

# Standard Operating Procedure #1

## Extracting annotation records and ROV data from SeaTube

Latest Revision: August 26, 2021 by J. Freeman

Purpose: This SOP describes how to download the ROV video annotation and vehicle data from SeaTube. These data are necessary to generate substrate maps for an ROV dive.

- 1) Open a web browser and navigate to <https://data.oceannetworks.ca/SeaTube>. *Note: SeaTube is an interactive online interface that allows users to view and annotate underwater video data. The SeaTube interface is used to record expert video annotations for NOAA OER ROV dives.*
- 2) When the SeaTube website is loaded, you will notice a list of various research cruises located on the left side of the screen under the tab “videos”.
- 3) From this list, navigate to the research cruise that you wish to download annotation data for.
- 4) If you expand the selected cruise, you will see individual dives listed. If you click on one of the dives, a video of that dive will be displayed as well as a map of the dive and a list of time stamped video annotations.
- 5) At the top of the screen there is a set of menus (Preview/Data Search/Plotting Utility/Sea Tube/More). Click on “More” then “Annotations” then “Annotation Search” to bring up a window that lets you select specific dives for which to export annotations and vehicle data.

Resource Type: Dive

Resource: EX1803-Dive01 2018-04-12 01:00:00.0 G

☒ Include annotations up topology tree

Date From (UTC): dd-MMM-yyyy hh:mm:ss

Date To (UTC): dd-MMM-yyyy hh:mm:ss

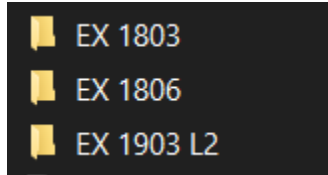
► Fields

► Owner

► Origin

Search Export... Save Search

- 6) To do this, under “Resource Type” select “Dive” and under “Resource” select the dive of interest that you want to export. The export button will prompt you to select Excel format or .csv. For this procedure, .csv should be selected.
- 7) Once the desired .csv files are downloaded, store them on a local directory in folders named by the research cruise/expedition the data were collected on:



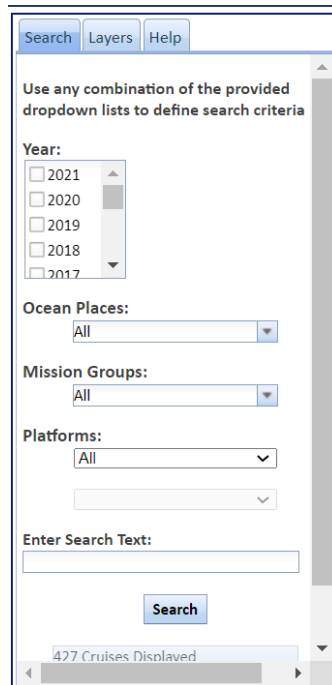
## Standard Operating Procedure #2

### Extracting 1Hz ROV data from the Ocean Exploration Digital Atlas

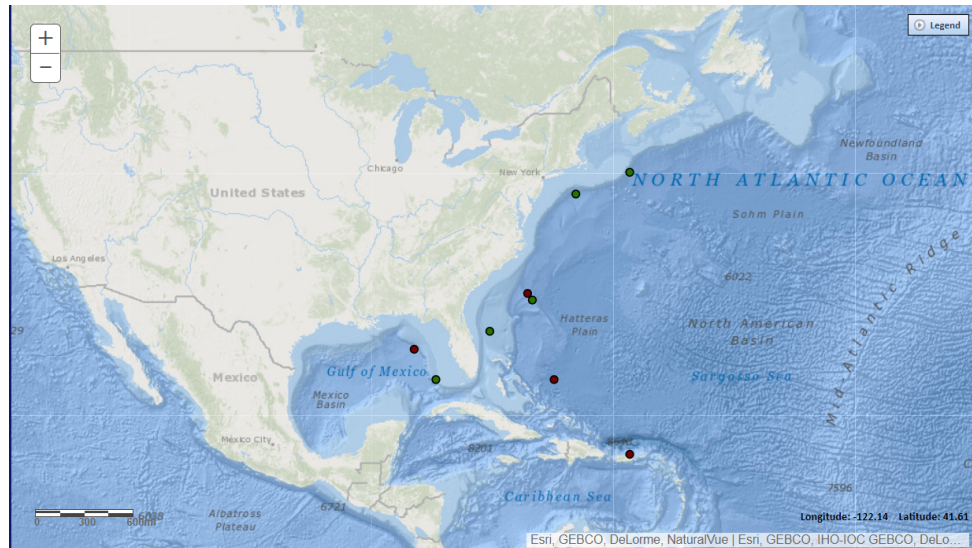
Latest Revision: August 26, 2021 by J. Freeman

Purpose: This SOP describes how to download 1Hz ROV vehicle data from NOAA's OER Digital Atlas. ROV navigation data recorded with frequency of 1 Hz are necessary to generate substrate maps for an ROV dive.

