Standard
Operating
Procedure

Navigation: Post-Dive Processing

PROCESS OWNER
NOAA Ship *Okeanos Explorer*

REVISION HISTORY

REV	Description of Change	Author
00	Initial release	Brian Bingham

REFERENCE DOCUMENTS

Document Number	**Document Title**

NOAA SHIP OKEANOS EXPLORER R-337

SOP: Navigation: Post-Dive Processing

Quick Start

The details are to follow, but below is a minimal summary of the current post-dive processing steps. The paths and such are particular to the current configuration are particular to the current setup of the EXHypack PC.

1. Use PyLab tools to post-process dive track:
   1.1. Start PyLab
   There is a shortcut 'PyLab@ExRovDiveRare' on the EXHypack PC, Desktop>>DivePostProcess folder
   1.2. > `import scsRare`
   1.3. Next we are going to read the data and select a subset of the data to post-process. This subset is typically a single dive and is selected based on data and time. There are two ways to accomplish this.
      1.3.1. Interactive mode> `scsRare.importWarehouseData('X:\EX1103L2')`
         1.3.1.1. Follow the prompts to inter the inclusive date/times to describe the data of interest.
      1.3.2. Specify the times in the call
      > scsRare.importWarehouseData('/media/cruisedata/EX1302/',"2013.06.04 00:00:00","2013.06.04 23:59:59")
   1.4. The importWarehosueData function produces two plots (figure windows). Within the figure windows zoom to locate the extent of the "dive"
   1.5. > `scsRare.processDive()`
   1.6. > `scsRare.generateDiveTrack()`
   1.7. Create a new directory on the Hypack PC for the dive products. This directory should be in the C:\RovData\ directory under the appropriate cruise/leg. This directory is automatically synced with the warehouse folder on an hourly basis.
   1.8. >
   `scsRare.exportDive('EX1103L2_DIVE05','C:\RovData\EX1103L2\RovProducts\EX1103L2_DIVE05_20110718')`
2. Initiate the dive summary report
   2.1. Copy the latest dive report in ..
   \\192.168.4.200\rov\ROV\DiveReports\
   2.2. Open the text file summary (eg., C:\RovData\EX1103L2\RovProducts\EX1103L2_DIVE05_20110718\EX1103L2_DIVE05.txt)
   2.3. Copy the text summary and past it into "ROV Dive Summary" row in the report.
   2.4. Update the "Dive Site", "Date" and "Dive Number" fields within the dive summary report.
3. Export and process Hypack Targets
   3.1. From Hypack>>Administrator right-click on the target file, Export for Google Earth. Save the KML file within the new dive directory. For example…

C:\RovData\EX1103L2\RovProducts\EX1103L2_DIVE2_20110725\
EX1103L2_DIVE12_HypackTargets.kml

3.2. From PyLab

```
> run procHypackTargets.py
c:/RovData/EX1103L2/RovProducts/EX1103L2_DIVE12_20110725/EX1103L
2_DIVE12_HypackTargets.kml
```

Overview

The purpose of this document is to explain the process of generating dive summaries from the raw data archived on the "warehouse" (\\192.168.4.200\rov\ROV). Our goal in developing these tools is to generate some "rare" data products that will provide the following:

- A convenient data product that is a single file containing all the pertinent measurements and observations in chronological order: ROV dive-track, logged events, imagery filenames and video filenames.
- A brief summary report of a dive to help in completing the cruise report and dive reports.
- Visual representations (figures) that summarize individual dives.

Prerequisites

To go through this process you will need the following:

- An installation of the Enthought distribution of Python including iPython and PyLab http://www.enthought.com/repo/.hidden_epd_installers
- A connection to the "cruisedata" shared drive on the Okeanos network (EXWAN is sufficient)
- The Python scripts and modules included in the **ExRovDiveRare** project.
  The code is available via SVN at
  http://192.168.4.200/svn/trac/code/python/

Step-By-Step Process to Generate Dive Track and Dive Summary

1. Locate the Warehouse Data

All the data is located on the "warehouse" repository in a shared folder called "cruisedata". It is recommended that, rather than copying the data to a local location, you mount the repository so that you have access to the data. In Linux you might do something like...

```
sudo mount -t smbfs //192.168.4.200/cruisedata /media/cruisedata
```

On a Windows machine you would find the shared drive by typing something like this into the Windows Explorer "Address" bar...

