkbox. A section of the NOAA drifter track format is given below:

```
06325 39096   3  4 L
06325 39096   3  4 K 1 2003-07-26 00:25:15  31.000  240.000  0.000 401647450
06325 39096   3  4 2
06325 39096   2  4 L 2 2003-08-04 05:28:41  26.000  239.000  0.000 401647448
06325 39096   2  4 K
06325 39096   2  4 2
06325 39096   3  4 K
```

This short section has only two points (31 N; 240E) and (26N; 239E). The rest of the lines have no data. Here 31 and 26 are the latitudes and 240 and 239 are, respectively, the longitudes. The longitudes here are converted by WIM from the [0, 360] degree range to the usual [-180, 180] range. WIM reads only the latitude and longitude values and ignores the rest.

Vector

Vectors can be read from two kinds of files. The type was usually set in *View - Settings* – *Misc* but currently the reading routine is expected to automatically recognize the vector type. The default vector format is: *x1 y1 x2 y2* that specifies the start and end coordinates of the vectors either in image or geographical coordinates. The image coordinates are in pixels (*Y* increases from top to down, *X* increases from left to right with pixel 0,0 in the top-left corner) and are assumed if the projection of the image is *Unmapped*. In the projection is specified then the geographic coordinates are assumed (x1 is the start longitude, y1 is the start latitude, x2 is the end longitude and y1 is the end latitude). You can easily create a vector file even with an editor. A sample vector file (*est256.vec*) in image coordinates for a sample image is provided. The file starts with a comment line followed by vector data each on a separate line, e.g.:

```
Comment
 180  100  170  111
 180  110  168   98
```

Another format is a specialized one that is used to plot the velocity vectors derived from the motion analysis of a sequence of 2 images (*Multi - Motion Detect*). Here the format specifies the start coordinates *x, y* (in video coordinates), the velocity components *u* and *v* in the *x* and *y* directions, respectively, and the cross-correlation coefficient of the calculation. The value of the cross-correlation coefficient gives an estimate of significance of the derived velocity vector. If the value is high, the vector is estimated with a high degree of confidence. The format is the following:

```
x  y  u  v  xcorr
```

The first 3 lines are not used and contain (1) a comment, (2) the names of the parameter file and the two image files, (3) the numbers of blocks where the maximum correlations were found, image sizes, and coefficients that can be used to convert the *u* and *v* speed values back to pixel separation in the *x* and *y* directions.

Here is an example:

```
Comment
param.par image1.pic   image2.pic
 20   20  256   256   0.7845 0.7845
180  100  12.5  -12.5  .89
190  110  11.0  -10.0  .90
```

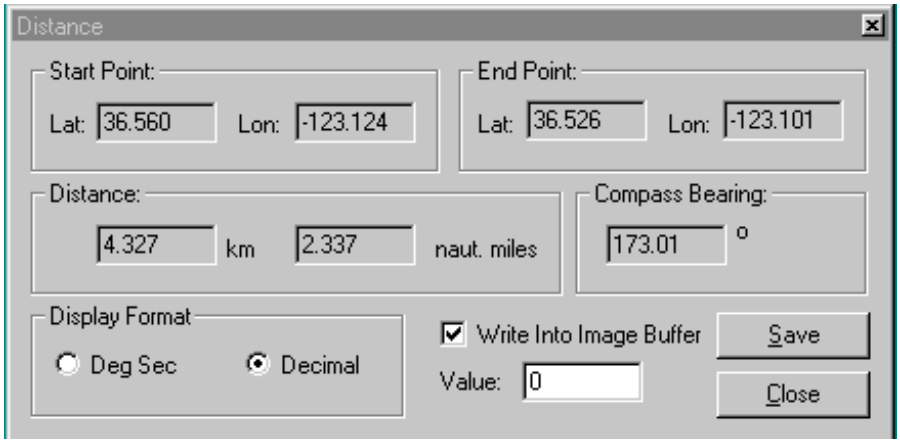
Note that the velocity values are given in relation to the mathematical coordinate system where positive *u* means direction to the right and positive *v* denotes upward direction on the image. Note that the *x* and *y* coordinates are integers or may be floating-point numbers meaning the longitude and latitude. The velocity and cross-correlation are float numbers. Before the vectors are plotted you are given an option to change their length by specifying a scale factor to which the vectors are multiplied, and to select a threshold value for the cross-correlation coefficient (float number) below which the corresponding vectors are ignored (not plotted) as they might be unreliable.

Grid

Creates a copy of the current image file with the latitude and longitude grid lines in it. The image must have projection information in order to create a latitude/longitude grid. The numerical values of the grid lines are optionally written into the image. A negative step for either the latitude or longitude grid disables the respective grid lines. For example, by using 10 as the latitude step and -1 as the longitude step produces only a latitude grid lines with 10-degree intervals.

Distance

Calculates distance in kilometers and nautical miles along a selected line between the selected start and end points on the surface of the Earth. The image must have a valid projection for distance calculation. The points, the distances and angles (compass bearing between points) can be saved in an ASCII text file that is similar to the .trk (track) file used in *Geo – Transect*. The format of the file is always decimal degrees of Longitude and Latitude followed by distance in km and angles in degrees whereas on screen the user can select between 2 different formats of Latitude and Longitude. The following example shows the screen layout.



The tracks for which the distances are calculated can be recorded in the screen bitmap or they may be recorded with a selected pixel value in the image buffer. When only the screen bitmap is changed the tracks disappear

when the bitmap is rebuilt. When values in the image buffer are changed the image with the changed pixel values can be save as a new image with the assigned pixel values along the tracks.

Read Vector Objects from HDF

Reads a *HDF* file with WIM vector objects such as points, transects, rectangles, or vectors (see Vector Objects). Vector objects can be read from ASCII text files with *Geo–Get Vector Objects*. The vector objects of an image are show with *View-Vector Objects*.

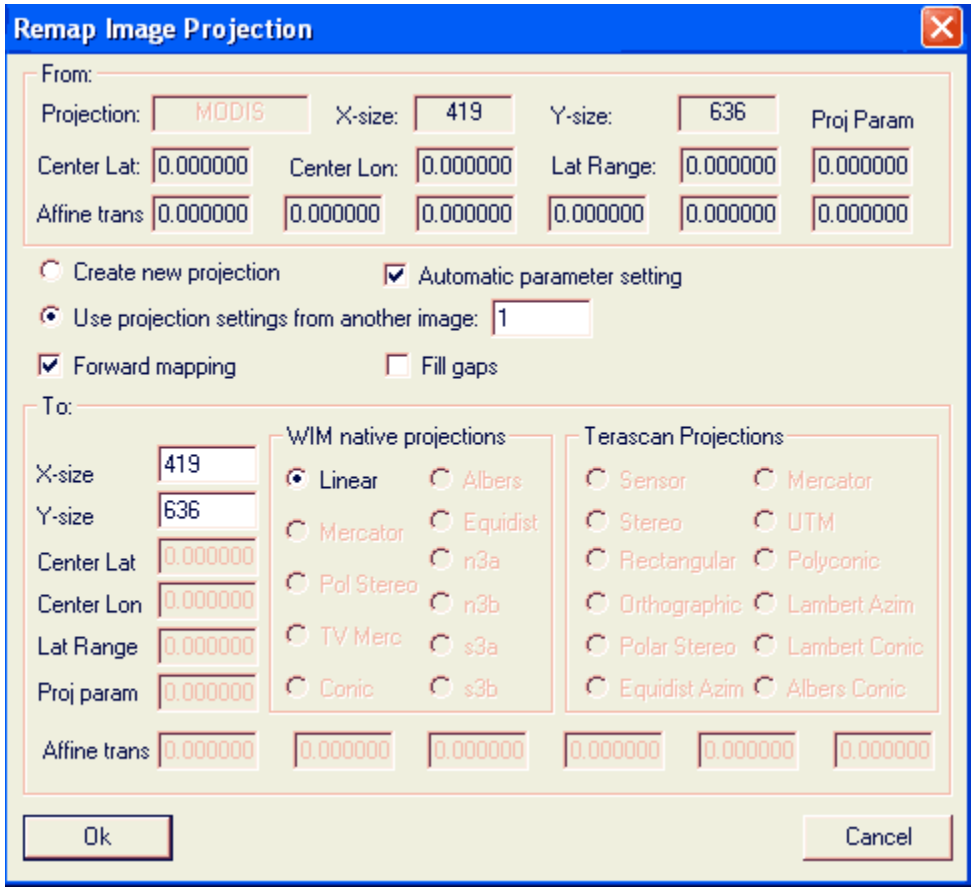
Remap Projection

Remaps the current image from the current geographical projection to another projection and/or to another size. It may be difficult to generate a new projection from scratch; it is much easier to pick a another projection from another image and remap the current image to that projection. Projections other than *Linear*, make use of additional reference variables (see *View - Settings* - Projection) that are read as HDF or CoastWatch attributes, CCAR image header or from the separate info-file. The meaning of the reference variables depends upon the projection chosen. In most cases, ref1 and ref2 define the latitude/longitude of a plane tangent to the sphere. The *Conic* and *Albers* projections may use all 4 reference variables. In a conic projection the globe is projected onto a cone. If the cone is tangent to the globe at one latitude, only *ref1* (reference latitude) and *ref2* (reference longitude) are used. The other variables (*ref3* and *ref4*) are needed only when the second reference plane is used.

Remapping is compute-intensive and can be slow for large images. On multi-core and multi-CPU systems the function is multi-threaded and uses multiple cores/CPUs concurrently. Unless the target image is much smaller than the source image the fastest method to use is ***Forward*** or direct mapping that takes every source pixel and puts it into the target image in the right position. The problem is that when the target image is larger than the source image then the result will be “gappy”, i.e. there will be pixels which did not get a value from a pixel in the source image. Gaps will not occur in case of *inverse* mapping as for each pixel in the new image the best matching pixel is located in the source image. Inverse mapping usually generates better looking images. However, for large target files inverse mapping is very slow. A compromise method is to first use *Forward mapping* and then inverse mapping (“*Fill gaps*”) only for those pixels that did not get a value with *Forward mapping*. We advise to experiment: start with *Forward* mapping and no *Fill gaps*. If gaps are a problem then use *Fill gaps* with *Forward mapping*. Anther way of eliminating gaps is to use the *Max* or *Mean-Pos* filters in *Transf - Filter*.

It is also possible to automatically remap the image to Linear projection. This operation finds the full latitude-longitude range of the image, creates a respective Linear projection and remaps the image to that projection. This is useful for images in LLA (Latitude-Longitude Arrays) projections such as MODIS and GLI Level-2 images. LLA projection is versatile but very slow. It is therefore often useful to remap a LLA image to Linear projection using *Automatic parameter setting* and deselecting *Use projection settings from*

another image. You can experiment with the *Forward* and *Fill gaps* settings. Be careful when using *Inverse* mapping as it may be very slow.



Unify Geo-coeff

This function copies the projection parameters (geo-conversion coefficients) (see *View - Settings - Projection*) of the selected image to the rest of the images in memory. This function is useful if you have a number of images in the same projection and you want to transfer the projection parameters to other WIM images. You can save the geo-coefficients (and other parameters) of to as text in a corresponding info-file with *File - Save Info*.

Unify Vector Objects

Is used to copy the *vector objects* of a selected image to all the other images in memory. This function is useful if you have a set of vector objects that you want to apply to many images. By using this function you can read the vector objects once and then apply them to all the images in memory.

View in Google Maps

This function opens a new window and displays the image boundaries in Google Maps (see http://www.google.com/maps) with a wealth of

geographical information. You must have active internet connection for this function to work.

View in Google Earth

This function opens *Google Earth* (must be installed, see http://www.earth.google.com) and displays the image. You must have active internet connection for this function to work. This function creates a temporary KMZ file in your %temp% folder and runs loads it with *Google Earth*. Note that you can also save your image in KML and KMZ files for later viewing in Google Earth using File – Save as – Google Earth KML or KMZ.
If you view many images then the temporary files accumulate in your temporary folder and you may want to delete from time to time the *wim_temp*.kmz* files from there. You can do that with Windows Explorer or run the following command:

```
del %temp%\wim_temp*.kmz
```

More information on using Google Earth, e,g, for making black areas transparent can be found in the exercises for ASTER images (http://www.wimsoft.com/Exercises_ASTER.pdf).

6.6 Transf - Image Transformations

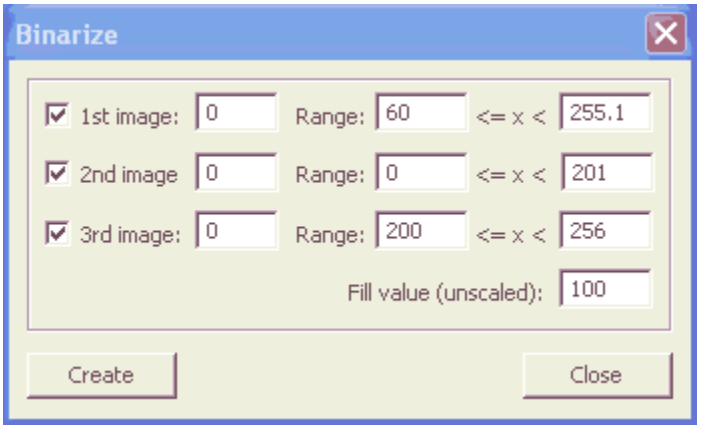
Transf consists of operations that are used to transform the current image to a new image. Transformations that use more than one image as input are included in the *Multi* menu section (multiple image operations). While most of the operations record their result in a newly allocated image buffer, some transform the current image buffer.

2 Byte To 1

Transforms a 2-byte-per-pixel image buffer to a 1-byte-per-pixel buffer. The options available are similar to those in *File - Open... Int -> Byte*. This flow-level function is **not recommended** for casual usage. Please use the related function *Transf – Convert* that adds much more functionality and is also much easier to use.

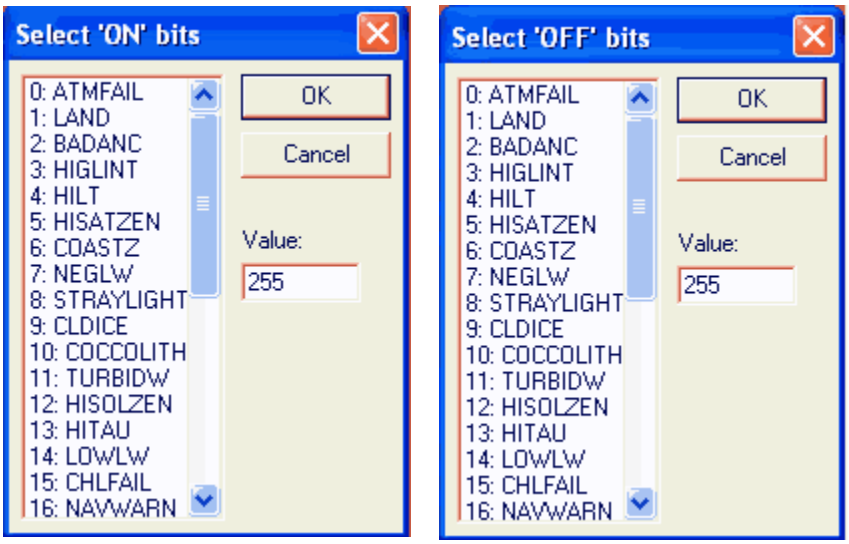
Binarize

The idea of the binarize function is to select pixels that satisfy certain criteria, for example larger or smaller than a threshold. It opens a new unscaled Byte image buffer and sets the "yes" pixels to a selected pixel value and the "no" pixels to zero. Regardless of the source image (1, 2 or 4 bytes-per-pixel) this operation returns a 1 byte-per-pixel unscaled image. This operation is useful for creating masks to be used in operations like *Multi – Mask w. Image*. Starting from WIM version 6.11 the binarize function is generalized to include criteria involving more than one image. Now you can set criteria on up to three images and include both lower and upper limits. For example, you may want to select all pixels that are lower than the threshold in image #1, above a threshold in image #2 and inside a range of values in image #3. The example below finds pixels where image #1 has values above a 60 and less than 255.1, image #2 has values below 201 and image #3 has values between 200 and 256. The new image will have unscaled pixel values 100 for pixels that satisfy the above criterion and zero for the other pixels. The dialog will remain open and the user can interactively modify the criteria, and get a new visual of the created image by clicking *Create* again. Clicking *Close* closes the dialog.



Bitmask

Opens a new (1-byte per pixel) image buffer and paints areas where the selected bits are ON and other selected bits are OFF with the selected pixel value. All others pixels are set to 0. The selected pixel Value should be from 1 to 255. The created mask images can then be used in operations like *Multi – Mask w. Image*. Bits can be selected with the mouse; multiple bits are selected with the mouse and the Control or Shift key pressed. This operation is useful for creating masks corresponding to the values of certain flags, e.g. for *SeaWiFS* and MODIS Level-2 images. as shown in the following example. Values of the bitmask flags can also be viewed directly on the source image by right-clicking on the image: the flags (bits) that are ON are shown in the window header.



First the user has to select the ON bits, then the OFF bits. For example, by selecting OCEAN as the ON bit and TURBIDW as the OFF bit the user will create a mask image with non-turbid ocean pixels having values 255 and all others with pixel values zero. MODIS Level-2 products have all quality flags that are not individually marked but are in groups of two. For example, Bits 1-2: Quality for all of Gordon's nLw bands, Bits 3-4: Quality for Carder's clear water epsilon band, etc. The user has to check the image attributes to see the corresponding flag groups. Two bit can code 4 numbers from 0 to 3. These numbers have the following meaning for MODIS quality products: 0 = good, 1 = questionable, 2 = cloud or sun glint contaminated, 3 = bad. The meaning of the different flags are usually given in the image attributes.