- 1) Open a web browser and navigate to <https://www.ncei.noaa.gov/maps/oer-digital-atlas/mapsOE.htm>. This Digital Atlas is an online map portal that allows the public access to data that is collected during NOAA OER expeditions.
- 2) When the digital atlas is loaded, you will see a search box located on the left side of the screen.

The image shows a web interface for searching NOAA Ocean Exploration data. At the top, there are three tabs: 'Search' (selected), 'Layers', and 'Help'. Below the tabs, a text prompt reads: 'Use any combination of the provided dropdown lists to define search criteria'. The search criteria section includes: 'Year:' with a list of years from 2017 to 2021 (each with an unchecked checkbox); 'Ocean Places:' with a dropdown menu set to 'All'; 'Mission Groups:' with a dropdown menu set to 'All'; and 'Platforms:' with a dropdown menu set to 'All'. Below these is a text input field labeled 'Enter Search Text:'. A blue 'Search' button is positioned below the text field. At the bottom of the search box, a status bar indicates '427 Cruises Displayed'.

- 3) Use this search box to search for the OER expedition of interests based on year, place, mission group, and/or platform.
- 4) When all parameters inside the search box are selected click "Search". The map frame will then display the queried OER expeditions.

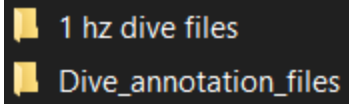


- 5) From the map, select the point for expedition of interests and click on it. This will display a window with the details of the expedition.
- 6) Click on the tab labeled “ROV Data Access.” Under “ROV Summary Products,” click on “ROV Data Access by Dive”.
- 7) When a “ROV Data Access by Dive” is selected a new window will be displayed, which will allow you to select any dive that was conducted during the expedition you selected in the map frame.
- 8) Select the dive that you want to download 1Hz data for and scroll to the bottom of the window, where a list of files is displayed:

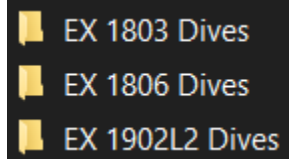
#### Download & View Files

Dive Summary Report (PDF - 921 KB)	<a href="#">View/Download</a>
Dive Track (KML - 67 KB)	<a href="#">View/Download</a>
ROV Ancillary Data (Zip - 3.74 MB)	<a href="#">Download</a>
ROV CTD/Sensor Data (Zip - 5.85 MB)	<a href="#">Download</a>
Camera Sled CTD/Sensor Data (Zip - 16.8 MB)	<a href="#">Download</a>
Low-Resolution Video Clips (Zip - 10.9 GB)	<a href="#">Download</a>
Underwater Still Images (Zip - 80.3 MB)	<a href="#">Download</a>
Dive Video Collection Self-Service Portal	<a href="#">Open</a>

- 9) Select the “Download” link for “ROV Ancillary Data”. These data will be downloaded to your local directory as a zipped folder. Once the data are unzipped, they should be stored within the same local directory as the SeaTube annotation data (see SOP # 1):



10) Once the desired .csv files are downloaded, store them on a local directory in folders named by the research cruise/expedition the data were collected on:



# Standard Operating Procedure #3

## Conversion of legacy substrate annotations to fine resolution CMECS units

Latest Revision: August 26, 2021 by J. Freeman

Purpose: This SOP describes the python script *CMECS classification script.py* that converts seafloor substrate annotations recorded in a legacy format into a new format compliant with the current CMECS standard.