\\192.168.4.200\

Then you should see all the shared drives. You can right-click on the "CruiseData" drive and map it to a local drive (e.g., X:\). The cruisedata shared drive is currently mapped to the X: drive on the Hypack PC.

2. Open Pylab

The current version of the processing tools is a set of interactive scripts that are called from within the interactive-Python (ipython) environment. (See notes below on getting and installing a proper ipython environment). You need to start the interactive shell. This can be done by opening the PyLab icon/shortcut or from the Linux command-line...

```
ipython -pylab
```

3. Prepare the Environment

The main processing scripts are contained in a file scsRare.py. In addition, there are some other files that have reusable functions and classes that are called by scsRare.py {abedict.py, navutils.py,ScsUtils.py,TimeBabel.py}. The easiest way to operate the scripts is to move to the directory that has all these files in one place.

For example, let's say that all these .py files are in a directory called '/home/bsb/Projects/NOAA_ROV/2011/ScsParse'. Now from within the ipython shell we can move to that director...

```
cd /home/bsb/Projects/NOAA_ROV/2011/ScsParse
```

Now we should be able to tell python that we want to use all the scripts in the scsRare.py module.

```
import scsRare
```

On the Hypack PC there is a shortcut to PyLab on the Desktop (user=rov) in a directory Dive PostProcess. This shortcut opens PyLab and sets the current directory to a location with the a working copy of the code (C:\home\rov\python\ExRovDiveRare)

4. Import the Warehouse Data

Hopefully the hard part is over! Now we will run a few scripts on the data to extract summaries. First we want to process all the data. In this example the data is from leg “EX1102” which is all in the cruisedata `shared` drive. To import all the data we just need to give the location of the pertinent leg directory.

```
scsRare.importWarehouseData('/media/cruisedata/EX1102')
```

or

```
scsRare.importWarehouseData('X:\EX1103L2')
```

Note: tab-complete will work in iPython for both the command/method name and the directory name.

The script will go through all the pertinent data and try to determine the temporal extent of the data. It will then prompt you for what subset of this data (in time) you would like to process.

You can do one of two things:

1. If you simply hit return it will take the default time given in brackets ([])
2. If you would like to use a different time to examine a subset of the available data, then enter a time using the format shown in the prompt.

Once you have specified the time-bounds you should see the script processing lots of data files and scanning directories....

Now the data is all loaded into Python as data structures, and we can proceed with processing.

Optional: If you would like to see how much data has been imported, you can issue a query which just prints to the screen.

```
scsRare.dataLen()
```

5. Plot the Data and Select a Dive [OPTIONAL]

[Note: This step is now automatically executed at the end of the importWarehouseData() function.]

Now we want to see what we've got. The following script makes two interactive figure windows to allow you to select (using zoom) what subset of the data you would like to transform into a "dive"

```
scsRare.plotRovData()
```

Which should generate two figures...

This figure shows depth an altitude.	This figure shows lat/lon. It doesn't make much sense yet since it is for the whole leg (many 'dives').

Now we can 'select' a dive (or any other subset of the data we'd like to export). You can do this by simply zooming in on a particular part of Figure 1. You will see that Figure 2 is updated each time you zoom to show the same time in lat-lon.

6. Process a Dive [OPTIONAL]

Once you have selected (using zoom) a single 'dive' in the record, the processing step will automatically find dive events. (Note: You don't have to select an actual dive, but I'm assuming that is the subset that will most often be of interest.) These dive events are...

- In the water (first time ROV at 50 m)
- On the bottom (first consistent altimetry between 5-10 m)
- Off the bottom (last consistent altimetry between 5-10 m)
- Out of the water (last time ROV at 50 m)

The process dive will also add some annotation to Figures 1 and 2

```
scsRare.processDive()
```

This should result in a adding some markers to the figures...

7. Generate a Dive Track

`scsRare.generateDiveTrack()`

This command filters (median filter and/or limits on lat/lon) the bottom track of Little Herc. and generates a track for being exported as KML and CSV files.