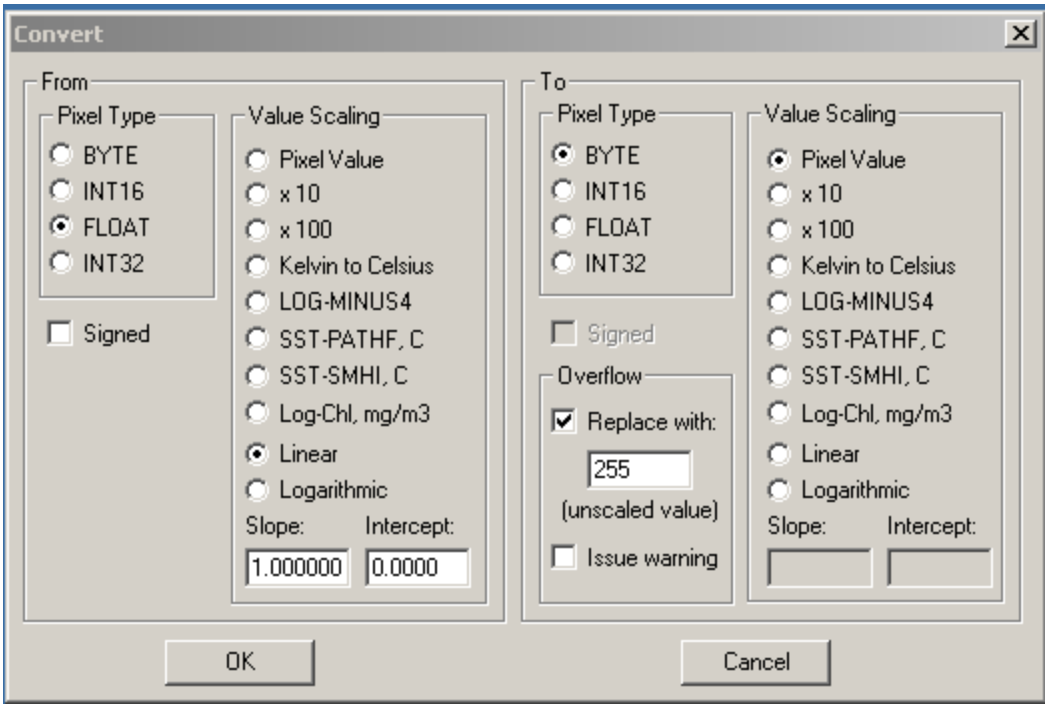
Convert

Makes transformations between 1, 2 and 4 byte-per-pixel image buffers with various *Value Scaling*.

For example, phytoplankton pigment (chlorophyll) concentrations in the ocean vary over several orders of magnitude. The distribution is usually log-normal with the accuracy of measurement approximately linearly related to the concentration. In order to compress a wider range of concentration

values into one byte, different logarithmic scalings can be used (see *Settings - Value_Scaling*.

When using the *Convert* on an image and a transverse conversion on the result, the original values should be restored. However, some discrepancies are inevitable due to the range limitations and the numeric rounding errors.
A separate option is available for overflow control. For example, when converting floating point numbers larger than about 66 to Byte values with Chl scaling, the pixel values get bigger than 255 that is the highest value for a Byte image. Overflow replacement can ensure that these values will be replaced with the maximum value of 255 and not truncated to a small pixel value that will probably not be desirable. A useful conversion is for SST from 2-byte scaled Kelvin images to scaled byte in Celsius using the Kelvin to Celsius option.



Convert to 24bpp (RGB)

Converts the current image to a 24-bit per pixel RGB image. Typical WIM images use a color palette to visualize pixel values as colors, but a RGB image has the red, green and blue components specified with a 8-bit unsigned integer. This function is useful to apply to a WIM image (other than RGB) to preserve the colors. After converting to RGB you can overlay it on a RGB image and the colors will stay the same (see *Examine* - RGB Image).

Decimal Exp (10^x)

Decimal Log (Log10(x)

Perform, respectively, pixel-wise exponential or logarithm operation on the image buffer and create a new FLOAT image.

Filaments

Finds filaments (linear structures) within the current image by using the Kasvand filter.

This operation is used e.g,. for finding ice ridges on ice images (Vesecky et al., 1989). If ridges are defined as bright linear structures, then we can find ridges with the following sequence of operations. (1) Find the filaments, (2) switch back to the original image, (3) find the histogram, (4) find the top 20% level (Histogram - Thresh), (5) binarize the original image at the top 20% value (only pixels at that value and above are marked as different from zero), (6) switch to the filament image, (7) mask (logically AND) with the binarized image (*Multi* - Mask w. Image). The final composite image is the image of bright AND linear structures.

Filter

Performs filtering of the current image. The selection of different filters is: *Median, Sigma, Mean, Max, Mean-Pos*. The image is scanned with a small window and the center point of the window is replaced by the result of the respective operation, e.g., mean, sigma or median. The borders of the image are directly copied from the source to the filtered image. The sigma filter (Lee, 1986) is often used to suppress speckle noise in synthetic aperture radar images. You must choose the standard deviation of the speckle noise (e.g., for a 3-look amplitude-averaged image, *sigma = 0.3017*). *Median* filter replaces each small window with its median, *Mean* filter with its mean, *Max* filter with its maximum, *Mean-Pos* with the mean of the non-negative pixel values. The *Max* and *Mean-Pos* filters can be used, for example, to smooth and fill gaps in sparse images generated from the *Lat, Lon, Value ASCII* input files.

Filter to Disk

Performs filtering operations similar to the **Filter** option above but instead of working with image buffers in RAM, transforms a disk file to another disk file. This operation is needed when the image is very large and impossible to load into RAM. For example, a “full scene” image of the Landsat Thematic Mapper™ single channel has 7942 x 6800 pixels. With each full image occupying about 50 MB it may be difficult to load the whole scene into RAM on a low-end PC. By first filtering the image file to another disk file and then sub-sampling it at lower frequency (*File - Open - Subset*) it is possible to get a better representation of the large image.

Gradients

Calculates a gradient image of the current image using either a 3-by-3 Sobel filter (summing absolute values of the vertical and horizontal convolutions) or by picking maximum centered differences in all possible 4 directions.

Linear Trans

Uses 2 float coefficients (*A, B*) to calculate a new image as a linear transformation of the current image. *New pixel value = A + B * old pixel value*. If the current image is a 1-byte-per-pixel image, the resulting pixel values are capped between 0 and 255. If you want to create an image with values over 255 then you need to first use *Transf – Convert* to create either a Int16 (2 bytes per pixel) or Float (4 bytes per pixel) image and then apply the linear transformation to that image. For example, when creating map overlays (*Geo – Get Map Overlay*) the result is a Byte image with values up to 255. If you want the coastlines to have pixel values higher than 255 then you can use *Transf – Convert* and *Transf – Linear Trans* and multiply by, e.g. 255.

Mirror

Makes an in-place mirror transform of the current image over the horizontal, vertical or both horizontal and vertical axes. Repeating the same operation again restores the original image. The image name is modified when the operation is first applied. Note that to get the image what you see when you turn your head upside down, you have to mirror it both over the horizontal and vertical axes.

Reduce Image

Makes a reduced image from the current image. Pixel values of the new, reduced image are assigned by picking the nearest neighbor, or by using the sum, maximum, minimum, or average over the source window. E.g., if the reduction is by a factor of 2, the operations will be done over non-overlapping windows of 2 x 2 pixels. When the sum over the window gets larger than 255, the new pixel value will be 255. Due to a requirement of the Windows memory management, the width (*X*) of the new, reduced image has to be divisible by 4. If *X* is not divisible by 4, *X* will be increased as necessary.

Replace Values

Opens a new image buffer of the same size and type (bytes per pixel, value scaling) and replaces a range of pixel values with a new value. If *Value Scaling* is set to *Pixel Value*, then a range of pixel values is replaced by a new pixel value. However, if a different *Value Scaling* option is used (see *View - Settings – General –* Value Scaling), WIM actually replaces values based on the current geophysical value and not the pixel value. The replacement value that you enter is also treated as a coded value. Only integer values can be entered. If you want to replace values based on their pixel values, set *Value Scaling* to *Pixel Value*.

Solar Correction

Does a simple correction for visible band images based on the calculated solar zenith angle and creates a new image buffer with the corrected values. Using the geo-conversion coefficients (*View - Settings - Projection*) or, equivalently, the longitude and latitude range to get the pixel geographical coordinates, and the time of the satellite pass the solar zenith angle is computed for each pixel. The sun zenith angle is calculated corresponding to the time of the image: the Julian day, and the GMT hour and minute. If the Julian day is 0, it is calculated from the year (last 2 digits, e.g., 94 for 1994), month and day. For a CoastWatch image all the necessary auxiliary data are retrieved from the image header.

Atmospheric correction is usually performed to approximate the water column reflectance defined as

$$R_w = \pi L_w(\lambda) / E_d(\lambda)$$

where L_w is the water-leaving radiance, and E_d is the downwelling irradiance entering the water, λ is the spectral band (e.g. Stumpf, 1992). While the full atmospheric correction is beyond the scope of a general program like WIM, a simple approximation is performed here by assuming

$$E_d \sim (1/r^2)\cos\theta_o \, T_o$$

where E_o is the solar constant, r is the normalized Earth-Sun distance, θ_o is the solar zenith angle (for calculation see Stumpf, 1992), and T_o is the transmission through the atmosphere

$$T_o(\lambda) = \exp[-(\tau_r(\lambda)/2 + \tau_g(\lambda))/\cos\theta_o]$$

where τ_r is the Rayleigh optical depth, τ_g is the gaseous absorption optical depth (i.e. for ozone and water vapor). Then a simple correction is performed by dividing the pixel value by E_d. The optical depths are specific for each sensor and can be modified with the *View – Attributes* dialog-box. The default values for NOAA/AVHRR bands 1 and 2 are taken from the *DECCON* program by Stumpf and Townsley.

Both for a 1- or 2-byte/pixel source image buffer the output will be a 1-byte/pixel buffer. For 1-byte/pixel buffer the correction is not performed for values 0 and 255, i.e. for out-of-range values. When calculating the output, pixels are scaled according to the albedo minimum and maximum values (*View - Attributes*). If no albedo range has been specified, the range is assumed to be from 0.0 to 25.5 and the corrected albedo values are multiplied by 10.0.

Square

Opens a new float (4 bytes per pixel) image and calculates the squares of pixel values of the current image.

Square root

Opens a new float (4 bytes per pixel) image and fills it with the square roots of the pixel values of the current image.

Texture

Calculates a new image with a texture indicator of the current image. The standard deviation or variance in a window can be selected. The image is scanned with a small window and the center point of the window is replaced either by the standard deviation or variance in the window. The borders of the image will have zero values. You must select the window size (side length) in pixels. Only uneven numbers between 3 and 31 are allowed. In case of a small window the features have better resolution but higher variance. You must keep in mind that only integer values between 0 and 255 can be stored in WIM images (8-bits per pixel). As the interesting range of the standard deviation and variance can be too low for that, you can choose an integer constant with which the real value is multiplied before converting to a byte value. Different regions of the image may require a different multiplication constant. The new image will be named <*TexSD_WxC*> or <*TexVar_WxC*> where *W* is the window size and *C* is the multiplication coefficient.

Zoom

Opens a new image buffer and zooms using either the *Nearest Neighbor*, *Bilinear* or *Bicubic* method. With the *Nearest Neighbor* method the new image is double in *X* and *Y* size. The *Bilinear* and *Bicubic* methods produce smoother looking images and also allow to select a floating-point scaling factor. When the scaling factor is less than 1.0 the new image is actually smaller than the source image. Please note that a somewhat related function View - Zoom zooms the viewable bitmap and does not create a new image buffer.

6.7 Multi - Multiple Image Operations

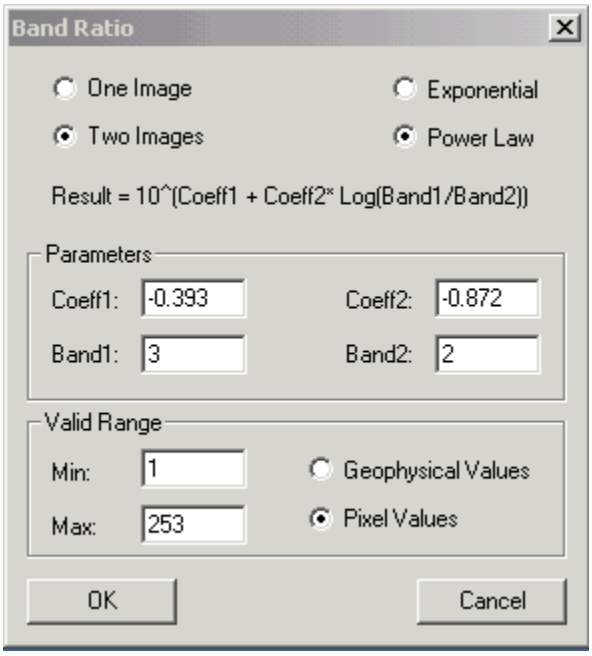
Multi consists of operations that are used to transform several images to a new image. It is usually assumed that the current image is the first operand, and you have to specify the second operand image by its index number. The resulting image is usually recorded in a newly allocated image buffer. However, with Mask w. Image and Overlay Image you can also record the result in the current image by specifying the second image index as a negative number. This option may be preferred if your images are very big and you want to save RAM.

Add 2 Images

Adds two images pixel-wise. If the *Value Scaling* of an image is set (see *View - Settings - General - Value Scaling*) the decoding operation is performed before the addition. In case when both images pixel size of 1 byte - if the sum is larger than 255, 255 is assumed as the new pixel value. In case of images with different pixel size, the resulting image has the maximum pixel size of the operand images. The images must be of the same size. The *Value Scaling* of the resulting image is set equal to the *Value Scaling* of the current image.

Band Ratio

Calculates various generalized band ratios for one or two images (bands) pixel-wise. Images are selected by their sequence number (see List of Images).



E.g. for the *Exponential* case and one image selected, the result is:
`Result = Coeff1 * Band1^Coeff2,`
for two images selected, the result is:
`Result= 10^(Coeff1 * (Band1/Band2)^Coeff2).`

A common format is designated *Power Law* and is often used with a 2-band ratio in bio-optical algorithms [see *O'Reilly et al*., 1998]:

```
Result= 10^(Coeff1 + Coeff2 * log(Band1/Band2)).
```

For example, the Morel-1 Chlorophyll algorithm [*O'Reilly et al*., 1999] is

```
Chla= 10^(0.2492 – 1.768 * log(Lwn443/Lwn555)),
```

where Band1=Lwn443 and Band2=Lwn555 (Lwn = normalized water-leaving radiance); the coefficients Coeff1=0.444, Coeff2=-2.431, Band1=Lwn490, Band2=Lwn555 give the CalCOFI two-band linear algorithm [*O'Reilly et al*., 1999].
The *Power Law* band ratio is a simple Linear regression in the log-log-transformed space. A simple Chl algorithm uses typically the blue and green bands of the are normalized water-leaving radiances ($L_{WN}(443)$ and $L_{WN}(555)$). A simple CDOM (colored dissolved organic matter) algorithm was proposed [*Kahru and Mitchell*, 2001] for SeaWiFS using $L_{WN}(443)$ and $L_{WN}(510)$:

```
CDOM = 10^(-0.393 – 0.872 * log(Lwn443/Lwn510)),
```

where CDOM is expressed as CDOM absorption at 300 nm (in m^{-1}). These coefficient values are the default values when the Band Ratio function is used for first time. When changed, the new coefficient values will retain their value in your computer's registry for easy access next time. For OCTS bands $L_{WN}(443)$ and $L_{WN}(520)$ the coefficients are Coeff1 = -0.411 and Coeff2 = -0.703.
The resulting image of the Band Ratio calculation is always a 4-byte-per-pixel FLOAT image. As it is hard to guess a convenient color range for the newly created FLOAT image, as a first step, you can do *View* - LUT Stretch to stretch the colors to the range between actual minimum and maximum pixel values. A FLOAT image can be transformed to the more convenient BYTE image with *Transf – Convert* – BYTE and choosing a convenient scaling. For Chl use the *Log-Chl* scaling. For CDOM a simple Linear scaling with Slope=0.01 and Intercept=0.0 is usually OK but limits the upper values to 2.55.

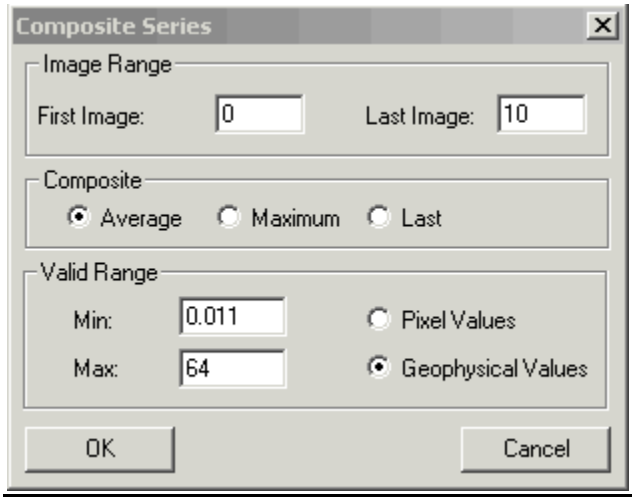
Another important aspect is filtering out pixels that are out of the *Valid Range*, i.e., usually clouds or land or otherwise failed. You can choose the *Valid Range* either for the *Geophysical values* (default) or *Pixel values*. When dealing with BYTE images (e.g., SeaWiFS Standard Mapped Images) it is easy to remember that values 0 and 255 are commonly used for land or clouds, i.e., the *Pixel Values* should be, e.g., from 1 to 254.

Compare

Compared two images pixel-wise considering the “Equality threshold”. Floating point images (e.g. Float32 or Float64) can be considered equal if their respective pixels differ by a very small to a value. The pixel-wise comparison stops if a pixel is different between the two images. A related operation Multi-Difference creates the difference image but you then have to examine that image if any pixel values are different from 0 or from another small threshold. Multi-Compare makes this process easier.