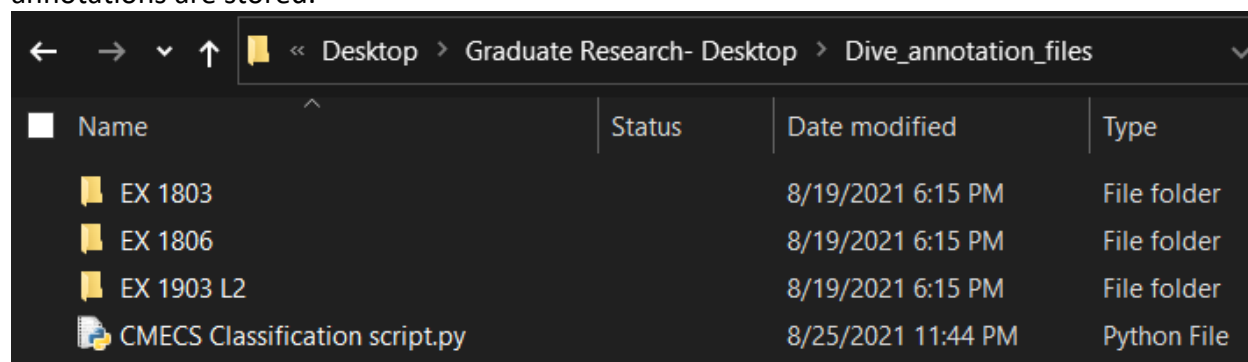
### Note:

For some NOAA expeditions (e.g. EX 1803 and EX1806), substrate observations were recorded using a classification system of four categories that refer to the primary (>50%) and secondary substrates: Hard/Hard, Hard/Soft, Soft/Hard and Soft/Soft (Bassett et al., 2017). The described Python Script converts those annotations into CMECS compliant substrate units following the scheme below:

- |  |                            |
|--|----------------------------|
| ● Fine Unconsolidated Mineral Substrate            | > Fine: Fine               |
| ● Fine and Coarse Unconsolidated Mineral Substrate | > Fine:Coarse, Coarse:Fine |
| ● Coarse Unconsolidated Mineral Substrate*         | > Coarse:Coarse            |
| ● Rock & Fine Unconsolidated Mineral Substrate     | > Fine:Rock, Rock:Fine     |
| ● Rock & Coarse Unconsolidated Mineral Substrate   | > Rock:Coarse, Coarse:Rock |
| ● Rock Substrate                                   | > Rock:Rock                |
| ● Unobserved or Unknown Substrate                  | > Unknown                  |

\* consider modifiers to 'Coarse' to represent *shell hash* and *coral rubble*

The script *CMECS classification script.py* should be stored in the same directory where the dive annotations are stored.



- 1) In the script, the variable "file" is a path to the .csv file containing downloaded SeaTube annotations (See SOP #1) that need to be converted to CMECS compliant substrate units. See script comments for example path.

- 2) The variable “outfile” is the path to a new .csv file that will contain the converted CMECS substate unit annotations. See script comments for example path.

```
# This is where the input and output annotation files are entered. The outfile must be set in a new file folder within the current working folder.
file = ('C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate Research- Desktop\\Dive annotation files\\EX 1806\\Dive14old.csv')
outfile = ('C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate Research- Desktop\\Dive annotation files\\EX 1806\\new\\Dive#14_new.csv')
```

- 3) The “outfile” must be located within the same working folder as the “file” that will be converted. For example, the image below is the EX1803 dive annotation folder. The “new” folder that will contain the reclassified .csv annotation files, must be located inside the same working folder as the source annotation files, or in this case inside the EX1803 annotation folder.

← → ↑ « Graduate Research- Desktop > Dive\_annotation\_files > EX 1803 🔍 Search EX 1803

Name	Status	Date modified	Type	Size
new				
Dive#1.csv		8/10/2021 4:32 PM	File folder	
Dive#3.csv		8/3/2021 3:05 PM	Microsoft Excel Co...	52 KB
Dive#4.csv		8/3/2021 3:06 PM	Microsoft Excel Co...	35 KB
Dive#5.csv		8/3/2021 3:06 PM	Microsoft Excel Co...	24 KB
Dive#7.csv		8/8/2021 4:42 PM	Microsoft Excel Co...	49 KB
Dive#8.csv		8/3/2021 3:09 PM	Microsoft Excel Co...	48 KB
Dive#9.csv		8/3/2021 3:09 PM	Microsoft Excel Co...	21 KB
Dive#10.csv		8/3/2021 3:13 PM	Microsoft Excel Co...	45 KB
Dive#11.csv		8/3/2021 3:14 PM	Microsoft Excel Co...	37 KB
Dive#12.csv		8/3/2021 3:14 PM	Microsoft Excel Co...	41 KB
Dive#14.csv		8/3/2021 3:18 PM	Microsoft Excel Co...	20 KB
Dive#15.csv		8/8/2021 4:44 PM	Microsoft Excel Co...	39 KB

- 4) Example of ROV annotation .csv substrate field prior to conversion.