See `help(scsRare.generateDiveTrack)` for the following help and explanation:

```
generateDiveTrack(medfilt=(0.0001,15),latlim=None,lonlim=None)

This step in the processing does the following based on the data selected
in the scsRare.plotRovData() set
1) Filters the data using a median filter
2) Applies bounds to the latitude and longitude
3) Interpolates the depth and altitude values onto the track

Filtering:
Median filtering is done on both the lat and lon (independently)
INPUT
medfilt = (dx,len)  Median filter parameters as tuple
- dx = threshold for deviation from median in decimal degrees lat/lon
- len = 1/2 of the length of the median filter.
latlim = (latmin,latmax)  Limits on latitude as tuple
- latmin = minimum value of latitude allowable in the track in dec.
  degrees
- latmax = maximum value of latitude allowable in the track in dec.
degrees
lonlim = (lonmin,lonmax) Limits on longitude as tuple
- same as above ;)

OUTPUT
Adds a new key-value to the diveDataD dictionary with the key "trackD"

EXAMPLES
1) Use the default filter values
>> scsRare.generateDiveTrack()
```

```
2) Make a little tighter fitler.
>> scsRare.generateDiveTrack(medfilt=(0.0002,15))
```

8. Export a Dive (Images, CSV files, KML files, text report, etc.)

Now we want to export a dive summary that consists of three things (you don't have to export all of them):

- The "dive track" which is a filtered version of the ROV USBL locations (as both CSV and KML files)
- A "dive track" with data interpolated at 1 Hz for georeferencing imagery and video (as a CSV file and as a set of GGA records (talk to Webb about this data flow)).
- A comma-separated-value (CSV) file that gives a chronological record of the dive includes:
  - USBL locations (lat/lon)
  - Depth and altitude measurements
  - Events logged by science
  - Time and filename of all still imagery
  - Time and filename of all videos
- Images of the figures we've created (see below)
- A short text dive report that includes summary values (see below)

There is one script (exportDive()) that does this. (The script actually calls three separate scripts. You can call these individually if you would like.)

In our example, we have selected a dive we'll call Dive 5. You can generate the CSV summary, the dive report and the PNG figure images by issuing...

```
scsRare.exportDive("EX1102_Dive5")
scsRare.exportDive("EX1103L2_DIVE01","./data")
```

The script takes an optional argument of the directory you would like to store the files in. The default is the current directory (i.e., './'). We could save all the exported files in a directory called "EX1102rare" by adding an optional argument...

```
scsRare.exportDive("EX1102_Dive5",diveDir='./EX1102rare')
or
scsRare.exportDive('EX1103L2_DIVE08','C:\RovData\EX1103L2\RovProducts\
EX1103L1_DIVE08_20110721')
```

Note that the directory must already exist or you will get an error!

Note: the file names for products include the name you give the dive. Don't use spaces or unfriendly characters in the dive label!

1. `exportDiveTrack(diveName,diveDir='./')`
2. `exportDiveCsv2(diveName,diveDir='./')`
3. `exportDiveFigs(diveName,diveDir='./')`
4. `exportDiveReport(diveName,diveDir='./')`

If all goes well, you should have a few outputs from for the dive...

This includes the CSV file with all the data, the TXT file with the dive summary report, and two PNG files which are 300 dpi images of figure windows 1 and 2.

EX1102_Dive5_DepthAlt.png	EX1102_Dive5_LatLon.png

Export Hypack Targets as KML

Hypack 2011 allows for exporting some objects as KML, but (as usual with Hypack) there are some peculiarities involved with how it accomplishes this simple task. The KML version of the targets as exported by Hypack has not icon described (so GE defaults to the yellow pushpin) and the targets are clamped to ground (altitude of 0.0). The first attribute is annoying, but the second attribute can cause problems since the targets will be at the surface and as users scroll with GE things can behave in unexpected ways.

The following describes how to export targets from Hypack and then process these targets using a short Python script.

1. From Hypack>>Administrator right-click on the target file, Export for Google Earth. Save the KML file within the new dive directory. For example…
   C:\RovData\EX1103L2\RovProducts\EX1103L2_DIVE2_20110725\
   EX1103L2_DIVE12_HypackTargets.kml
2. Open PyLab and navigation to the "ExRovDiveRare" directory.
   You can use the same shortcut described above that should open PyLab with the current director set to (C:\home\rov\python\ExRovDiveRare).
3. Execute the script "procHypackTargets.py" and point it to the
   > `run procHypackTargets.py c:/RovData/EX1103L2/RovProducts/EX1103L2_DIVE12_20110725/EX1103L2_DIVE12_HypackTargets.kml`
   This should generate a new KML file in the dive directory with a suffice of _converted added to the file name. For example, the above command generated a file named…
   EX1103L2_DIVE12_HypakTargets._converted.kml