Composite

Calculates a generalized "average" (composite) of a series of images while trying to eliminate missing pixels e.g., due to clouds. The images must be in a contiguous series of image buffers and with the same image dimensions. The composite can be calculated in several ways: (i) taking a pixel-wise average of the series of images; (ii) picking the pixel-wise maximum over the series of images; (iii) picking the last valid value for each pixel. Invalid pixel values can be discarded from the calculation and the number of valid pixels is accumulated (only for methods i and ii) to create an image of valid pixel counts, i.e., the number of valid values used in the averaging process of the composite. When using the *Last* pixel method for compositing it is assumed that the images are sorted in increasing time. For example, if using images numbered 0 to 10 it is assumed that image 0 is the earliest and image 10 is the latest. The *Last* pixel method is useful when you are interested in the current situation but some of the areas on the latest images are covered by clouds or unusable for other reasons. The pixels missing on the last image are therefore retrieved from the nearest previous image.



You can specify the range of valid pixel values (Valid Range Min and Max) as often some pixel values (e.g., 0 or higher end values near 255) are used to indicate missing or invalid data and should be excluded from the calculation of averages. The valid pixel range can be specified either in pixel values or the real geophysical values (e.g., pigment concentration or SST). Images with different pixel size can be used. The type of the *Composite*

image is the same as the first image used in the series or float (4 bytes per pixel) for images with no Value Scaling. The *Count* image is always a 1-byte image as the number of valid pixels cannot exceed the number of images in the series (< 256). *Value Scaling* of the resulting *Average* image is set equal to the *Value Scaling* of the current image, *Value Scaling* of the *Count* image is always *Pixel Value*.

Difference

The operation is related to *Multi – Subtract Image*. It calculates an image of absolute differences between the two images. The images must be of the same size. *Value Scaling* (see *View - Settings - General*) is used for calculating the real value before the difference is taken. The resulting image is created as a 4-byte Float image that is usually not the best option but is the most versatile. In most cases is convenient to scale the resulting image into a Byte or Int16 image using the *Transf – Convert* function.

Divide w. Img

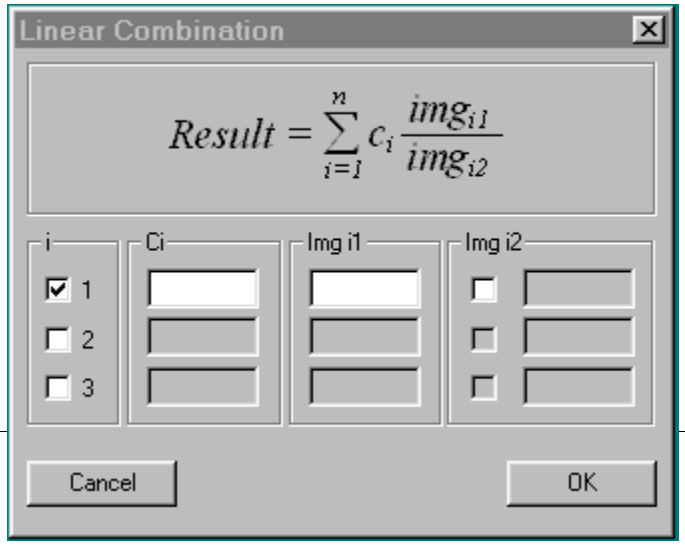
Divides the current image with another image pixel-wise. If the pixel value of the second image is 0, the result of the division is 255 (the maximum value for 1 byte-per-pixel image). *Value Scaling* is used to calculate the real values before the division. The images must be of the same size. The *Value Scaling* of the resulting image is set equal to the *Value Scaling* of the current image and the pixel size is always set to float (4 bytes per pixel).

Insert Image

Allows to insert one image (the source) to another image (the target) with the upper left corner of the inserted image starting at a specified position in the target image. If you specify a target image index that is not allocated in memory (any number larger than the last image buffer), a blank image buffer with the specified size is allocated and used for the insertion.

Linear Comb

Calculates a generalized linear combination of up to three images or image ratios pixel-wise.



By either selecting or not selecting the second image, each term is either a an image multiplied by a coefficient or a ratio of two images multiplied by a coefficient. Images are selected by their sequence number (see List of Images). The resulting image is always a 4-byte-per-pixel float image.

Mask w. Image

This function allows selection of the pixels depending on the pixel value of another image (i.e. a mask image). A new image is created with only the selected pixels retaining their value while the other pixels (unselected) will be converted to zero (pixel value zero). The pixels that are selected on the mask image retain their values whereas those pixels whose counterparts on the mask are not selected will become zero. It prompts for the mask image index, the mask pixel value and whether to select all pixels smaller and equal or larger and equal to the mask value. The default values (0 for mask value), and "Yes" for smaller and equal are suitable for picking the "good" pixels of MODIS images. MODIS data are delivered with a corresponding Quality flag image that has pixel values corresponding to the quality flag as follows:
0 – good
1 – questionable
2 – cloud
3 – bad
This meaning of the flags here is given as an example only and may depend on a particular convention and data type. The default selection would selecting pixels with Mask value =< 0, i.e. "good". When selecting pixels with mask value >= 3, for example, only the pixels of quality "bad" are returned.

Various masks can be easily generated with WIM. For example, sometimes it is necessary to mask all ocean pixels or all land pixels. A simple way to do that is thresholding an appropriate image with *Transf – Binarize*. In order to save disk space, masks are efficiently stored as compressed images (*Save As... Compressed*).

Motion Detect

Calculates the apparent motion between two consecutive images using the maximum cross correlation method (e.g. Emery et al., 1991). Several parameters that are needed to tune the subroutine have to be previously recorded in a parameter file. A sample parameter file (*64.par*) as well as a pair of test images (*64_1.img*, *64_2.img*) are provided. As this is a compute-intensive routine, the test images are only 64 x 64 pixels in size to reduce the calculation time. The structure of the parameter file is the following:

```
64    dx1
64    dy1
64    dx2
64    dy2
0     x1st
0     y1st
```

```
64    dx1c
64    dy1c
0     x2st
0     y2st
10    xwin
10    ywin
8     maxmot
10    rangebeg
250   rangeend
5     overlap
24.0  timediff
1.1   pixx
1.1   pixy
0.6   thresh
```

The numerical values must be first on the line, the following names are just comments following a "whitespace". Here *dx1, dy1* and *dx2, dy2*, represent, respectively, the dimensions of the first and second image, *x1st, y1st* and *dx1c, dy1c* - respectively, the start and dimensions of the area where correlation is to be done, *xwin, ywin* - the correlation window sizes, *maxmot* is the maximum displacement in pixels that is expected, *rangebeg* and *rangeend* are the range of pixel values to be used, *overlap* is the overlap between neighboring windows in pixels, *timediff* is the time difference between the images in hours, *pixx* and *pixy* are the pixel dimensions in km, t*hresh* is the cross-correlation threshold below which the correlations are disregarded. The apparent velocity vectors are saved in a file and can be overlaid on the current bitmap with File - Vectors. Note: Do not forget to set the vector file format with *View - Settings - Misc* beforehand!

Warning: As this operation is very compute-intensive, it is prohibitively slow on large images even on fast PC-s. Standalone versions of the program (see Emery et al., 1991) should be used for more extensive calculations.

Multiply

Multiplies the current image to another image pixel wise. The resulting image is always a float image (4 bytes per pixel). The *Value Scaling* options (see *View - Settings - General -* Value Scaling) are used to calculate the real pixel value before the multiplication.

Overlay Image

Overlays another image on the current image. Pixels assume values of the current image if the corresponding pixel on the second image is zero, and values of the second image if the corresponding pixel on the second image is not zero. Prompts for the overlay image index. If the index number is non-negative, a new image buffer is allocated and the result is put there. If the index number is negative, overlaying will be done in-place and the absolute number is used as the overlay index. This operation could be used, for

example, to overlay coastlines, tracks, etc. on the current image. As an example, load (*File* - Open -. Image) a sample image *est256.img*, then load the overlay image (*File - Open - Overlay*) *est256.ovl*, switch back to *est256.img* and then run *Overlay Image* by giving the sequence number of the *est256.ovl* image buffer.

Polarization Ratio

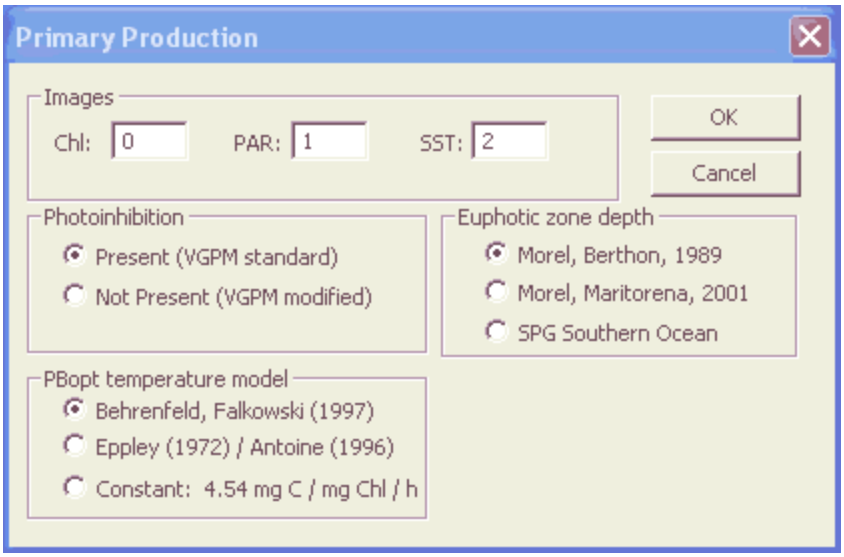
Performs the (TBH - TBV)/(TBH + TBV) operation on pixel values of 2 images (TBH and TBV, respectively) and saves it in a new image buffer. The operation is used for the horizontally and vertically polarized microwave signals (see Chapter 7 on SSM/I Products). To prevent division by zero, the result is set to 1.0 if TBH + TBV = 0.

Primary Production

Calculates an image of phytoplankton net primary production (NPP, mg C/m^2/day) based on the Behrenfeld and Falkowski (1997) Vertically Generalized Production Model (VGPM) and using images of surface chlorophyll *a* concentration (Chl-a, mg m^{-3}), incident photosynthetically active radiation (PAR, Einstein m^{-2} Day^{-1}) and sea-surface temperature (SST, °C). Please see the following options dialog box for *Primary Production* calculations. Currently there are options for photo-inhibition, temperature model of P^B_{opt} and euphotic zone depth calculation. More options will be added in the future. The original VGPM model used the Morel and Berthon (1989) euphotic zone depth parameterization as a function of surface chlorophyll. That model was later revised by Morel and Maritorena (2001). In the Southern Ocean mixed layer is deeper at similar chlorophyll concentrations and a different parameterization (Mitchell, Kahru, unpublished) is available as an option.

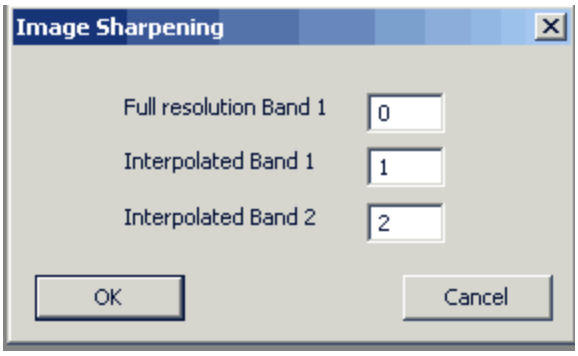
Global and regional images of Chl and PAR are available, among others, from the Goddard DAAC http://oceancolor.gsfc.nasa.gov/ftp.html) and SST images are available, among others, from the PO DAAC (http://podaac.jpl.nasa.gov/sst/). It is assumed that the sequence of images is: Chl-a, PAR, SST and the user has made the Chl-a image the current image (by clicking on it). However, the actual sequence of images can be adjusted in the dialog box. The function needs to calculate the day length for each pixel. If the Chl-a image has attributes “Start Day” and “End Day” as the standard SeaWiFS, OCTS and MODIS images have, the program uses the middle day of the start and end times to calculate day length. If the Chl-a image does not have those attributes, the program prompts for the Julian day (e.g. January 10 of any year is Julian day 10) of the calculation period. The function creates a new image of int16 NPP values with the scaling parameters of *Slope* = 1.0 and *Intercept* = 0.0. **Please check out the attributes** ( on the Toolbar) of the calculated image with View - Attributes. It is essential that all three input images are of the same size and projection. The VGPM model was developed for monthly images but it is also possible to apply it to shorter time intervals, e.g. 8-day composites. The function produces results only for pixels that are valid for all three of the input images. As the SST images may have large areas blocked due to cloud

contamination when using short compositing period, it may be advisable to use longer compositing (e.g. monthly) for SST even when using a shorter (e.g. 8-day) Chl-a images. SST usually does not usually change very rapidly and the influence of small variations in SST on the calculated product is usually of the order of 10% that is well below the typical uncertainty in the NPP calculation.



Sharpening

Resolution sharpening uses one band with higher spatial resolution to increase the spatial resolution of another band. For example, the MODIS sensors have bands 1 and 2 (620-670 nm and 841-876 nm) at 250 m resolution, bands 3-7 at 500 m resolution and bands 8-36 at 1000 m resolution. Assuming consistent ratios between measured and interpolated bands it is possible to create fake 250 m resolution images from bands 3-36. For example, in order to create a fake MODIS band 3 image at 250 m resolution we need band 1 image at 250 m resolution ("Full resolution Band 1"), band 1 sampled at 500 m and interpolated (with linear interpolation) to 250 resolution ("Interpolated Band 1"), band 3 sampled at 500 m and interpolated to 250 m resolution ("Interpolated Band 2"). The simple algorithm used here assumes that the ratios of full-resolution and interpolated (i.e., smoothed) bands are constant within each spectral band. More details on how to perform resolution sharpening of MODIS bands can be found in the "Practical exercises with WIM and WAM".



Shift Image

Sometimes you may notice that when you create a coastline overlay for an image and lay it on top of the image, the image is shifted relative to the coastline (or vice versa). With this function you can shift the coastline relative to the image so that it fits the image. When you are satisfied with the fit, the image is shifted by a discrete number of steps in X and/or Y direction. This is a very simple geometrical correction and is useful only when the shift is relatively small and uniform for the whole image. In order to fit an image to a "standard" overlay (e.g., coastline), the overlay (a contoured image) is moved in the *X* and *Y* directions over the image until the user accepts the fit by pressing Enter. The overlay contour is taken as the template according to which the image will be corrected. It is inevitable that when shifting the image inside its frame, some borders of the image will have no values and will be converted to zero. This operation is used when you have a set of images of the same area that you want to make to match each other as good as possible. You need to have a standard overlay (contour map) of the area or you have to create the standard overlay image yourself from the contours of one of them (using edge detection, thresholding, zero-crossing, etc.). This overlay will then be used as a reference according to which all the other images are shifted. To use the operation, make the image that you want to shift as the current one (by clicking on it), and have the overlay also in a memory buffer. After you have specified the overlay index (the number of its memory buffer), the contours of the overlay will be drawn as white pixels on the current image. By pressing the arrow keys the overlay is shifted by one step in the respective direction after each key action. Combining the arrow key with the Shift or Control key makes a shift of respectively 5 or 20 pixels in the corresponding direction. When pressing PageUp or PageDown keys the shift is by 10 pixels up or down. When pressing Home the shift is 20 pixels up and 20 pixels left. When pressing *End* the shift is 20 pixels down and 20 pixels right. To end the positioning, press Enter, ESC, or Space. You may now choose to rebuild the image using the shifting information, and save the corrected image for later use.

If you already know the shift that you want to make, instead of the overlay index (buffer number) specify a negative number or a number that is larger than the last image buffer. Positive number for a shift in the *X*-direction means shifting to the right, positive shift in the *Y*-direction means shifting down.

SST (ch4, ch5)

Calculates the sea-surface temperature image from AVHRR channel 4 and channel 5 images. It assumes that the current image is the channel 4 image and prompts for the channel 5 image number. The calculated SST image is put into a newly allocated image buffer. The source image values are interpreted depending on the *Value Scaling* flag in the *View - Settings - General* dialog box: if either *Pixel Value* or *x 10* is selected, the image values are assumed to be temperature values multiplied by 10; if *Value Scaling*: *SST-SMHI* is selected, it is assumed that the *SST-SMHI* transformation is used (see *Transf -* Convert *).*

The brightness temperatures of a single channel are influenced by changes in the atmospheric water vapor. The split-window method (McClain et al., 1985) uses the values at two different channels to correct for the atmospheric interference. Here the coefficients estimated by Coll et al. (1991) are used:

```
T = T4 + [1.54 + 0.22(T4 - T5)] * (T4 - T5)
```

where *T* is the true sea-surface temperature, *T4* and *T5* are, respectively, the brightness temperatures in AVHRR channels 4 and 5.

Subtract Image

The operation is similar to *Multi – Difference*. It subtracts a specified image from the current image pixel-wise. The images must be of the same size. *Value Scaling* (see *View - Settings - General*) is used for calculating the real value before subtraction. The resulting image is created as a 4-byte Float image that is usually not the best option but is the most versatile. In most cases is convenient to scale the resulting image into a Byte or Int16 image using the *Transf – Convert* function.

Turbidity (ch1, ch2)

Calculates the turbidity index from 2 visible band images according to the algorithm of Stumpf (1992). Remote sensing data is often contaminated by the interference of the atmosphere. This function has been adapted from the program *DECCON* by Townsley and Stumpf and is intended to obtain an image of water reflectance corrected for some atmospheric effects. As the water-leaving reflectance is almost zero in the near-infrared band 2 of AVHRR in all but the most turbid coastal waters, it is possible to correct for some of the aerosol effects by subtracting a modified band 2 data from the modified band 1 data. After the subtraction there remains a residual bias (primarily from the Rayleigh radiance) that can be removed by the user by subtracting a value from the entire scene so that the clear water in all scenes have the same value. This operation assumes that the Rayleigh radiance is relatively stable over the scene.

For CoastWatch images all the necessary auxiliary data are retrieved from the image header. The current image is assumed to be the channel 1 image and the user is prompted for the channel 2 image number.