[illegible]

5) Example of ROV annotation .csv substrate field after application of the *CMECS classification script.py*:

[illegible]



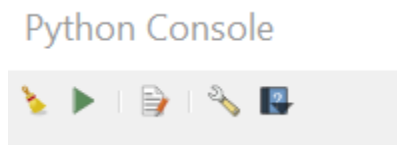
# Standard Operating Procedure #4

## Generation of dive path and viewshed shapefiles in QGIS

Latest Revision: August 26, 2021 by J. Freeman

Purpose: This SOP describes the process of using the automated python script *Mapping Script.py* to generate digital shapefiles of the ROV dive path (line feature) and imaged as well as classified seafloor viewsheds (polygon features).

- 1) Download and install QGIS (<https://qgis.org/en/site/forusers/download.html>). QGIS is open-source desktop geographic information system software that works on most common computing platforms/ operating systems.
- 2) Open QGIS.
- 3) Create a new project and name it "ROV dive mapping program".
- 4) At the top of the screen, click on "Plugins"
- 5) Click on "Python Console".
- 6) Within the python console in the top left corner, you will notice five different icons:



- 7) Click on notepad icon. This is the third icon from the left.
- 8) Once the python editor is open, select the yellow file icon in the top left corner of the python editor. This will allow you to open *Mapping Script.py*, which is saved in your working folder.



- 9) Each module of the script and its purpose is described below.

10) Create points layer from table tool (navigation):

**Code:**

```
result = processing.run("qgis:createpointslayerfromtable",  
  
{'INPUT':"C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate  
Research- Desktop\\1 hz dive files\\EX 1803  
Dives\\EX1803_DIVE03_20180416\\DIVE031Hz.csv",  
'XFIELD': 'LON_DD',  
'YFIELD': 'LAT_DD',  
'ZFIELD': None,  
'TARGET_CRS': QgsCoordinateReferenceSystem('EPSG:4326'),  
'OUTPUT':"C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate  
Research- Desktop\\NOAA  
Shapefiles\\path_dive_EX1803#03.shp"})  
onehzpath = result['OUTPUT']
```

11) The first part of the mapping script uses the QGIS “Create points layer from table” tool. This tool takes navigational data stored in the previously downloaded 1Hz file (See SOP #2) and creates a set of points for the route that was taken by the ROV during the dive. This point shapefile will not be shown in the final dive map but are necessary for the creation of the “Dive Path” shapefile.

- “INPUT” is a path to the previously downloaded 1Hz data file from the ROV in .csv format (See SOP #2). In the example code above, the path is set to the location of the 1 Hz .csv file for dive 03 of expedition EX 1803.
- “XFIELD” by default is 'LON\_DD'
- “YFIELD” by default is 'LAT\_DD'
- “ZFIELD” is "None".
- “TARGET\_CRS” is "QgsCoordinateReferenceSystem('EPSG:4326')".
- “OUTPUT” is a path for the desired folder in which the output shapefile will be stored.

12) Points to path tool:

**Code:**

```
result = processing.run("qgis:pointstopath",  
{  
'INPUT': onehzpath,  
'ORDER_FIELD': 'DATE',  
'GROUP_FIELD': None,  
'DATE_FORMAT': '',  
'OUTPUT':"C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate  
Research- Desktop\\NOAA Shapefiles\\Dive Path.shp"})  
divepath = result['OUTPUT']
```

- 13) This portion of the mapping script uses the QGIS “Points to path” tool. This tool creates a dive path shapefile from 1Hz dives points that were generated from the previous “create points layer from table” portion of the mapping script.
- “INPUT” is “onehzpath”, which is the output of the previous portion of the mapping script.
  - “ORDER\_FIELD” is “DATE”.
  - “GROUP\_FIELD” is “None”.
  - “DATE\_FORMAT” is (' ').
  - “OUTPUT” is a path for the desired folder in which the output shapefile will be stored. For the dive path to be generated correctly the name of the shapefile should have the format “Dive Path.shp”.