For an explanation of the processing script issue > `run procHypackTargets.py –h`

```
In [10]: run procHypackTargets.py -h

USAGE:

procHypackTargets.py inputFname [outputFname]

DESCRIPTION

This script process the .kml files exported by Hypack.

- Changes the icons used for the targets to placemarks with
highlightin

- Applies an altitude mode relative to the seafloor.  All points are
pu

ve the seafloor so that they are visable above any overlays clamped to

oor (see the <gx:altitudeMode> in KML for more details)

ARGUMENTS

inputFname - The .kml file written by Hypack
```

```
outputFname - [optional] The converted .kml file output.
              If this argument is not given, the default is to append
              "_converted" to the input file name (inputFname)
HISTORY
2011.07.23  bsb Created

EXAMPLES
>> ./procHypackTargets.py ~/Desktop/EX1103L2_DIVE09_HypackTargets.kml
```

Summary of Rare Data Products

The following table describes the products (files) that should be available as a result of the procedure described above.

Example Filename	Description
EX1103L2_DIVE12_20110725.txt	Dive summary statistics. This information is copied into the dive summary report.
EX1103L2_DIVE12_20110725_DiveTrack.png	Image of the filtered and raw USBL positions of the ROV during the dive.
EX1103L2_DIVE12_20110725_LatLon.png	Image of the raw USBL positions of the ROV during the dive.
EX1103L2_DIVE12_20110725_DepthAlt.png	Image of the raw depth and altitude measurements from the ROV during the dive.
EX1103L2_DIVE12_20110725_RovTrack.csv	Comma-separated-value file of filtered USBL positions of the ROV during the dive. Includes date, time , lat, lon, depth and altitude.
EX1103L2_DIVE12_20110725_RovTrack1Hz.csv	Equivalent to the CSV file above, but the track is interpolated (nearest neighbor) to each integer second during the dive. This is done for the convenience of the telepresence (Webb) and mapping (Mashkoor) teams.
EX1103L2_DIVE12_20110725_VIDEOS.csv	Comma-separated-value file of ROV positions (from dive track) at the start time of the indicated video segments. Determined by interpolating (linear) the ROV dive track.
EX1103L2_DIVE12_20110725_IMAGERY.csv	Same as above, reported for the times of the still image collection.
EX1103L2_DIVE12_20110725_EVENTS.csv	Same as above, reported for the times of the events logged.

`EX1103L2_DIVE12_HypackTargets.kml`	Post-dive targets as exported from Hypack.
`EX1103L2_DIVE12_HypackTargets._converted.kml`	Post-dive targets exported from Hypack and processed so they have an altitutde of 1.0 m above the seafloor and use a different marker with hightlighting.
`EX1103L2_DIVE12_20110725_Track.kml`	KML of ROV USBL track with times for viewing the track in Google Earth. This track is equivalent to the filtered track in the *_RovTrack.csv and *_DiveTrack.png files above.
`EX1103L2_DIVE12_20110725_Path.kml`	KML of raw ROV USBL track as a simple line (no times) for viewing in Google Earth. This track is raw and not filtered.
`EX1103L2_DIVE12_20110725_EndPoints.kml`	KML of the first and last raw ROV USBL positions for viewing in Google Earth.

Check the Synchronization with CruiseData

The dive data stored on the Hypack PC in c:/RovData is sync'ed with the "cruisedata" shared folder on the data warehouse (192.168.4.200). The data is organized by cruise/leg and the post-dive data products are stored in Products/ROV/

Copying the files from the Hypack machine to the data warehouse enforces some naming conventions. If those naming conventions are not followed, the data is not copied to the warehouse and consequently is not pushed to shore!

You should periodically check the status of the automated synchronization. You can do this using the WinMerge program installed on the Hypack PC. There is a project file for WinMerge in the "DivePostProcess" folder on the desktop. Double-clicking this WinMerge project folder will automatically open the project and examine the contents of the directories on the Hypack PC and on the data warehouse. You should see something like this…

You can see that there are some files that have not been copied to the data warehouse. By monitoring this you can check on the status of the synchronization and adjust accordingly.