If used for non-CoastWatch images the user has to provide the necessary auxiliary data. The solar zenith angle needed for this function is calculated for

the center of the image based on the timing of the image (valid for relatively small images). For the timing, the Julian day, GMT hour and minute as specified in *View* - Attributes are used. If the Julian day is 0, it will be calculated from the year, month and day. The year is specified with the last 2 digits, e.g. 94 for 1994.The longitude and latitude range (*View - Attributes*) is used to estimate the geographical coordinates of the center of the image. The default optical depths due to the Rayleigh scattering (τ_r) and gaseous absorption (τ_g, i.e. for ozone and water vapor) are estimated by Stumpf and are the same as in the DECCON program. A new 1-byte/pixel image buffer is created with the turbidity index values. The output is always a 1-byte-per-pixel buffer. For 1-byte/pixel input buffer the correction is not performed for values 0 and 255, i.e., for out-of-range values. When calculating the output, pixels are scaled according to the albedo minimum and maximum values (*View* - Attributes). If no albedo range has been specified, the range is assumed to be from 0.0 to 25.5 and the corrected albedo values are multiplied by 10.0 .

Vegetation Index

Finds the Normalized Difference Vegetation Index (*NDVI*) from two images. It is assumed that the current image is from AVHRR channel 1 (0.58-0.68 µm) and another image is channel 2 (0.725-1.10 µm). *NDVI* is calculated according to the formula *NDVI = (Ch2 - Ch1) / (Ch1 + Ch2)* and ranges from -1 to 1. A new image buffer is allocated with the corresponding pixel values equaling *100 * NDVI + 100*.

NDVI is a rough index of the amount of green plant biomass. In general, *NDVI* is negative for water, near zero for clouds and bare soil, and changes between 0.05 and 0.6 for vegetative surfaces (Holben, 1986). *NDVI* has been used to detect thick surface phytoplankton blooms (green soup) of the blue-green algae (Prangsma and Roozekrans, 1989). As *NDVI* for water surface is normally negative since water is nearly a black body at near-infrared (channel 2), positive values indicate dense accumulations of surface floating algae. In fact, positive values result only from very dense accumulations and even small negative numbers indicate surface floating algae. In order to show negative pixel values, 100 is added to *NDVI* values multiplied by 100. *NDVI* values less than -0.1 are converted to zero. For testing the operation, you can calculate the NDVI image for a sample pair of images (channels 1 and 2) from *sbaltic.lan* (use *File- Open - Erdas/Lan* to read it). Note that due to the transformation, the pixel values below 100 are negative. As another example, try it with sample images *s_calif1.img* and *s_calif2.img* of Southern California. You can see reduced vegetation density or “greenness” in the Los Angeles and San-Diego/Tijuana urban areas. These sample images were produced with the CCAR navigate program (Baldwin and Emery, 1993; Emery, 1995).

6.8 Edge

Shade-Edge

Finds edges according to the Cluster shade method of Peckinpaugh (1991) and Holyer and Peckinpaugh (1989).

Contours

Finds contours of pixels between low and high values ("zero-crossing"). As Byte images are represented by unsigned bytes with values from 0 to 255, it finds crossings over 128. In order to find isolines corresponding to a certain pixel value, you should first binarize the image with that value (*Transf - Binarize*). Then use *Contours* with default values. If the contours look too thick, you can use *Edge - Thin Lines*. Prompts for the initial threshold (*Init_Thr*), minimum threshold (*Min_Thr*), and the maximum number of passes (*MaxPass*). If a pixel has a value that is at least *Init_Thr* away from 128, and any of the 8 neighbors at least *Init_Thr* away from 128 but on the other side, the pixel is marked by 255 and considered an edge. If *MaxPass > 0*, additional passes are made attempting to extend the found edges. Pixels that are not marked as edges but have edge pixels among one of their 8 neighbors, are tested if they are crossing the 128 border with *Min_Thr*. The operation is repeated until no more pixels are found, or the maximum number of passes has been made.

Clean Edge

Cleans an image of isolated small blobs of non-zero pixels. Prompts for the size of the window where the test is made. All isolated blobs of pixels that do not cross the window border will be converted to zero.

Dilate

Dilates (expands) non-zero pixels (e.g. edges) in all directions trying to connect broken or disconnected parts of an assumedly continuous contour (edge).

Thin Lines

Uses the sceletonization algorithm of Pavlidis (1980) to thin the dilated edges by peeling off the unnecessary layers of edge pixels.

SIED

Uses the Single Image Edge Detection (SIED) method of Cayula and Cornillon (1992) with the variable window (VW) modification of Diehl et al (2002). It works with multi-byte images but the unscaled pixel values must be between 0 and 255. It is recommended to use only with Byte images. If the source data is not Byte then you should convert the to Byte using *Transf-Convert* and a suitable scaling method (e.g. SST Pathfinder for SST data). The user can select a fixed window size for edge detection or accept 0 for the variable window size of Diehl et al (2002). Bigger window size detects larger-scale fronts. A typical fixed window size is 32 (Cayula and Cornillon, 1992). If the fronts look too thick then you can use the *Edge-Thin Lines* after Edge-SIED.

6.9 Segm – Segmentation

This section contains routines for edge detection as well as segmentation (classification) of an image into different sub-areas (segments). The segmentation algorithms have been developed for classifying a highly variable (speckled) synthetic aperture radar (SAR) image and are not designed to work effectively with other, less-variable images where simple filtering and thresholding could yield better results. **As these routines have not been updated for a long time – please use with caution!**

Typical classes for a SAR image are: open water, even ice, uneven ice. A sequence of segmentation routines is usually the following (e.g., Skriver, 1989): edge detection, edge thinning, edge linking, edge distance calculation, kernel finding, kernel connecting and kernel growing-pixel tracing. The routines in this section are mostly following the ideas in the paper of Sun Yan et al. They should be considered 'experimental' and a lot of experimenting is needed on behalf of the user to get an acceptable result.

Find Edges

Finds edges and edge directions by the gamma-ratio method (Madsen, 1986). The method compares ratios of pixel values in all possible directions around each pixel and assigns the maximum ratio value to the edge image pixel. The corresponding edge direction image is saved in the edge direction image with the following coding: 1 - E-W, 2 - NE-SW, 3 - N-S, 4 - NW-SE, 5 - W-E, 6 - SW-NE, 7 - S-N, 8 - SE-NW. The first direction corresponds to the higher side in the window. Warning: this is a compute-intensive operation and may take a long time on a PC without a math coprocessor. The best window size depends on the variability of the data. Small window size gives sharper edges but may be strongly influenced by the local noise.

Thinning

Uses the edge value and direction images from the previous operation to create a new image with thinned edges. This operation may be iterated several times to produce acceptable results.

Threshold

Using the specified threshold value, binarizes the image as 1 (equal and larger than the threshold), i.e. edge pixels, and 0 (less than the threshold). This is the same operation as *Transf - Binarize* that is simply duplicated here for the convenience of the user. You have to experiment in order to find the best level of separating edges from noise.

Clean

This operation eliminates (transforms to zeros) isolated edge pixels or small blobs of pixels. It can be used as well with other binary images to eliminate

disconnected pixels. It is the same routine as *Edge - Edge Clean* that is simply duplicated here for the convenience of the segmentation user.

Connect

Allows to connect disconnected edge pixels. It uses an image of edge pixels and looks for loose ends among edges (edge pixels are different from zero and non-edge pixels are zero). It then reads tries to continue the loose ends along the gradient image (actually, perpendicularly to gradients). You therefore need to have the edge (gradient) direction image as well in memory (it was produced during edge finding (*Edge Find*). Prompts for the maximal number of pixels that will be traced in both directions along the assumed edge direction. If another edge pixel is found, the intermediate pixels are converted to new edge pixels (value 128, different from the original edge pixels = 255). If image border or is reached, or the maximum number of gaps traced has been reached without reaching another edge - nothing is converted. Quite often this routine does not produce acceptable results as the gradients may be followed in parallel to another edge. You can then try *Dilate* and *Thin Line* operations from the *Edge* menu section, or may wish to skip this operation at all.

Distance

From an edge image creates a new image of distances to the nearest edge pixel using a method called chamfering (Borgefors, 1983). It includes thresholding where you can specify the threshold level between edge and not edge pixels. Normally edge pixels are set at 255 and not edge pixels at 0. If the distances get larger than 255, the routine has to scale them in order to be able to store the values in an 8-bit image. The distances are transformed by taking square root of them, and then scaled from 0 to 255.

Kernel Find

Finds local maxima, i.e., kernels for segmentation, and the gradient directions. Makes 2 new images - the kernel and the gradient direction image. The gradient direction image is later used for connecting adjacent kernels as well as for growing the kernels and tracing the pixels to kernels. The gradient direction is coded as following: 1 - right, 2 - up-right, 3 - up, 4 - up-left, 5 - left, 6 - down-left, 7 - down, 8 - down-right (the direction points to the higher side).

Kernel Combine

Is used to combine closely located kernels (maxima) into the same class (segment). This routine uses the kernel images from the previous operation and the distance image to the nearest edge (*Edge Distance*). It tries to connect only kernels for which to sum of separation in rows and columns is less than 100. The criterion for linking two maxima *m1* and *m2* is: *dist(m1, m2) < (dt(m1) + dt(m2)) * a*, where *dt* is the distance transform from the nearest edge (*Edge Distance*) of the pixel, and *a* is a coefficient (initially 0.6). The final number of different kernels must be less than 255 (value 0 is

assumed to represent a non-kernel pixel). If the number of different kernels after combining is still >= 255, the routine increases a to connect more kernels.

Kernel Grow

Uses the combined kernel image (must be current) and the gradient direction image of the edge distance (*Kernel Find*) to grow the kernels. It should be used once.

Pixel Trace

Should be used after *Kernel Grow*. It uses the same source and gradient image as *Kernel Grow* and tries to convert non-kernel pixels to kernel pixels by following each pixel along it's gradient direction. If non-classified (zero or black) parts are left after it, this operation can be repeated several times in a row.

Fill Holes

Replaces zero-valued pixels with the mean value of neighboring non-zero pixels. Invalid pixels are most often assigned a value of 0. If the missing pixels have a value different from 0 (e.g., 255) then they need to be converted to zero before *Fill Holes* can be used. This function allows to fill small “black holes” due to small clouds, image speckle or other noise. Can be run iteratively to fill bigger areas. Although there is no real substitute for missing data, this function can be used to fill small areas of no data. Be careful when using floating point pixel values as very small positive or negative values are still different from 0. As an unwanted consequence, the function expands the valid pixels into areas where there should not be valid pixels, e.g., ocean values over land. To fix the latter problem one can use Multi-Mask w. Image to put a proper land or another mask on top of the modified image.

Set Segments

Uses the segmented image from *Pixel trace* and the original source image to produce two new images where pixel values are set, respectively, to the average and standard deviation for each segment. If there is memory enough to open only one new memory buffer, it skips the standard deviation and produces only the segment average image. If you wish to exclude the edge pixels (or any other set of pixels) you can do that as the routine only counts not zero pixels when calculating average and standard deviation. Get the edge image that was used to generate the edge distance image (you may dilate it to exclude even bigger surroundings of edges), binarize it (*Transf* - Binarize) with a negative value (e.g. -128), turn back to the original source image and mask it (*Multi* - Mask w. Image) with the newly created negative edge image. Now use this image as the source image when using *Set Segment* and you get the segments set to the values of averages and standard deviations over their interiors.

6.10 Window - Window Arrangements

Cascade

Tile

Arrange Icons

6.11 Help - WIM Help

Index

Get an index of WIM help topics.

Using Help

Instructions for using Windows Help.

 **About WIM**

Displays info about WIM, including the version number and the licensed username.

License

Allows to insert your final license number.

7 SSM/I Products

The National Snow and Ice Data Center (NSIDC) (CIRES, University of Colorado at Boulder, Campus Box 449, Boulder, CO 80309-0449, USA, http://www-nsidc.colorado.edu/) has made available a huge data-set of, among others, daily North and South polar brightness temperatures, sea ice concentration grids, and monthly averaged sea ice concentrations obtained from the SSM/I sensor. The SSM/I is a seven channel, four frequency, linearly polarized, passive microwave radiometric system and is part of the Defense Meteorological Satellite Program. In order to facilitate the use of these datasets a few additions were made to the WIM program. The NSIDC SSM/I data products are in four specific grids in the polar stereographic projection: n3a*, n3b, s3a, s3b, respectively for the Northern and Southern polar regions*. These grids (projections) are automatically recognized by WIM when reading the corresponding HDF data file.
A specific operation (*Multi* - Polarization Ratio) was added for the horizontally and vertically polarized microwave signals.

8 SeaWiFS Products

SeaWiFS standard products are delivered in HDF format. Image data in HDF is delivered in raster-8 (2-dimensional images with 1 byte per pixel) and scientific data sets (SDS). WIM can read both raster-8 and SDS data types and display them as images. Special enhancements are included to make it easier to work with SeaWiFS standard products from the Goddard DAAC (http://daac.gsfc.nasa.gov) and the SeaWiFS Project (http://seawifs.gsfc.nasa.gov/SEAWIFS.html).

When reading Level-1A products WIM reads the radiance counts in selected bands out of the total of 8 SeaWiFS bands (corresponding to center wavelengths of 412, 443, 490, 510, 555 and 670 nm) and the attribute information. The projection of Level-1 images is automatically set to Swath and the projection parameters are read from the HDF file.

When reading Level-2 standard products, WIM sets the Value Scaling options according to the value scaling information in the file. E.g., the water-leaving radiances are scaled with *Slope* = 0.001 and *Intercept* = 0.0, chlor_a is scaled with *Slope* = 0.001 and *Intercept* = 32.0. Please note that the pixel values are signed 2-byte integers. The color scaling values for the Level-2 chlor-a and CZCS-pigment images are automatically set to -32000 and -29000, respectively (see *View - Settings* - Color Scaling). This corresponds to the concentration range of 0-3 mg m^{-3}. Anything above 3 mg m^{-3} is white. If your range of interest is higher than that you have to manually set a higher value to the Max value, e.g. –22000 corresponding to the concentration of 10 mg m^{-3}. The projection of Level-2 unmapped products is automatically set to Swath and the parameter values are read from the HDF file. This allows to create coastline overlays, latitude-longitude grids and other geo-referenced images for the Level-2 files. WIM allows to discard arrays narrower than the specified number of pixels (*View - Settings – HDF Options - Minimal width of image to be read*). This prevents reading arrays of various calibration coefficients and other auxiliary information as images.

SeaWiFS Level-2 products include **Level-2 flags**. Each of these flags indicates certain conditions. The number of flags was increased from 16 to 32 in reprocessing number 3 (spring 2000). Some of the flags are not used (spare). The values of the flags that are ON (1) can be viewed by right-clicking on the image. These flags are then listed in the window header. Alternatively, areas where selected flags (bits) are ON can be shown by creating mask images with *Transf – Bitmask*.

SeaWiFS Level-2 flag bit numbers, along with their associated masking keywords are:

1. atmospheric correction algorithm failure,	ATMFAIL
2. land,	LAND
3. missing ancillary data,	BADANC
4. Sun glint,	HIGLINT
5. total radiance greater than the knee value,	HILT

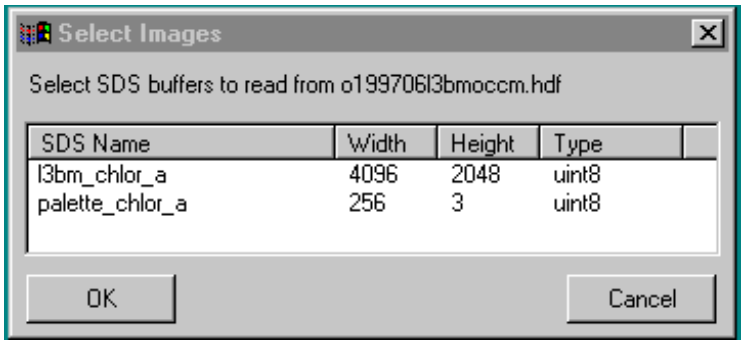
6. large sensor zenith angle,	HISATZEN
7. shallow water,	COASTZ
8. negative water-leaving radiance,	NEGLW
9. stray light,	STRAYLIGHT
10. cloud or ice,	CLDICE
11. coccolithophores,	COCCOLITH
12. Case 2 (turbid) water,	TURBIDW
13. large solar zenith angle,	HISOLZEN
14. high aerosol concentration,	HITAU
15. low water-leaving radiance at band 5,	LOWLW
16. chlorophyll algorithm failure,	CHLFAIL
17. questionable navigation,	NAVWARN
18. absorbing aerosol,	ABSAER
19. tricodesmium,	TRICHO
20. NIR algorithm exceeded maximum iteration,	MAXAERITER
21. moderate sun glint,	MODGLINT
22. chlorophyll < 0.01 or > 64,	CHLWARN
23. epsilon out of range,	ATMWARN
24. dark pixel,	DARKPIXEL
26. spare,	SPARE
27. spare,	SPARE
28. spare,	SPARE
29. spare,	SPARE
30. spare,	SPARE
31. spare,	SPARE
32. ocean,	OCEAN

When reading Level-3 *Standard Mapped Images* WIM automatically sets the projection to *Global Equal Angle* and appropriate Value scaling. Both the *browse* and the 9 km resolution (full) files can be viewed. The Value Scaling used for both chlor_a and CZCS_pigment data is *Log-Chl*. You can then use *Geo* - Get Map Overlay to generate coastline and other overlays in high or low resolution.
Among the **mapped** SeaWiFS images other than the *Global Equal Angle* only the SeaDAS Cylindrical projection with the parameters *Latitude Center*=0, *Rotation*=0, *Position*=[0, 0, 1, 1], *Isotropic*=0, *Scale*=0 can be navigated in WIM. This SeaDAS projection is automatically converted to the WIM *Linear* projection with correct parameters.

A wealth of information on SeaWiFS images is included as global or local attributes - you can view this information with *View* – Attributes .