14) Create points layer from table tool (annotation):

**Code:**

```
result = processing.run("qgis:createpointslayerfromtable",
{'INPUT':"C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate
Research- Desktop\\Dive_annotation_files\\EX
1803\\new\\Dive#3_new.csv",
'XFIELD': 'Environment - Longitude',
'YFIELD': 'Environment - Latitude',
'ZFIELD': None,
'TARGET_CRS': QgsCoordinateReferenceSystem( 'EPSG:4326' ),
'OUTPUT':"C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate
Research- Desktop\\Graduate Research\\NOAA
Shapefiles\\annotation_points_EX1803_Dive#3.shp"})
annotation = result['OUTPUT']
```

- 15) This portion of the mapping script uses the QGIS “Create points layer from table” tool. This tool takes annotation data stored in the previously downloaded SeaTube annotation data file in .csv format (See SOP #1 and SOP #3) and creates a set of points at each location where seafloor annotation was recorded.
- “INPUT” is a path to the previously downloaded dive annotation data file from SeaTube in .csv format (See SOP #1 and SOP #3). In the example code above, the path is set to the location of the annotation .csv file for dive 03 of expedition EX 1803.
  - “XFIELD” by default is 'Environment - Longitude'
  - “YFIELD” by default is 'Environment - Latitude'
  - “ZFIELD” is "None".
  - “TARGET\_CRS” is "QgsCoordinateReferenceSystem('EPSG:4326')".
  - “OUTPUT” is a path for the desired folder in which the output shapefile will be stored.

## 16) Wedge buffer tool

### Code:

```
result = processing.run("native:wedgebuffers",
{'INPUT': annotation,
'AZIMUTH':
QgsProperty.fromExpression("attribute('Environm_1')"),
'WIDTH': 44,
'OUTER_RADIUS': 0.0005,
'INNER_RADIUS': 0,
'OUTPUT':"C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate
Research- Desktop\\Graduate Research\\NOAA
Shapefiles\\wedge_buffers_EX1903_dive#3.shp"})
wedgebuffer = result['OUTPUT']
```

17) This portion of the mapping script uses the QGIS “Wedge Buffer” tool. This tool creates wedge polygons that represent the seafloor extent of the ROV camera viewshed at the point in time when each annotation was made. The created wedge polygons have an assigned substrate attribute based on the dive annotation file input in the previous portion of the mapping script.

- “INPUT” is “annotation”, which is the output of the previous portion of the mapping script.
- “AZIMUTH” is “QgsProperty.fromExpression(“attribute(‘Environm\_1’)”)”.
- “WIDTH” is set to “44”. To represent an estimated camera view angle of 44°(This may be modified to represent different camera view geometries)
- “OUTER\_RADIUS” is set to 0.0005. This represents an estimated viewshed depth in map units. (This may be modified to represent different camera view geometries and environmental conditions)
- “INNER\_RADIUS” is “0”.
- “OUTPUT” is a path for the desired folder in which the output shapefile will be stored.

## 18) Dissolve tool:

### Code:

```
result = processing.run('qgis:dissolve',
{'INPUT': wedgebuffer ,
'DISSOLVE_ALL': None,
'FIELD': 'Substrate',
'OUTPUT':"C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate
Research- Desktop\\NOAA Shapefiles\\CMECS Substrate
Units.shp"})
dissolvebuffer = result ['OUTPUT']
```

19) This portion of the mapping script uses the QGIS “Dissolve” tool. This tool dissolves the boundary between overlapping wedge polygons with common substrate classifications.

- “INPUT” is “wedgebuffer”, which is the output of the previous portion of the mapping script.
- “DISSOLVE\_ALL” is “None”.
- “FIELD” is 'Substrate'
- “OUTPUT” is a path for the desired folder in which the output shapefile will be stored. The output name must remain “CMECS Substrate Units. Shp” for the map to be properly displayed.