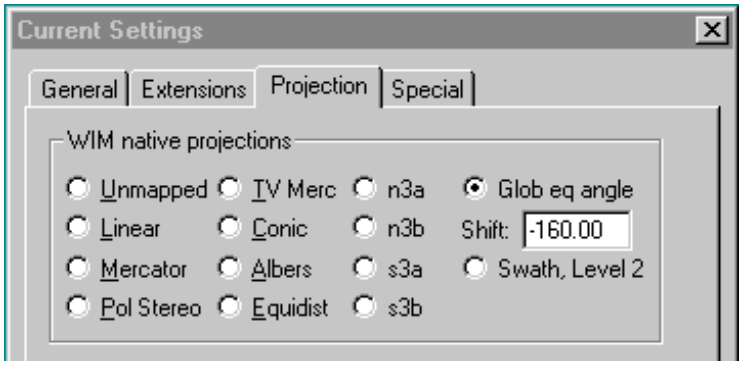
9 OCTS Products

OCTS standard products are delivered in HDF and are very similar to the respective SeaWiFS products. In Level-3 mapped chlor_a images both the image and the palette are delivered as scientific data sets (SDS):



You can skip the palette as it is only 3 pixels high.

When reading Level-3 mapped images Value Scaling for both chlor_a and CZCS_pigment data is set to *Log-Chl*. For global images in *Standard Mapped Image* format set the *Projection* type to Global Equal Angle (*Glob eq angle*) in *View - Settings* - Projection.



In contrast to similar SeaWiFS maps the OCTS maps are shifted by –160 degrees. You must type it into the Shift field.
You can then use e.g. *Geo* - Get Map Overlay to generate coastline and other overlays in high or low resolution.

A wealth of information about the image is included as attributes (metadata) and can be viewed with *View* - Attributes.

The OCTS global GAC dataset was reprocessed in 2001 by NASA and is available at http://seawifs.gsfc.nasa.gov/cgibrs/octs_browse.pl. The format of these files is identical to those of SeaWiFS, e.g. no spatial shifting is necessary.

10 MOS Products

The Modular Optoelectronic Scanner (MOS) was a spaceborne imaging pushbroom spectrometer in the visible and near infrared range of optical spectra (400 - 1010 nm) which was specially designed for remote sensing of the ocean-atmosphere system. MOS-PRIRODA and MOS-IRS instruments were basically identical providing 17 spectral channels with medium spatial resolution in the VIS/NIR. The advanced instrument built for the IRS spacecraft has one additional channel in the SWIR at 1.6µm. More information on MOS can be found at http://www.ba.dlr.de/NE-WS/ws5/.
MOS data products of Level 1b are available in the HDF format. Both MOS-A and MOS-B data files contain a 1 byte per pixel quick-look image and radiometrically corrected and calibrated, 2 byte per pixel, band interleaved by line data from all the channels. Special functions in WIM unpack the band interleaved format automatically into images.

Two new functions were added to WIM together with the support for reading MOS images: *Multi* - Band Ratio and *Multi* - Linear Comb. These functions are applicable to any images but may be useful for analyzing MOS data in particular.

11 MODIS Products

Overview

Moderate Resolution Imaging Spectroradiometer (MODIS) is a key instrument aboard the Terra (EOS AM-1) and Aqua (EOS PM-1) satellites. MODIS is viewing the entire Earth's surface every 1 to 2 days, acquiring data in 36 spectral bands. For an overview of the instrument and science, see About MODIS.

MODIS data is available at the Distributed Active Archive Centers (DAAC's). These data are divided into land, ocean, atmosphere products. For information about data availability, products, and sources, see the Data section.

MODIS has hundreds of different data products (see http://eosdata.gsfc.nasa.gov/CAMPAIGN_DOCS/MODIS/index.shtml). As the default format of these data products is HDF, WIM can read most of them.

Ocean Products

The Mapped Level-3 products are easiest to work with. They are provided in spatial bins of 4 km, 36 km and 1 degree and temporal bins of 1 day, 8 days, month, year (see http://acdisx.gsfc.nasa.gov/data/dataset/MODIS/03_Ocean/03_Level3_Mapped/index.html). Monthly and other composites can be downloaded or generated with WIM (Multi – Composite).

MODIS Level-2 products are in the highest possible resolution (1 km for ocean channels, 250 or 500 m for some land channels) but have the disadvantage of being in the satellite-view projection. In contrast with SeaWiFS Level-2 products that include algorithms for calculating the latitude and longitude of each pixel, MODIS provides a separate product MOD03 that provides the many ancillary parameters such as the zenith and azimuth angles for the sensor and the sun, satellite location, etc. Most significantly, the latitude and longitude arrays are provided in MOD03 files for the corresponding MODIS Level-2 products. The latitude and longitude arrays (LLA) can also be supplied in the MODOCQC product group and GSUB subset files. After selecting the Scientific Datasets to be read from MODIS Level-2 products, if the latitude and longitude arrays LLA) are not found in the same file, the user has to select a file where the LLA (e.g. the corresponding MOD03 or QC product). This can be quite a challenge as the filenames are very long and hard to memorize. Starting with WIM version 6.15 a smart algorithm is being used to help the user to locate the right file with the LLA. The algorithm uses the "longest common substring" to suggest the best matching filename with the LLA. For example, having the following files in the folder and trying to load a dataset from the first file, we need to select the third file in this list for geo-location (LLA) and not the second file in the list. The algorithm uses the "longest common substring" to suggest the best matching filename with the LLA.

```
MYDOCL2B.A2003168.2115.003.2003179212054.hdf  <== getting data from
MYDOCL2B.A2003181.2045.003.2003192152344.hdf  <== similar data file
```

```
MYD03.A2003168.2115.003.2003170050731.hdf      <== file with LLA
```

This seems easy when we have just a few files in the folder but becomes overwhelming with many similar files with long names. Starting with WIM version 6.17 this algorithm was improved by eliminating some file types that are known to not have the LLA.

Most MODIS Level-3 mapped products and Level-2 products are scaled as 16-bit signed integers with a linear scaling: Slope = 0.001 and Intercept = -5.0. The default color scaling range for these provided by WIM is 5000 to 10000 in pixel values corresponding to 0.0 to 5.0 in the geophysical value. The user can change these default values in the *Current Settings* dialog box. The 16-bit integer images can be converted to 8-bit unsigned integer (byte) images with the *Transf – Convert* function.

The function *Multi-Mask w. Image* has a flexible masking that corresponds to the MODIS *Quality* image. The default values (0 for mask value), and "Yes" for smaller and equal of the *Multi – Mask w. Image* function are suitable for picking the "good" pixels of MODIS images. MODIS data are delivered with a corresponding Quality flag image that has pixel values corresponding to the quality flag as follows:
0 – good
1 – questionable
2 – cloud
3 – bad
This meaning of the flags here is given as an example only and may depend on a particular convention and data type. The default selection would selecting pixels with Mask value =< 0, i.e. "good". When selecting pixels with mask value >= 3, for example, only the pixels of quality "bad" are returned.
For screening multiple images you can use the WAM program *wam_screen* that processes all selected file in a folder, picks only pixel values with 0 quality and saves the screened images in new files. The program can be modified to perform other functions, e.g. pick different quality pixels.

Radiances and Geo-location

Creating true color images

Both MODIS-Terra and MODIS-Aqua instruments can produce great looking true color images. WIM has several tools and functions for that. As this involves operations on multiple bands, it is better handled with the WIM Automation Module (WAM). Please see the WAM manual (*WAM.pdf*) and the WIM and WAM exercises for examples.

Atmosphere Products

Level-2 products

Aerosol products, Water Vapor products, Cloud products, Atmosphere Profiles, Cloud Mask products, Joint products: recent data can be downloaded from the Data Pool at http://daac.gsfc.nasa.gov/data/datapool/MODIS/02_Atmosphere/01_Level_2/index.html

Level-3 products

Only a small selection of products is listed here. For example, the Atmospheric Monthly Global Joint Product contains 739 statistical datasets that are derived from the Level 3 Daily Global Joint Product (**MOD08_M3**). Recent data can be downloaded from the Data Pool at http://daac.gsfc.nasa.gov/data/datapool/MODIS/02_Atmosphere/02_Level_3/index.html.

Land Products

Level-2 products

Level-3 products

Many products, e.g. Vegetation Indices 16-Day L3 Global 1 km data (**MOD13A2**) are available from http://edcdaac.usgs.gov/dataproducts.asp.

MODIS Data at NSIDC

The National Snow and Ice Data Center holds many MODIS products on snow and ice at http://nsidc.org/data/modis/data.html.

Snow Cover products

For example, MOD10A2 has 8-day Terra snow cover data at 500-m resolution in the Sinusoidal projection.

Sea Ice products

For example, MOD29P1D has daily Terra sea ice extent data at global 1-m resolution.

12 GLI Products

GLI (Global Imager) was an advanced sensor aboard the MIDORI-II (ADEOS-2) spacecraft of the Japanese Space Agency (http://sharaku.eorc.nasda.go.jp/GLI/index.html). MIDORI-II was launched on December 14, 2002 and the initial data products looked very promising. Unfortunately the spacecraft was lost on October 25, 2003. Special functions for all levels (from 1 to 4) are being developed for WIM. The GLI Level-1b and Level-2 products are somewhat similar to MODIS respective products. For example, the Latitude/Longitude arrays (LLA) are in a separate file. Starting with version 6.14 WIM can display and navigate GLI Level-1 and Level-2 products. Selecting the matching file with the proper LLA can be quite a challenge as the filenames are very long and hard to memorize. For example, having the following files in the folder and trying to load a dataset from the first file, we need to select the second file in this list for geo-location (LLA) and not any other file in the list.

```
A2GL103040847120D2_OCSFR020000000040A000.00  <== getting data from
A2GL103040847120D2_PS1BC431030.00            <== corresponding LLA
A2GL103041647110D1_PS1BC431030.00
A2GL103041647120D2_OCSFR020000000040A000.00
A2GL103041647120D2_ONLFR020000000100000.00
```

This seems easy when we have just a few files in the folder but becomes overwhelming with many similar files with long names.

A smart algorithm is being used to help the user to locate the right file with the LLA. The algorithm uses the "longest common substring" to suggest the best matching filename with the LLA.

The Level-3 binned and mapped data are similar to the SeaWiFS and other SMI (Standard Mapped Images) products. The main difference is that the left side of the image is at the 0 meridian and not at -180. WIM is automatically setting the *Global Equal Angle* projection with *Shift* = 180 (see Projections).

More specialized functions for GLI data will be added in the future as different data products become available.

13 Landsat Products

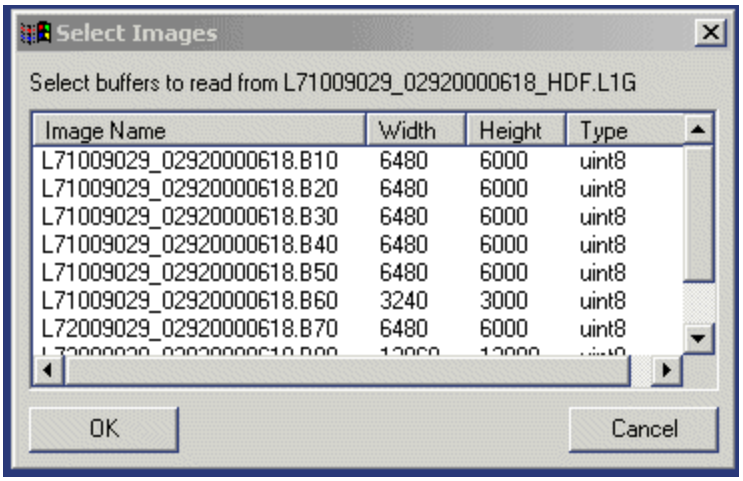
The Landsat Program is the longest running enterprise for acquisition of imagery of the earth from space. The first Landsat satellite was launched in 1972; the most recent, Landsat 7, was launched on April 15, 1999. These sensors typically provide high-resolution (30 m) imagery. The resolution is enhanced (15 m) for a wide-band channel and reduced (60 m) for the thermal infrared channel. The main disadvantage of the Landsat imagery has been its high cost.

In cooperation with federal and provincial governments of Canada, GeoGratis (an initiative of the Canada Centre for Remote Sensing) is distributing full resolution Landsat 7 imagery for all of Canada free of charge. Landsat 7 data of selected scenes of Canada is freely available for downloading at http://geogratis.cgdi.gc.ca/Landsat7/hdf/ . New scenes will be accessible when they become available.

The Landsat 7 satellite carries the enhanced thematic mapper plus (ETM+) sensor. Landsat 7 data are collected from a nominal altitude of 705 kilometers in a near-polar, near-circular, Sun-synchronous orbit, imaging the same 183-km swath of the Earth's surface every 16 days. The L1G product is a radiometrically and systematically corrected L0R image.

The format of individual bands is simple binary raster with 1 byte per pixel (WIM Image). In order to use *File – Open as Image* you typically need to know the size (*DX, DY*) of the image. The size, geo-referencing and other information is available under the *Description files* link. By clicking on that link you can download a compressed set of files (**.tar.gz*) that you can uncompress and *untar* into the same directory where the uncompressed image files are. A particular file *productdescription.self* has all the metadata about the particular Landsat scene. You can get the size of the images from there but a better way is to read as HDF files the “launch-pad” HDF file *(*.HDF.L1G*) that has all the links to the individual band images. Just make sure that **.HDF.L1G* is listed as one of the extensions of HDF files in Current Settings - Extensions. If not, please add it by appending “*;*.HDF.L1G*” into the HDF box.

Geo-referencing of these files is currently done using the upper-left and lower-right coordinates in the *productdescription.self* file corresponding to the particular Landsat scene. A *Linear* projection with corresponding coefficients is automatically produced. Due to various non-linearities and errors this geo-referencing is not very accurate and better methods may be introduced in the future.



Above is a typical dialog box for selecting bands from the “launch-pad” HDF file. As you can see the image bands are of different size due to the different resolution (30 m vs. 60 m vs. 15 m). A commonly used option for these images is to create a RGB-composite of a selected set of three bands. When making a RGB composite the component bands have to of the same size. You can reduce an image with Transf-reduce Image and you can increase the size of an image by 2X with Transf-Zoom. The RGB-compositing can be slow for large images like these, therefore it is advisable to find the best color combination on a small subset of the images and then applying that to the larger image set. When saving a RGB-composite to a file please use *Save as - 24-bit Bitmap* or *Save as JPEG* as they preserve the correct colors. *Save as TIFF* may not produce the same colors as visible on the screen and *Save as GIF* may produce some distortion.

Geo-referencing of these images is possible by using the values in the *productdescription.self* file but is currently not very convenient as it requires the user to calculate the Longitude and Latitude coefficients from the coordinates of the center and corner points. A more convenient function for geo-referencing may be added to WIM in the future.

14 GOES-SST Products

The Sea Surface Temperatures have been derived from the series of NOAA's Geostationary Operational Environmental Satellites (GOES). Data is provided at a near real-time rate as 1 hour, 3 hour, and 24 hour gridded files at a spatial resolution of 6km (http://podaac.jpl.nasa.gov/noaa_goes/).

Three hour and 24 hour files are averages of the 1 hour derived SSTs. The Level 3 product includes data from both the GOES East (GOES-10) and GOES West (GOES-12) satellites. Data is processed at the National Environmental Satellite, Data, and Information Service (NESDIS) and is available from ftp://podaac.jpl.nasa.gov/sea_surface_temperature/goes/NOAA/.

These files are a special version of the simple raster (byte) image with the upper left corner at (180W,60N) and the lower right corner at (30W, 45S). These images are always 3000 columns by 2100 rows and in a simple Linear projection (Longitude coefficients are –180, 0.05, Latitude coefficients 60, -0.05). The scaling of the SST is *Linear* with *Slope* = 0.15, *Intercept* = -3.15. When read as GOES-SST WIM assigns the correct scaling and projection. These files can also be read as generic raster images (*.img) but then the user has to manually set the size, scaling and projection. Pixel values (0,2, 4) are used as flags for (space, land, cloud). The 24-hour files seem to be using only the flag with the pixel value 0.

File names are of the form sst1_YYYY-DDD-HH for the hourly results and sst3_YYYY_DDD_HH for the three-hourly results, e.g. a 1-hour result is sst1_2002_062_14, and a 3-hr result is sst3_2002_062_12. Please note that these files have no extension. In Windows the extension is used to associate files with the application, e.g. WIM. Therefore it is not possible to set up an association so that by just clicking on a GOES-SST file the data will be automatically read into WIM. Instead, the *File-Open-GOES-SST* has to be used.

.

15 Altimetry Products

Maps of Sea Level Anomaly (MSLA) in the netCDF format obtained from data of various sensors (TOPEX/POSEIDON, Jason, ERS-1/2, Envisat) can be read with WIM. The maps are provided on a MERCATOR 1/3° grid. Resolutions in kilometers in latitude and longitude are thus identical and vary with the cosine of latitude (e.g. from 37 km at the equator to18.5 km at 60°N/S). Units are centimeters. The files are in a NetCDF format and filenames are as follows: hh_msla_oer_tp (or merged) _h_date.nc (e.g. T/P map for January 23, 2002 = hh_msla_oer_tp_h_19015.nc). Mapping errors (in percentage of signal variance) are provided on separate files.
More details are provided in the SSALTO/DUACS (CLS/CNES near real time altimeter processing system) handbook.available at: http://www.jason.oceanobs.com/documents/donnees/duacs/handbook_duacs_uk.pdf.
The Jason-1 data is downloadable from ftp://ftp.cls.fr/pub/oceano/enact/msla/j1/

A list of various altimetry data is available at http://www-aviso.cls.fr/html/donnees/welcome_uk.html. For example, the 1 degree monthly average MSLA data from the merged TOPEX/POSEIDON, Jason, ERS-1/2 dataset is available from the following FTP side: ftp://ftp.cls.fr/pub/oceano/enact/msla/merged_monthly_average/ . The 1 degree data are compatible with the WIM Global Equal Angle projection.

The netCDF files should be read with WIM as regular HDF files. They have the *.nc extension. You can select WIM as the application to open *.nc files so that you can read them by clicking (double-clicking).. The original float32 data are converted to Byte data with Linear scaling and no appreciable loss to the quality of the data. As Sea Level Anomaly (SLA) values can be both negative and positive, the full scale is set to -50 to 77.5 cm. The time of the data file is specified in the attribute “Date_CNES_JD” that is the sequential Julian day starting with January 1, 1950. http://www-aviso.cls.fr/html/donnees/tools/jjtocd_uk.html provides a conversion utility. For example, CNES day 19750 corresponds to January 28, 2004. WIM converts the CNES day to attributes “Start Year” and “End Year”.

16 SST data from the Institute Maurice Lamontagne

Sea-surface temperature data for various regions of Canada is provided by the Department of Fisheries and Oceans, Quebec. The data is processed by the Institute Maurice Lamontagne and is available from ftp://calypso.qc.dfo-mpo.gc.ca/period/msst/hdf/

The data is in the format of SDS in HDF4 and as such is readable by WIM. However, some adjustments were made to WIM in order make the use of these datasets easier. Namely, the color scaling *Min* and *Max* are automatically set to -1.8 and 27.97, respectively, when reading those Float32 datasets (see View-Settings).

When using WAM utilities (e.g. *wam_match* and *wam_statist*) WAM will recognize the *period_start* and *period_end* attributes and convert those to the respective attributes that are needed to identify the time period of the dataset. The time period of a dataset is needed when doing match-ups with in situ data or when building time series of satellite data.

Geo-referencing information (projection) is automatically read from the HDF file attributes and operations like Geo-Get Map Overlay work without problems. However, in order to better visualize the data it is advised to convert the Float32 datasets to Byte with SST-PATHF scaling with Transf-Convert-Byte-SST-PATHF.

17 AMSR-E sea ice data from the University of Bremen

Sea ice concentration from the AMSR-E (Advanced Microwave Scanning Radiometer) sensor for both the Northern and Southern hemispheres as processed by the University of Bremen in Germany and is available from http://iup.physik.uni-bremen.de:8084/amsredata/asi_daygrid_swath/l1a/s6250/

The geo-referencing information has to be read from an external HDF file, e.g. *LongitudeLatitudeGrid-s6250-Antarctic.hdf* for the Southern hemisphere.

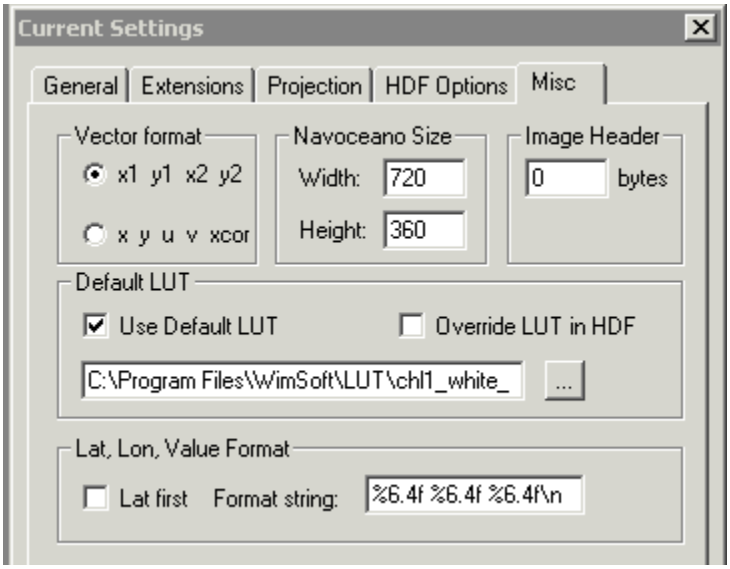
The data is in the format of SDS in HDF4 and as such is readable by WIM. However, some adjustments were made to WIM in order make the use of these datasets easier. Namely, the color scaling *Min* and *Max* are automatically set to 0 and 100, respectively, corresponding to ice concentration from 0 to 100% (see View-Settings).

In order to better visualize the data it is advised to convert the Float32 datasets to Byte with *Pixel Value* scaling with Transf-Convert-Byte-Pixel Value. The loss in accuracy (less than 1%) should probably be acceptable.

18 New Generation SST for Open Ocean

New Generation Sea Surface Temperature (NGSST) for Open Ocean (http://www.ocean.caos.tohoku.ac.jp/~merge/sstbinary/actvalbm.cgi?eng=1) is a merged SST product for the ocean around Japan that is being assembled from various sources by the Kawamura Lab in the Tohoku University, Japan. The main advantage compared to other traditional satellite products is that by merging data from different sources and modeling achieves high spatial resolution at high temporal frequency. The data format is a simple raster binary (*Image* in WIM terminology) with a header of 200 bytes. However, this format is not a modern format like HDF and therefore the reading program has to be specifically modified to read it. In order to use NGSST files with WIM do the following:

- Download the data from ftp://www.ocean.caos.tohoku.ac.jp/pub/mergedsst_binary/
- Uncompress the compressed files (e.g. with a command line program *gzip –d *.gz*)
- Set the ***Image Header*** to **200** bytes in *Settings - Misc* (see below)

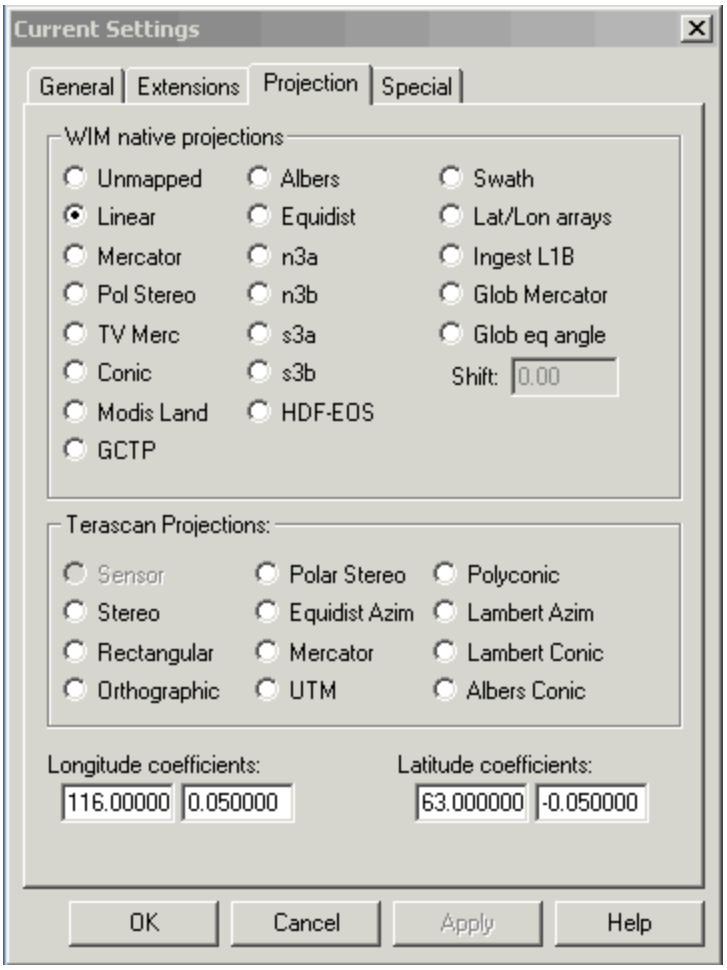


Load an image with *File-Open* with ***Files of type: Image (*.img)***. The default extension for the raster Image files is ****.img*** but these files have an extension ****.raw***; therefore you have to either specify ****.raw*** in the ***File name*** text box or add ****.raw*** to ***Settings-Extensions*** for ***Image file***. Separate the **.raw* from the other extensions with a semicolon, e.g. **.img;*.dat;*.nec;*.raw*.

After selecting the file you get to another screen of the *Current Settings* with an option to set ***Image Size***. Set ***Width = 1000, Height = 1000.*** Set the ***Value Scaling*** in the same screen to ***SST-PATHF, C***.

After loading the image you can see that it is upside down, To fix that use ***Transf – Mirror - Horiz. axis***. Now the image should look OK but it has no geo-location.

To set the geo-location open ***Settings-Projection*** and set the *WIM native projections* to ***Linear***, the ***Longitude coefficients*** to ***116*** and ***0.05*** and the ***Latitude coefficients*** to ***63*** and ***-0.05***. (see below).



Now you can verify that you have the correct geo-location by creating coastlines with *Geo - Get Map Overlay - coast_inter.b* with *Background Value* of 0 and *Foreground Value* of 255. Click back to the original image and overlay the coastlines with *Multi - Overlay Image*. They should match perfectly. You can now do *File – Save Info* and that will save these settings for this file. Next time when you load the same image you only have to do

Transf – Mirror – Horiz. axis as all the other settings will be read from the corresponding **.inf* file that you saved. The geo-location that was saved in the **.inf* file is automatically applied only to the corresponding file and not to other NGSST files with the same projection. However, you can transfer the geo-location info to all other images in WIM memory by selecting an image and then *Geo – Unify Geo-coeff*.
The information in the header (200 bytes) is currently not interpreted by WIM. However, a WIM Automation Module (WAM) program ***wam_convert_ngsst*** converts all NGSST files to HDF files including all attributes that are essential when performing time series analysis.

19 AMSR-E data from Remote Sensing Systems

Various data products from AMSR-E (Advanced Microwave Scanning Radiometer) are available from Remote Sensing Systems, Inc. (http://www.ssmi.com/amsr/amsr_data_description.html#amsre_data).

The gridded data files are available at 0.25 degree resolution in binary format. These files can be directly read with WIM but it is much more convenient to use a WAM program *wam_convert_amsre* to convert the binary files into HDF files and at the same time adding some important attributes. The HDF files can then be easily analysed with as times series using WIM and WAM programs. If you cannot use *wam_convert_amsre* to convert the binary files into HDF files, you have to go through multiple steps in order to read a single file.
When reading a binary file AMSR-E file directly with WIM you need to do the following.

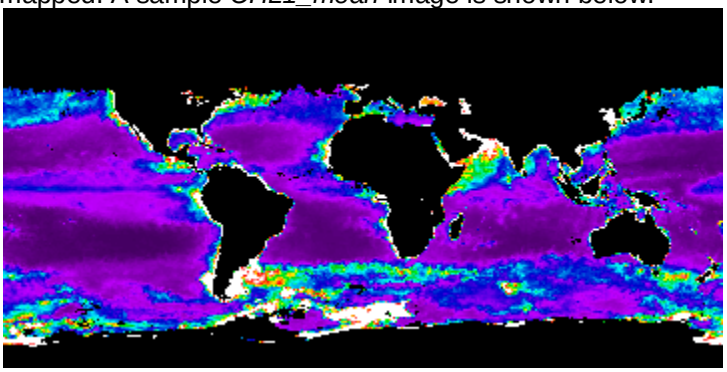
- Open the files with *File-Open* as (*Files of Type*) *Band Sequential.* As these AMSR-E data files do not have an extension, you need to specify * in the *File name* text box in order to see the available files.
- Then specify 5 (for 3-day, weekly and monthly files) or 6 (for daily files) as the Total number of bands. You can leave the Bands to read box open and read all the specified “Total number of bands”.
- Next specify 1440 as the *Width* and 720 as the *Height* of the images.
- Now the images should be in WIM but you can notice that they are upside down. To fix that do *Transf – Mirror - Horiz. axis*.
- To fix geo-location set *Settings - Projection* to *Glob eq angle* with a *Shift* of 180. You can confirm the correct geo-location with *Geo-Get Map Overlay – coast_inter.b, Background Value* = 0, *Foreground Value* = 1. Note to use 1 as the foreground value as choosing 255 would create white coastlines that may not be visible. Click back on one of the images and select *Multi - Overlay Image*, select the sequence number of the coastlines overlay.
- Now set the correct scaling with *Settings - Value scaling*. For SST you can choose *Value Scaling = SST-PATHF, C*. For all others select *Linear* with the appropriate *Slope* (e.g. 6 for TIME, 0.2 for WSPD, 0.3 for VAPOR, 0.01 for CLOUD, 0.1 for RAIN). The *Intercept* must be 0 for all (except SST that is -3 but it is fixed in the *SST-PATHF, C* scaling).

As you can see, reading AMSR-E data in the original format is rather cumbersome. Therefore it is better to use the WAM program *wam_convert_amsre* and convert all available AMSR-E binary files into standard HDF files that can be easily analyzed with WIM, WAM or other HDF capable software.

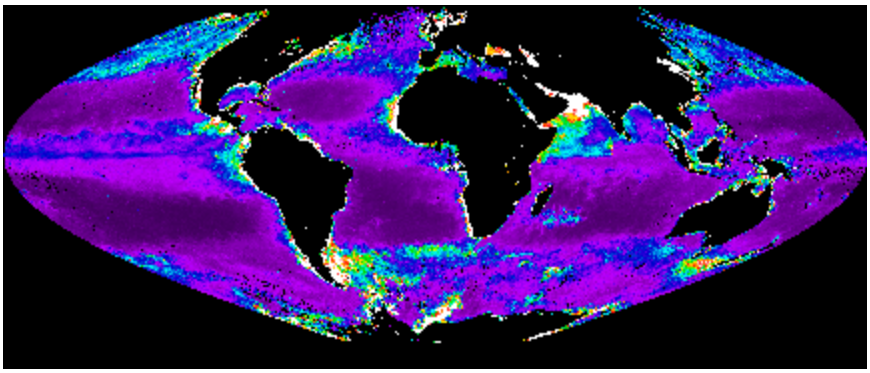
20 *GlobColour* ocean color products

GlobColour is an ESA project (http://www.globcolour.info) that produces global and local ocean color datasets from individual sensors or merged from multiple sensors. *GlobColour* global (FPS = Full Product Set) data can be downloaded from (http://www.globcolour.info/data_access_full_prod_set.html). Local datasets for many "diagnostic" data sites are available from http://www.globcolour.info/data_access_dds_list2.html.

Global ***mapped*** datasets in netCDF format can easily read into WIM as netCDF (old, version 2 and 3 netCDF files could be read also as HDF datasets). Either the *Grid* or *Global Equal Angle* projection (also called *Equirectangular* or Plate-Carré) should be automatically set. However, these mapped datasets are available at low resolutions, namely at 25 km (*L3m_*_GLOB_25_*.nc*) and 100 km (*L3m_*_GLOB_100_*.nc*). Note that the letter "m" after level designator "L3" means that the products have been mapped. A sample *CHL1_mean* image is shown below.



Global ***binned*** datasets are not mapped in any traditional projection but are binned on an *Integerized Sinusoidal* (ISIN) grid with a resolution of 1/24 degrees at the equator (~4.63 km). The number of columns is decreasing as the distance of a row of bins increases from the equator. The data is also distributed in netCDF format but as 1-dimensional arrays. WIM is transforming these arrays into a rectangular WIM image format using a specific projection (*GlobColour ISIN*). These global binned datasets at 4.63 km resolution have filenames like *L3b_*_GLOB_4_*.nc*). The files are rather big (~180 MB). A sample *CHL1_mean* dataset is shown below (reduced 20X for visualization here).



21 MERIS Level-3 products

MERIS is an advanced sensor on the ESA Envisat satellite. MERIS data are available from http://envisat.esa.int/level3/meris/. Global Level-3 datasets of normalized water leaving radiances at several wavelengths, Chlorophyll-a for Case 1 waters (Chl1) and others are available at Daily and Monthly compositing since 2002. Each dataset has corresponding files in JPG, XML, NX.GZ and HDF.GZ formats. Please note that the datasets that are useable by WIM are those in netCDF format (*.nc) and not those in HDF. After downloading, the compressed files (*.nc.gz) need to de uncompressed, e.g. with the command `gzip –d *.gz`. After that they are directly readable with WIM, e.g. by double-clicking on the filename in Windows Explorer. The XML files provide additional information about scaling, etc but are not needed in WIM. Each Level-3 file has the following datasets: *count*, *mean*, *stdev*, *min*, *max*. Most useful of those is ***mean***. The *stdev* is actually not the standard deviation but the sum of the squares of the input MERIS pixel values divided by the number of values. The Level-3 data are provided on global sinusoidal equal-area grid (Global ISIN) of approximately 9 km resolution (4320 x 2160) for the monthly images and of approximately 4.5 km resolution (8640 x 4320) for the daily images.

22 Cross-Calibrated Multi-Platform (CCMP) Ocean Surface Wind products

This dataset is a multi-satellite climate data record for ocean surface wind that spans nearly 21 years (Atlas et al. 2008, 2009). The product is derived through cross-calibration and assimilation of ocean surface wind data from SSM/I, TMI, AMSR-E, SeaWinds on QuikSCAT, and SeaWinds on ADEOS-II. Cross-calibration is performed by Remote Sensing Systems (RSS) under the DISCOVER project (http://www.remss.com/). The Level 3.5 data sets data are stored on a uniform grid with a resolution of 0.25°x0.25°, near global (78.375°S to 78.375°N) domain averaged over 5-day and monthly periods. The data are in netCDF format and can be downloaded from ftp://podaac.jpl.nasa.gov/ocean_wind/ccmp/L3.5a/data/.

After downloading you need to unzip the data:

```
gzip -d *.gz
```

WAM has a special converted program *wam_convertr_ccmp* that converts the netCDF files into HDF files that are much more convenient to work with WIM and WAM:

```
wam_convert_ccmp *.nc
```

Each file has 5 products: *uwnd*, *vwnd*, *wspd*, *upstr*, *vpstr*, *nobs* (see the user's guide at ftp://podaac.jpl.nasa.gov/ocean_wind/ccmp/L3.5a/doc/ccmp_users_guide.pdf). *wspd* is the scalar wind speed magnitude, *uwnd* is the eastward component, *vwnd* is the northward component, *upstr* is the eastward wind stress, *vpstr* is the northward wind stress and *nobs* is the number of observations.

If you don't have WAM (and *wam_convert_ccmp*) then you need to do manually the following set of operations (all done automatically with *wam_convert_ccmp*):
-. When loading the *.nc file, cancel the selection of *Geolocation file*
- *View - LUT Stretch*
- *Transf – Mirror - Horizontal axis*
- *Settings – Projection*, set *Linear* projection with *Longitude* coefficients 0, 0.25 and *Latitude* coefficients 78.375, -0.25.
- Check with *Geo – Get Map Overlay* that you have the correct projection.