20) Load substrate shapefiles in QGIS:

**Code:**

```
qmlfile = "C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate
Research- Desktop\\NOAA qml files\\Substrate type by
colorv2.qml"
fileInfo = QFileInfo(dissolvebuffer)
baseName = fileInfo.baseName()
rlayer = QgsVectorLayer(dissolvebuffer, baseName)
if rlayer.isValid():
    print ('Layer loaded!')
QgsProject.instance().addMapLayer(rlayer)
processing.run("native:setlayerstyle", {'INPUT':
dissolvebuffer, 'STYLE': qmlfile})    ##returns
nothing.....
extent = rlayer.extent()
rlayer.triggerRepaint()
```

21) This portion of the mapping script loads the previously generated and dissolved substrate wedge shapefiles into QGIS and applies a uniform prescribed color symbology.

- “qmlfile” sets the exact color and symbology for the polygon shapefiles. The substrate qml file should be stored in the same working folder as all other shapefiles and scripts used to generate the maps.
- “fileInfo” is “QFileInfo(dissolvebuffer)”.
- “basename” is “fileInfo.baseName()”.
- “rlayer” is “QgsVectorLayer(dissolvebuffer, baseName)”.
- The if statement and the parameters found within it should not be modified.

22) Load dive path shapefile in QGIS:

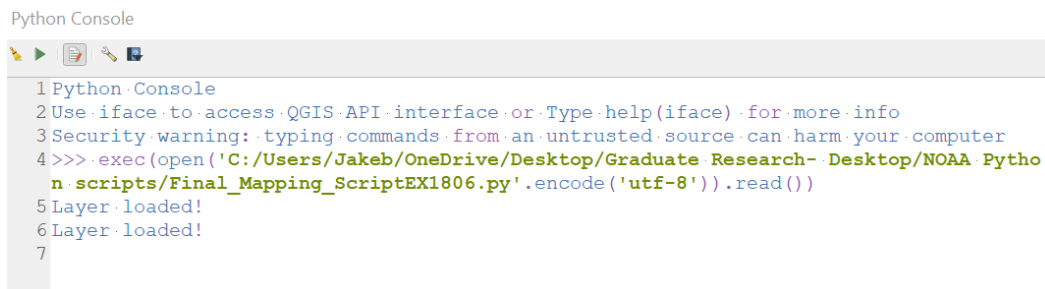
**Code:**

```
qmlfile= "C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate  
Research- Desktop\\NOAA qml files\\divepath.qml"  
fileInfo = QFileInfo(divepath)  
basename = fileInfo.baseName()  
rlayer = QgsVectorLayer(divepath, basename)  
if rlayer.isValid():  
    print('Layer loaded!')  
QgsProject.instance().addMapLayer(rlayer)  
processing.run("native:setlayerstyle", {'INPUT': divepath,  
'STYLE': qmlfile})  
extent = rlayer.extent()  
rlayer.triggerRepaint()
```

23) This portion of the mapping script loads the previously generated and dive path shapefile into QGIS and applies a uniform prescribed color symbology.

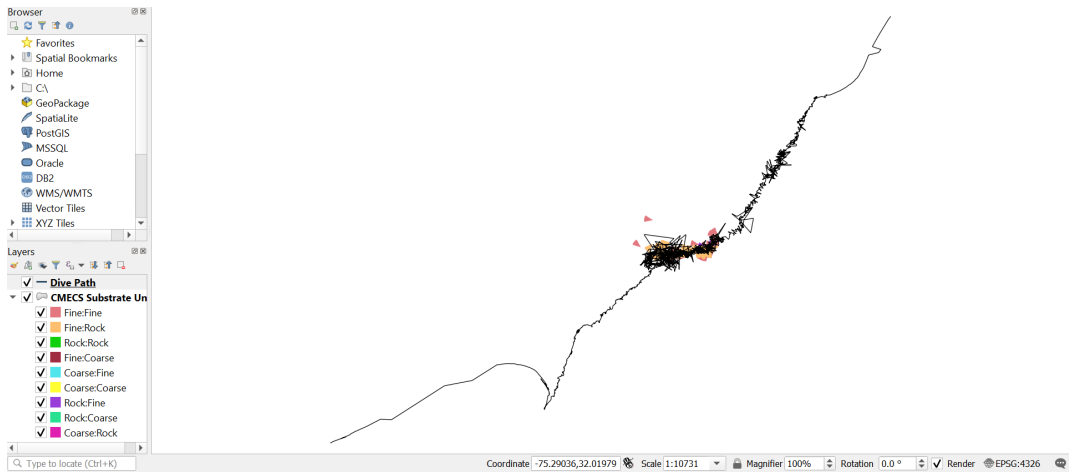
- “qmlfile” sets the exact color and symbology for the line shapefile. The substrate qml file should be stored in the same working folder as all other shapefiles and scripts used to generate the maps.
- fileInfo” is “QFileInfo(divepath)”.
- “basename” is “fileInfo.baseName()”.
- “rlayer” is “QgsVectorLayer(divepath, baseName)”.
- The if statement and the parameters found within it should not be modified.

24) If the script is running correctly, you will notice two “Layer Loaded” icons on the left panel in the python editor. This indicates that both the line shapefile, and substrate shapefiles have successfully loaded in the QGIS canvas.



```
Python Console  
1 Python Console  
2 Use iface to access QGIS API interface or Type help(iface) for more info  
3 Security warning: typing commands from an untrusted source can harm your computer  
4 >>> .exec(open('C:/Users/Jakeb/OneDrive/Desktop/Graduate Research- Desktop/NOAA Python  
  scripts/Final_Mapping_ScriptEX1806.py'.encode('utf-8')).read())  
5 Layer loaded!  
6 Layer loaded!  
7
```

25) The QGIS map will show both the “Dive Path” and “CMECS Substrate Units” shapefiles.



# Standard Operating Procedure #5

## Creation of .pdf output dive map

Latest Revision: August 26, 2021 by J. Freeman

Purpose: This SOP describes the process of creating a .pdf format output map with *Map production script.py*

- 1) After successfully running the *Mapping script.py* python script as described in SOP #4, open *Map production script.py* in the python plugin in QGIS.
- 2) Unlike *Mapping script.py*, *Map production script.py* does not require any input directories to be set.
- 3) Setting the layers to be mapped:
  - This portion of the script is where the map layers that will be displayed in the QGIS map layout and final .pdf map are called.
  - Both the “Dive Path” and “CMECS Substrate Units” should be called.

### **Code:**

```
layers = QgsProject.instance().mapLayersByName('Dive Path')
layer = layers[0]
layers2 = QgsProject.instance().mapLayersByName('CMECS
Substrate Units')
layer2 = layers2[0]
project = QgsProject.instance()
manager = project.layoutManager()
```

- 4) Creation of map layout:
  - The variable “layoutName” assign a name to the map layout that will be created in the current map canvas. For this, a simple name such as “Dive Map Layout” will work. This parameter does not have to remain “Dive Map Layout” it can be renamed each time a new ROV dive map is produced.

### **Code :**

```
layoutName = 'Dive Map layout'
```

- 5) Editing the map title:
  - Each time a new ROV dive map is created, the title of the dive can manually be edited in the “Title” section of the script.