With WAM you can do time series analysis of the 21 year wind records in your area of interest. Note that if you want to extract the *wspd* image with *wam_series*, you need to select SDS number 2 (the sequence is 0, 1, 2, 3, 4). Also note that *wam_statist* is always using the first (0th) SDS in the file. Therefore, if you want to use *wspd* in *wam_statist*, you need to extract it first with *wam_series* into a series of HDF files with a single SDS.

References

Atlas R., Ardizzone J., Hoffman R.N., 2008: Application of satellite surface wind data to ocean wind analysis, Proc. SPIE, Vol. 7087, 70870B (2008); DOI:10.1117/12.795371.

Atlas R., Hoffman R. N., Ardizzone J., Leidner S. M., Jusem J. C., 2009: Development of a new cross-calibrated, multi-platform (CCMP) ocean surface wind product. AMS 13th Conference on Integrated Observing and Assimilation Systems for Atmosphere, Oceans, and Land Surface (IOAS-AOLS)

23 OLCI products

OLCI is an advanced sensor on the ESA Sentinel-3A satellite launched in February 2016. Another OLCI sensor was launched on the Sentinel-3B satellite in 2018. Detailed information on OLCI is available at https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-3-olci/data-formats.

Below is the summary for WIM/WAM user.

OLCI data are distributed in the so-called SENTINEL-SAFE format. The main difference from similar products by NASA OBPG and even earlier MERIS sensor is that the main data unit is not a netCDF (or HDF) file with all the main datasets but a directory with many files. The data files are netCDF files but they contain only one variable plus its error (uncertainty) dataset. For example, a file *chl_oc4me.nc* contains the chlorophyll-a estimate and its error estimate. Note that the file name does not have any information about the time, etc. For example, a comparable MODISA Level-2 file has a name like A2016001200000.L2_LAC_OC.x.hdf. All the auxiliary information has been moved to the name of the directory. The directory names have become very long, e.g.

S3A_OL_2_WFR____20161001T181051_20161001T181351_20161001T205616_0179_009_198_2340_MAR_O_NR_001.SEN3

Notice, the following tags in the directory name: S3A = Sentinel-3; OL_2 = OLCI level-2; WFR = Water Full Resolution, i.e. water and atmosphere geophysical products. Instead of WFR we can have WRR = water reduced resolution products. Full resolution means about 300 m and reduced resolution about 1200 m resolution. Each such directory represents a data granule recorded during 3 minutes on the orbit and includes many files including the data manifest in an XML file. Note that in contrast with typical NASA OBPG level-2 files the latitude and longitude fields are not in the same file with the data but in a separate file called *geo_coordinates.nc*. Therefore, when manually reading chlorophyll data from *chl_oc4me.nc* you also need to pick *geo_coordinates.nc* for the latitude and longitude fields. Similarly, when reading any other data products, e.g. PAR or multiple reflectance bands, you need to specify the file with geo-coordinates. Currently there is no option select only some of the products and the whole directory needs to be downloaded. The size of a minimal download can be very large. Most users do not need all the variables and all the error fields and would rather extract only the relevant datasets and save disk space. WAM has tools for that. Currently there is a command line tool *wam_remap_lladir* and more will be added. With *wam_remap_lladir* extract a selected variable from a selected SAFE directory and save as HDF with geo-coding. For example, a command

wam_remap_lladir S3A_OL_2_WRR* chl_oc4me.nc

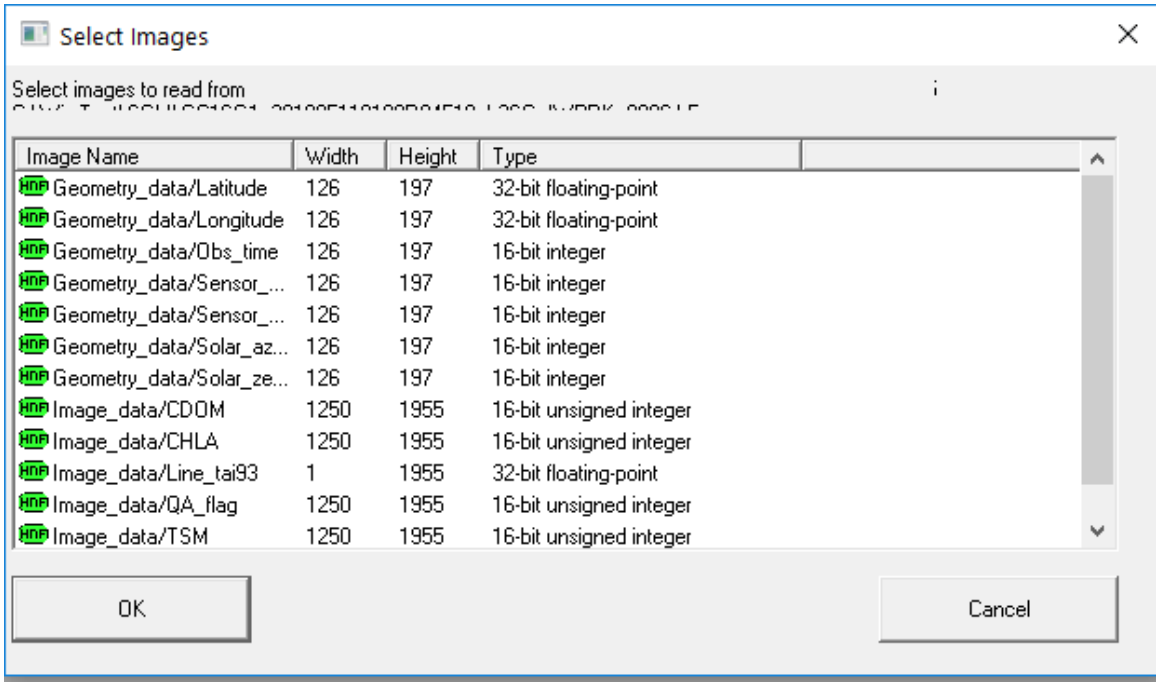
would process all directories matching S3A_OL_2_WRR*, read the file named chl_oc4me.nc and extract the CHL_OC4ME product (not the error field) and save in a HDF4 file by adding the Latitude and longitude fields (from *geo_coordinates.nc* in the same directory) and timing information. The file would have a name like *P2016275180004.L2_WRR_chl_oc4me*.x.hdf where P is for OLCI (as letter O has been taken for OCTS), followed by year (YYYY), day of the year (DDD), time (HHMMSS) in hours, minutes and seconds, followed by the product type (WFR or WRR) and the variable name (chl_oc3me). This naming convention follows NASA OBPG tradition. You can also remap the dataset to your standard map projection by adding the HDF map file name, e.g.

wam_remap_lladir S3A_OL_2_WRR* chl_oc4me.nc map=H:\Projections\map1.hdf

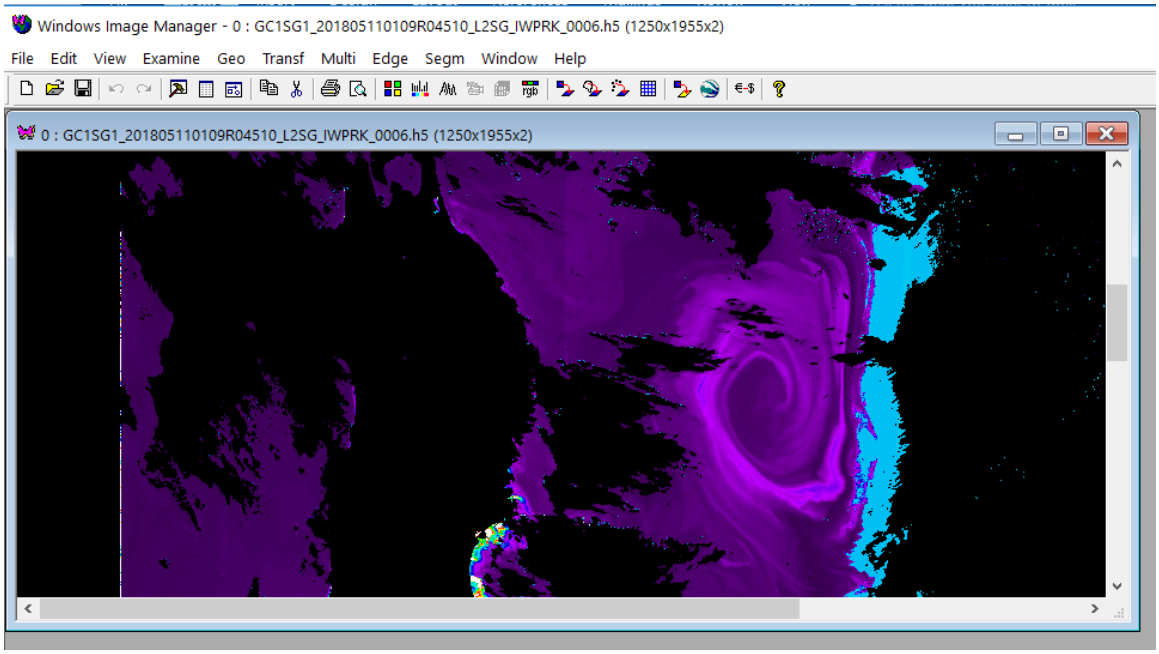
A command like **wam_remap_lladir S3A_OL_2_WRR* *reflectance.nc** would extract ALL reflectance bands and save in single file with Latitude and Longitude. Mote infprmation on WAM tools for OLCI are in the WAM manual.

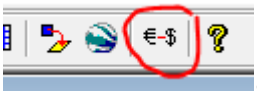
24 JAXA SGLI products

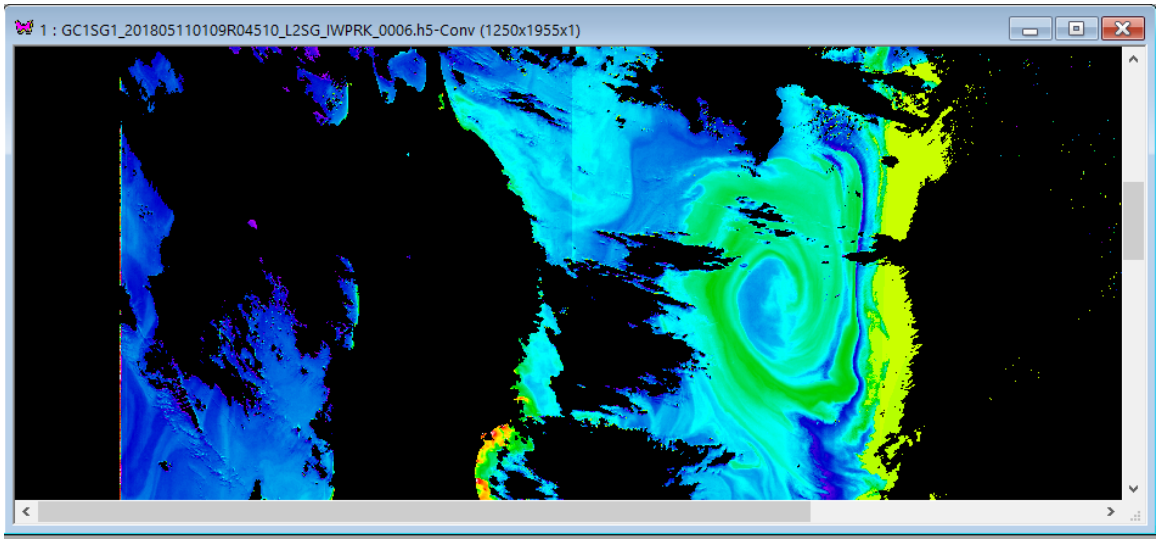
SGLI (second generation GLI) is the new color sensor on the JAXA GCOM-C satellite launched in December 2017 (http://global.jaxa.jp/projects/sat/gcom_c/).
Departing from the common format used for most satellite datasets, i.e. that of netCDF, SGLI data are distributed in the HDF5 (*.h5) format. Support for HDF5 and SGLI have been added starting from WIM version 11 and are being extended.
You can read the SGLI datasets with the regular ***File – Open – HDF5 file (*.h5)*** menu option. However, it is recommended to read them simply by clicking (double-clicking) on the .h5 file. To do that you need to associate WIM with the .h5 file name extension – right-click on any .h5 file name and select *Properties*. Then select ***Open with:*** and find WIM.exe in the *Program Files (x86)* directory. Then double-click (click) on the file name and select that you read the file in WIM with *Open – HDF5 file (*.h5)*. After doing that you can always just click (double-click) on a .h5 file and see the datasets that you can read.
As a demonstration, we will read a 1 km resolution file GC1SG1_201805110109R04510_L2SG_IWPRK_0006.h5. A corresponding 250-m resolution file is named GC1SG1_201805110109R04510_L2SG_IWPRQ_0006.h5, i.e. “Q” instead of “K”.
When I click (double-click) on the file, I see the following datasets:



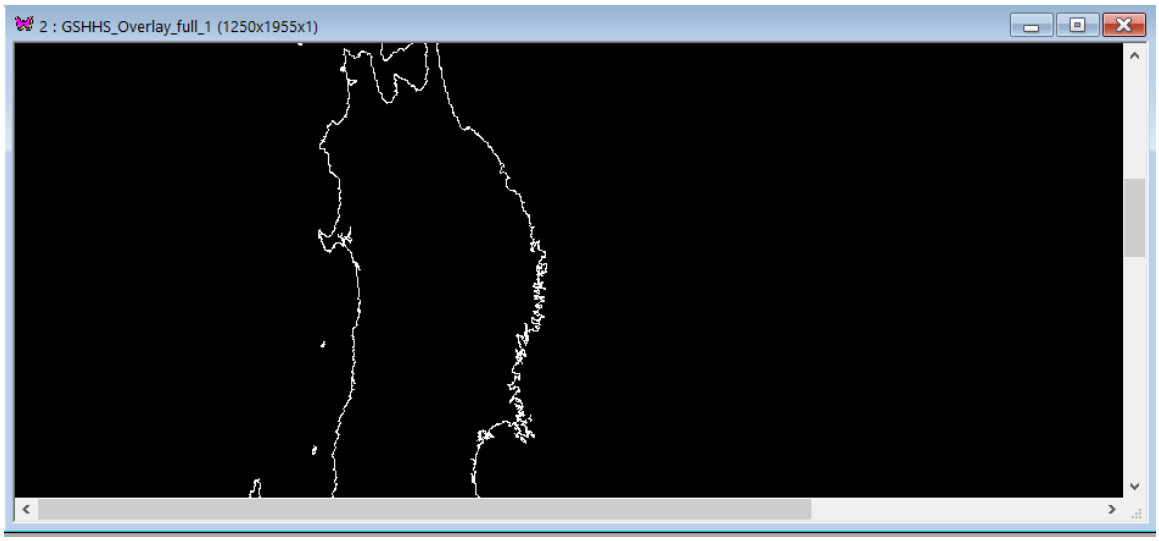
You can recognize *Latitude*, *Longitude*, *CDOM*, *CHLA* and others. Notice that these names are preceded by a “directory” name like *Geometry_data* or *Image_data*. Now select *CHLA* and we will see something like that:



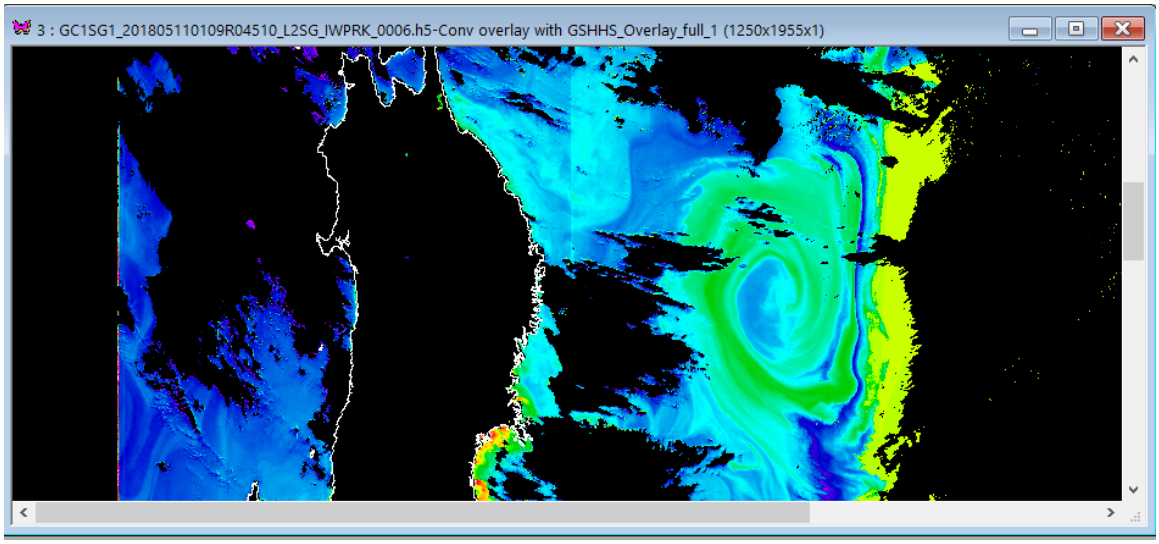
This is Chla concentration in linear scaling. You can see the pixel values in both scaled integers and as Chla concentration (in brackets) by right-clicking on the image but to better visualize the contrast we convert the image to Log-scaling using the Euro-$ tool  and *Log-Chl, mg/m3*. We will get something like that:



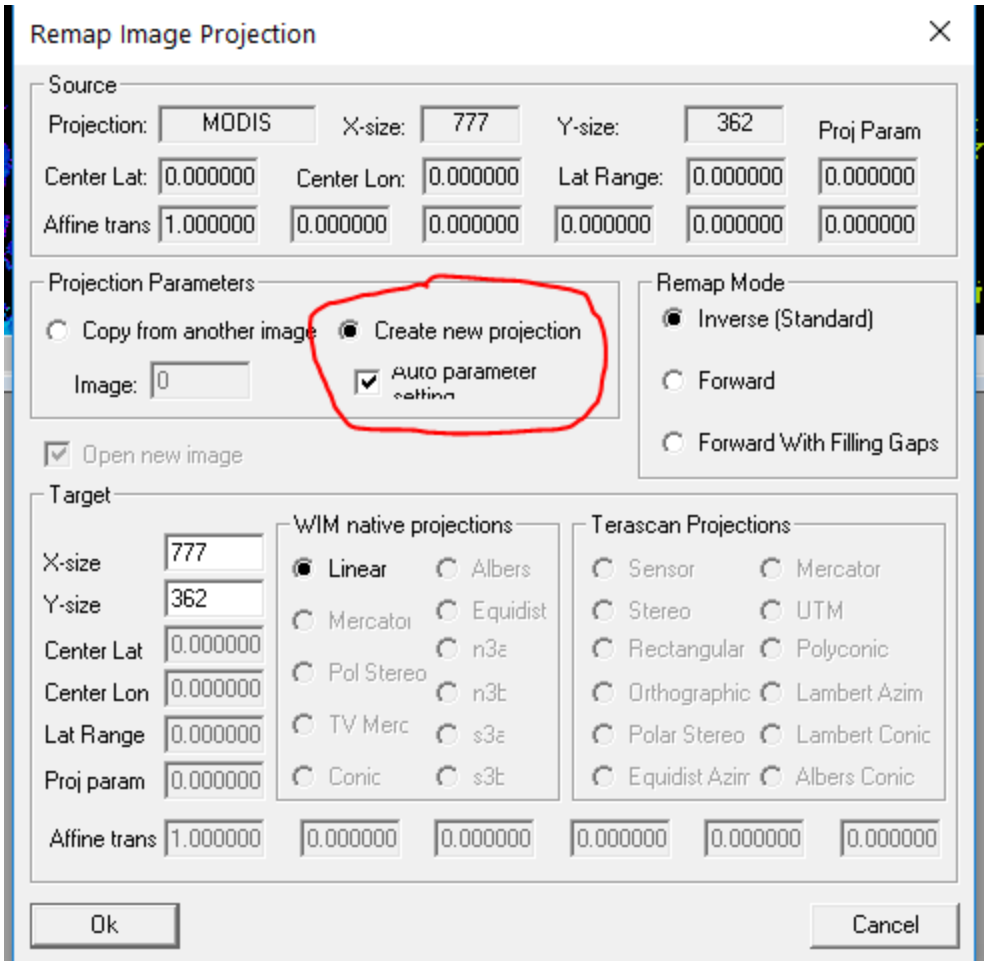
To check that we have the correct geo-location, we generate a coastlines image with *Geo-Get Map Overlay* with *coast_full.b*, *Background Value* of 0 and *Foreground Value* of 255:



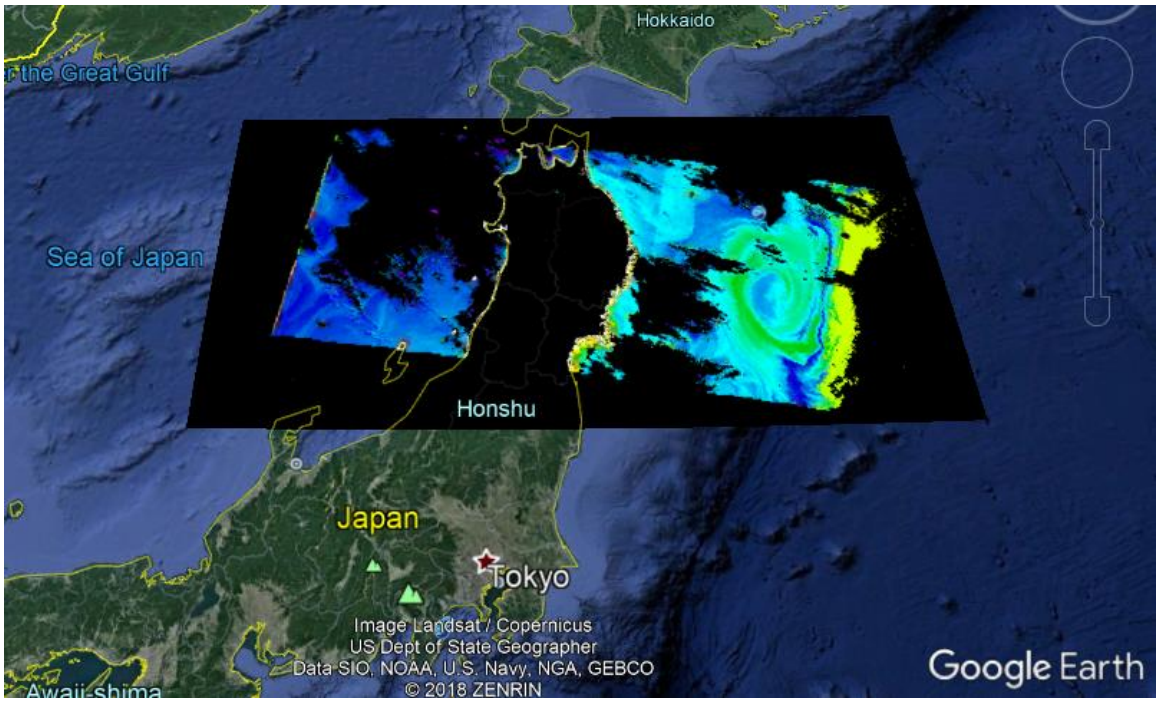
We now switch back (click) to the previous image and overlay the coastlines image with the Overlay tool . The coastlines match very well the image, i.e. geo-location is correct:



We can even view this image in Google Earth but for that the image has to be in Linear projection. To remap the image into a Linear projection we cut out the interesting part with the Scissors tool on Toolbar and then remap to itself using *Geo-Remap Projection* or icon on Toolbar and then using the option *Create new projection*:



After that we click on the Google Earth icon  on Toolbar and see the image in Google Earth:



As you can see, we have a nice eddy seen by the Chla field east of the Japanese island Honshu. The yellow area east of the eddy is obviously biased high by the high sensor zenith angle (edge of the swath) and should be masked using quality flags in *Image_data/QA_flags*. Support for using these flags will be added in the next update.

25 Useful Hints

Problem:
After reading a new image from a file, the image appears distorted on the screen.

Hint:
In order to display an image, WIM has to know its dimensions, most importantly the number of pixels in a line. The number of lines is not so important as the missing data is assumed to be zero and excessive data is simply ignored. WIM gets the image size automatically from its attributes (e.g. from HDF or other complex file formats) but plain raster formats do not have any attributes. You can check the current image dimensions from *List of Images* box or *View - Settings - General* - Image Size. If your image has different size, then before reading the image, specify the correct size in *View - Settings*. NB! If WIM finds an info file for the selected image file (the same name with the extension **.inf*), the image dimensions (*DX, DY*) are always read from the info file **disregarding the specified values**. It is convenient to make info-files for your images if their size is different from the default size of 512 x 512 (use *File* - Save Info). An info file contains at least two numbers: *DX* and *DY* (the number of pixels in a line and the number of lines), and optionally the four geo-conversion coefficients.

Problem:
You specify the correct image size before loading it (in *View - Settings - General* - Image Size) but after reading the image from a file, the size is changed to different values.

Hint:
You have an info file with the same image name and with the extension **.inf*. If WIM finds an info file for the selected image file, the image size (*DX, DY*) is taken from it. Delete the info file or correct it with an editor. When reading Erdas/Lan file, CoastWatch file or HDF file then the image dimensions are obtained directly from the file itself (header) thus overriding any default or specified values.

Problem:
While specifying a rectangular area on the image by mouse (e.g. for cutting part of the image, calculating statistics or histogram of an area) the rectangle borders and the numbers indicating its size are hardly visible on a fine-grained image with highly variable colors.

Hint:
By using *View - Set Colors* make the image look darker or lighter so that WIM can pick a color for drawing the borders and numbers that will be clearly visible.

Problem:
How to find the geo-conversion coefficients to convert the screen (video) coordinates to the geographical longitude and latitude in *Linear* projection ?

Hint:
The conversion coefficients are two pairs (*A, B, C, D*) of linear regression coefficients that convert the video coordinates (0; 0 in the upper left corner) to the longitude and latitude: *Lon = A + B * X; Lat = C + D * Y*. If you do have the coordinates (latitude and longitude) of the corner points then A equals the longitude of the upper-left corner and C equals the latitude of the upper-left corner. To find B (D), subtract the longitude (latitude) of the lower-right corner from the longitude (latitude) of the upper-left corner and divide the result by the corresponding image dimension minus 1.
If you don't know the coordinates of the corner points you have to find them the hard way. To find the regression coefficients, select around 10 points (ground control points) that are clearly distinguishable on the image and on your reference map, e.g. islands, headlands, lakes, and record their video coordinates to a file using *Examine - Save Points* option (select a file, point with the mouse exactly to each of the points and press the right button). Find the longitude and latitude of the points from the map (in decimal numbers), enter them as new columns of data into a worksheet where you have loaded the ASCII file with the video coordinates. Find the intercept and the slope of the linear regressions from the x-coordinate to longitude and y-coordinate to latitude. What you should get is something like that: 14.83356, 0.016362, 58.29435, -0.00891. You can now use an editor to create an info file that contains first the image dimensions (*DX* and *DY*) and then the coefficient values (each value separated with a space) in one line. The name of the file should correspond to the image file name with an extension **.inf* and should be in the same directory as the image file. Instead of using an editor, you can also enter the coefficients in *View - Settings* dialog boxes and use *File* - Save Info. Next time when you read the image, WIM will automatically read the coefficients from the info file and use them.
You may want to check out the info-files corresponding to some sample images (e.g., *calchl81.inf*).

Problem:
The image data is in the GIF, TIFF or some other format not directly readable by WIM.

Hint:
Use one of the many format converter programs. Many are available in the public domain or commercially. One of the most versatile image converters is Image Alchemy (Handmade Software, Inc.) that is available both for MS-DOS and UNIX. For example, to convert a GIF or TIFF 24-bit file use Alchemy with the BIF option that produces band interleaved by pixel (BIP) files, then use *File - Open - Pixel Interleaved....*

Problem:
How to save/print the image together with the corresponding color-bar (scale)?

Hint:
Use *View* – Annotate (since WIM 5.11).
In older versions of WIM you can use a more cumbersome procedure. You can use *Examine* - Scale to Clipboard and then use another program (e.g. MS

Paint) to place the color scale into a copy of the image at the bitmap level. To do that load the program Paint included with every copy of Windows 95. Copy the image bitmap from WIM to Paint via the Clipboard (*Edit – Copy* then within Paint *Edit - Paste*). Then copy the color-bar via Clipboard to Paint (*Examine* - Scale to Clipboard, within Paint *Edit - Paste*), move the color-bar to a suitable location, save the bitmap or print within Paint.
If your pixel values are coded (e.g. real values multiplied by 10, don't forget to select the *Value Scaling* option in *View - Settings - General*) in order to get the correct scale.
The transfer of color images works fine in 16/24 bit color and when the bitmap is saved as a 24-bit bitmap. With 8-bit color the colors may be distorted when Windows runs out of available colors.

26 List of Files

The following is a list of files that are normally installed with WIM. If you are short of disk space you may delete all the image files. Only the *.exe files and the *.dll files are essential to run WIM.

wim.exe	- WIM main program
wimLE.exe	- the limited evaluation version of WIM
hd414m.dll	- dynamic link library for HDF files
hm414m.dll	- dynamic link library for HDF files
wl1b.dll	- dynamic link library for AVHRR Level-1B files
gctp.dll	- dynamic link library for GCTP projections
tilemap.dll	- dynamic link library for MODIS Land projections
PointTypes.xml	- extensible list of symbols for point objects in XML format
Wim.hlp	- old help file for WIM, NOT updated with newer features
geotifcp.exe	- utility called by WIM when saving GeoTIFF tags to TIFF files
Wim.pdf	- WIM User's Manual in Adobe Acrobat format
WAM.pdf	- User's Manual for WIM Automation Module (WAM)

Look-up tables (LUT):

anomaly.lut, *anomaly5.lut,* *anomaly7.lut*	- sample LUT files for anomaly images
chl.lut, *chl1.lut,* *chl1_white_end.lut,* *chl2.lut, chl3.lut*	- sample LUT files for suitable for ocean color and other types of images
chl.raw, *spectrum.raw*	- a sample raw LUT files
Igbp_lgnd.lut, *mld.lut,* *petes24.lut,* *pigment.lut,* *spectrum.lut* *sst.lut,* *sst_kuring.lut*	- various sample LUT files, e.g. one for the mixed layer depth (mld), or one used by the IGBP Global Land Cover Characterization program (http://edcdaac.usgs.gov/glcc/images/gif/igbp_lgnd.jpg), for SST, etc.

Map datasets:

Global low-resolution datasets, the corresponding high-resolution (1 km and 100 m) datasets are installed from the WIM CD.

boundhigh.dat	country boundaries in high resolution
boundhigh.ndx	
boundlow.dat	country boundaries in low resolution
boundlow.ndx	
coast_crude.b	GSHHS shorelines in crude resolution
coast_full.b	GSHHS shorelines in highest resolution (~ 100 m)
coast_high.b	GSHHS shorelines in high resolution
coast_inter.b	GSHHS shorelines in intermediate resolution
coast_low.b	GSHHS shorelines in low resolution
coasthigh.dat	coastlines in high resolution, using GSHHS is faster!

coasthigh.ndx	
coastlow.dat	coastlines in low resolution, using GSHHS is faster!
coastlow.ndx	
GEB_050.DI6	50 m isobath sample for the US east coast
GEB_100.DI6	100 m isobath sample for the US east coast
riverhigh.dat	rivers in high resolution
riverhigh.ndx	
riverlow.dat	rivers in low resolution
riverlow.ndx	
world_bathy_5min.dat	world bathymetry for *Geo-Bathy Image* at 5 min resolution
world_bathy_2min.dat	world bathymetry for *Geo-Bathy Image* at 2 min resolution

Terascan Map Files:

Sample Terascan projection files of North America in:

albersco.hdf	Albers conic equal area projection
equidist.hdf	Equidistant azimuthal projection
lamberta.hdf	Lambert azimuthal projection
lambertc.hdf	Lambert conic projection
mercator.hdf	Mercator projection
orthogra.hdf	Orthographic projection
polarste.hdf	Polar stereographic projection
polyconi.hdf	Polyconic projection
rectangu.hdf	Rectangular projection
stereogr.hdf	Stereographic projection
utm.hdf	Universal Transverse Mercator projection

ESRI Shapefiles:

USstates.shp	ESRI shapefiles with US state boundaries
USstates.dbf	
USstates.shx	

27 References

Abbott M.R. and P.M. Zion (1985) *Cont. Shelf Res*., 4: 661-680.
Baldwin D.G. and Emery W.J. (1993) *Ann. Glaciol*. 17: 414-420.
Behrenfeld, M.J., Falkowski, P.G. (1997) Photosynthetic rates derived from satellite-based chlorophyll concentration. *Limnology and Oceanography*, 42, 1-20.
Borgefors G. (1983) Scandinavian Conference on Image Analysis, 3rd, pp. 250-255.
Cayula J.-F. and P. Cornillon (1992) *J. Atmos. Oceanic Techno*l., 9: 67-80.
Coll C., V. Caselles and J.A. Sobrino (1991) In: Ian Dowman (ed.), SPATIAL DATA 2000, Christ Church, Oxford, 149-157.
Diehl, S. F. J. W. Budd, D. Ullman, J.-F. Cayula (2002) *J. Atmos. Oceanic Technol*., 19: 1105-1113.
Emery W.J. et al. (1991) *J. Geophys. Res*., 96(C3): 4751-4768.
Emery W.J. (1995), personal communication.
An on-line archive of NOAA/AVHRR and other satellite data has been set up at http://jester.colorado.edu/EOSDIS.html. For details please contact Tim Kelley at kelley@jester.colorado.edu. The user can remotely run the **CCAR *navigate*** program and then download the produced images.
Holben B.N. (1986) *Int. J. Remote Sensing*, 7: 1417-1434.
Holyer R.J. and S.H. Peckinpaugh (1989), *IEEE Trans. Geosci. Remote Sens*., GE27(1): 46-56.
Lee J.-S. (1986) *Optical Engineering*, 25(5): 636-643.
Kahru, M., B.G. Mitchell (2001) *J. Geophys. Res*, 106(C2): 2517-2529.
Madsen S.N. (1986) Ph.D. Thesis, Danish Electromagnetics Institute, LD 62.
McClain E.P., W.G. Pichel and C.C. Walton (1985) J. Geophys. Res., C6: 11587-11601.
National Snow and Ice Data Center (1996) DMSP SSM/I brightness temperatures and sea ice concentration grids for the polar regions. User's Guide. 110 p.
O'Reilly et al. (1998) *J. Geophys. Res*., 103(C11): 24,937-24,953.
Pavlidis T. (1980) *Computer graphics and image processing*, 13: 142-157.
Peckinpaugh S.H. (1991) CVGIP: Graphical models and image processing, 53: 574-580.
Prangsma G.J. and J.N. Roozekrans (1989) *Int. J. Remote Sensing*, 10(4-5): 811-818.
Prentice G.S. (1987) *Int. J. Imag. Rem. Sens. IGS*, 1(1): 53-55.
Skriver H. (1989) Ph.D. Thesis, Danish Electromagnetics Institute, LD 74.
Snyder J.P. (1982) Map projections used by the U.S. Geological Survey. USGS Bulletin 1532, 313 p.
Stumpf R.P. (1992). First Thematic Conference on Remote Sensing for Marine and Coastal Environments, New Orleans, Louisiana, June 1992, (ERIM, Ann Arbor, Michigan): 293-305.
Sun Yan, A. Carlström and J. Askne (XXXX) *Int. J. Remote Sensing.*
Vesecky J.F., M.P. Smith and R. Samadani (1990) *IEEE Trans. Geosci. Remote Sensing* 28(4): 741-744.
Wessel, P., and W. H. F. Smith (1996) A global self-consistent, hierarchical, high-resolution shoreline database, *J. Geophys. Res*., 101, 8741-8743.