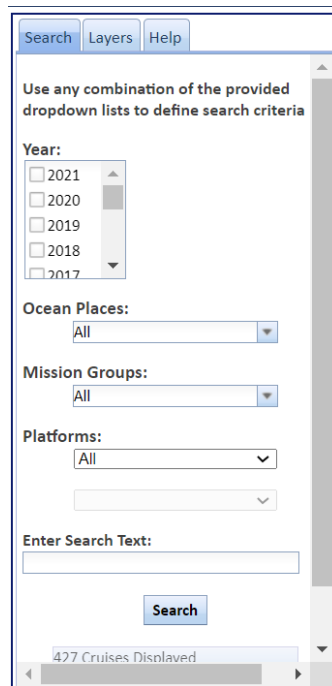


**Code:**

```
title.setText("EX 1903 L2 Dive 19")
```

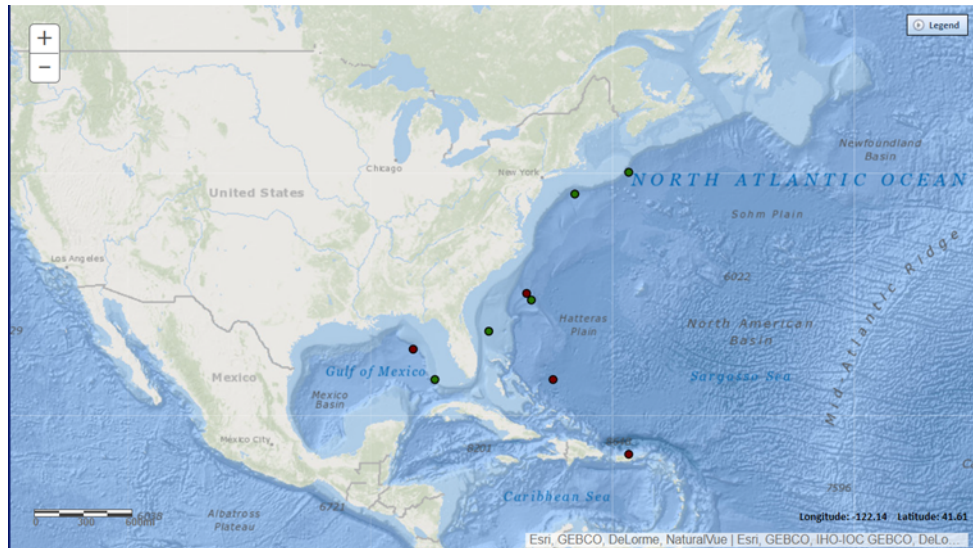
6) Editing the “Date”, “Max Depth”, and “Bottom Time” of the ROV dive:

- In this portion of the script, the date, maximum depth, and total bottom time for each dive can be specified.
- This information for each dive can be accessed through NOAA’s OER Digital Atlas. (<https://www.ncei.noaa.gov/maps/oer-digital-atlas/mapsOE.htm>.) This Digital Atlas is an online map portal that allows the public access to data that is collected during OER expeditions.
- When the digital atlas is loaded, you will see a search box located on the left side of the screen.



The screenshot shows the search interface of the NOAA OER Digital Atlas. At the top, there are three tabs: "Search", "Layers", and "Help". Below the tabs, a message states: "Use any combination of the provided dropdown lists to define search criteria". The search criteria section includes several dropdown menus: "Year:" with a list of years from 2017 to 2021 (2021 is selected), "Ocean Places:" with a dropdown set to "All", "Mission Groups:" with a dropdown set to "All", and "Platforms:" with a dropdown set to "All". Below these is a text input field labeled "Enter Search Text:". A blue "Search" button is positioned below the text field. At the bottom of the search panel, a status bar indicates "427 Cruises Displayed".

- Use this search box to search for the OER expedition of interests based on year, place, mission group, and/or platform.
- When all parameters inside the search box are selected click “Search”. The map frame will then display the queried OER expeditions.



- From the map, select the point for expedition of interests and click on it. This will display a window with the details of the expedition.
- Click on the tab labeled "ROV Data Access." Under "ROV Summary Products," click on "ROV Data Access by Dive".
- When a "ROV Data Access by Dive" is selected a new window will be displayed, which will allow you to select any dive that was conducted during the expedition you selected in the map frame.
- Select the dive that the map is being created for.
- When you have selected the dive, you will see an "Overview" section.
- In this overview section you will find "Date", "Max Depth", and "Bottom Time"
- Once you have updated the "Date", "Max Depth", and "Bottom Time" for the automated map, you are ready to set the name for the final map .pdf, as well as the folder you wish to store your final map in.

**Code Example for "Date":**

```
map_label.setText("Date: -----July 11, 2019")
```

**Code Example for "Max Depth":**

```
map_label.setText("Max Depth:      1623 Meters")
```

**Code Example for "Bottom Time":**

```
map_label.setText("Bottom Time: 5 Hours 58 Minutes 00  
Seconds")
```

**7) Creating the final .pdf map and output folder:**

- Navigate to the directory where you have been saving previous data and files in for this project.

- Create a new folder titled “Final Maps”.
- In *Map production script.py* set the variable *fn* as a path to the “Final Maps” folder that was just created.
- At the very end of the path, assign the desired name for the .pdf map that will be created. In the example below the map is named “Divetest.pdf”

### Final Output Directory Example:

```
# This final section of code creates a final PDF document of the map.
# fn should equal your current working folder, or a desired folder you wish to store the final PDF output of the map.
fn = "C:\\Users\\Jakeb\\OneDrive\\Desktop\\Graduate Research- Desktop\\NOAA Final Maps\\EX 1903 L2\\Divetest.pdf"
#exporter.exportToImage(fn, QgsLayoutExporter.ImageExportSettings())
exporter.exportToPdf(fn, QgsLayoutExporter.PdfExportSettings())
```

### 8) Example Final Map:

